

# Internet-of-Things (IoT)

Introduction



What is the Internet-of-Things?

# How Does My Fridge Do That?

- You are leaving the home (sense user)
- There's no milk in fridge (sense object)
- Use this information to make a decision (process)
- Inform user of decision (communicate)

You are leaving the home (sense user)

There's no milk in fridge (sense object)

- What type of sensor?
- Is milk needed?
- No milk or “little” milk? (prediction)

Use this information to make a decision (process)

Inform user of decision (notify)

# How Does My Fridge Do That?

You are leaving the home (sense user)

- What type of sensor?
- Distinguish between parent and child
- Identify reason for leaving home
- Identify other contexts (e.g., store hours)

There's no milk in fridge (sense object)

Use this information to make a decision (process)

Inform user of decision (notify)

# How Does My Fridge Do That?

You are leaving the home (sense user)

There's no milk in fridge (sense object)

**Use this information to make a decision (process)**

- Where is processor?
- What are the rules?
- Fixed rules versus dynamic rules (learning)

Inform user of decision (notify)

# How Does My Fridge Do That?

You are leaving the home (sense user)

There's no milk in fridge (sense object)

Use this information to make a decision (process)

### **Inform user of decision (notify)**

- How?
- When?
- Privacy?
- Subtleness?
- Information overflow?

# How Does My Fridge Do That?

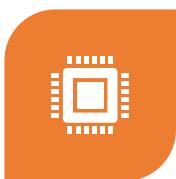
# Internet-of-Things (IoT)

Physical object (“thing”)  
+  
Controller (“brain”)  
+  
Sensors  
+  
Actuators  
+  
Networks (Internet)



# Related Areas/Terminology

---



**EMBEDDED SYSTEMS:**  
NOT NECESSARILY  
CONNECTED



**SENSOR NETWORKS:**  
COLLECTION OF  
SENSOR DEVICES  
CONNECTED THROUGH  
WIRELESS CHANNELS



**CYBER-PHYSICAL  
SYSTEMS:** FOCUS ON  
INTERACTION  
BETWEEN PHYSICAL  
AND CYBER SYSTEMS



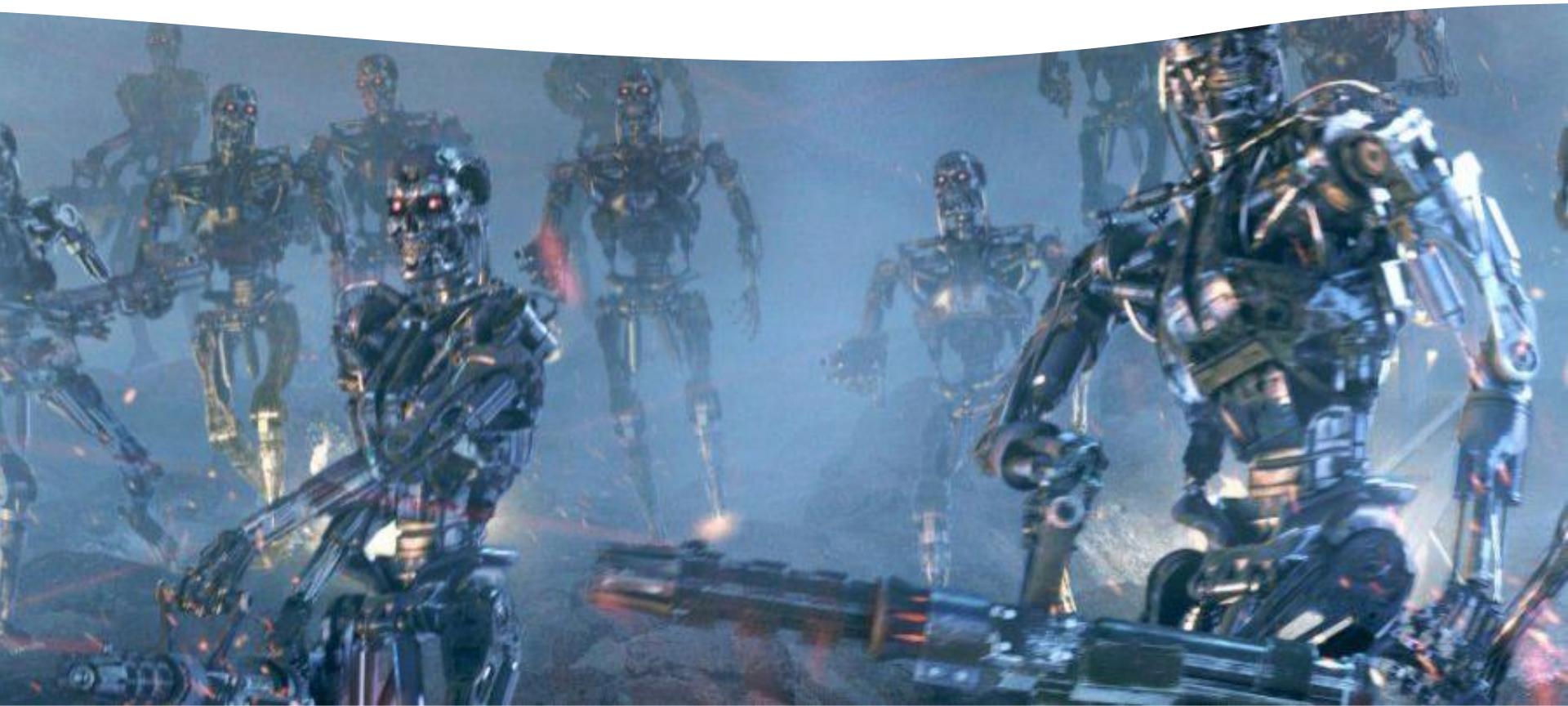
**REAL-TIME SYSTEMS:**  
FOCUS ON TIME  
CONSTRAINTS



**PERVASIVE/UBIQUITOUS COMPUTING:**  
FOCUS ON  
ANYTIME/ANYWHERE  
COMPUTING

## Related Areas

- Machine-to-machine (M2M) communications
- Internet of Everything (Cisco Systems)
- “Skynet” (Terminator movie)



# “Internet-of- Things”

---

Term coined by British entrepreneur Kevin Ashton, while working at MIT Auto-ID Labs

---

Referred to (and envisioning) a future global network of objects connected specifically by RFID (radio-frequency identification)

---

Complete automation of data collection

---

First article about IoT in 2004 from MIT; called it ‘Internet 0’.

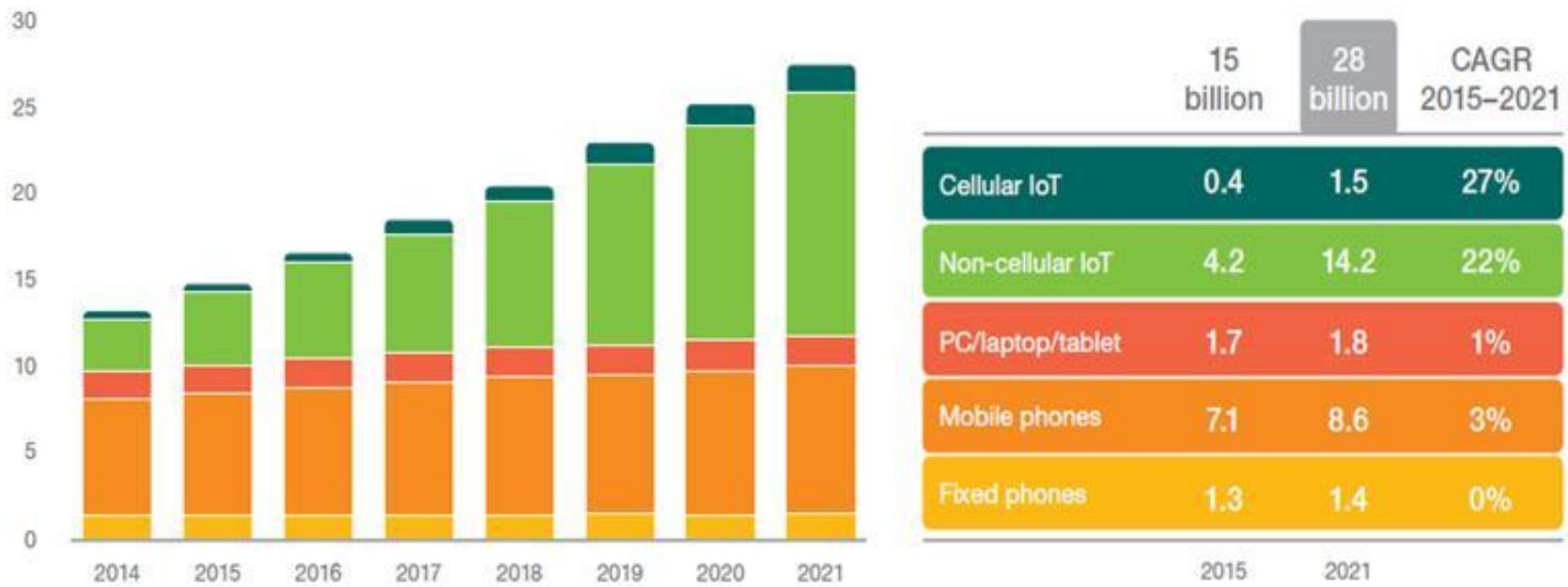
---

\*[https://en.wikipedia.org/wiki/Internet\\_0](https://en.wikipedia.org/wiki/Internet_0)

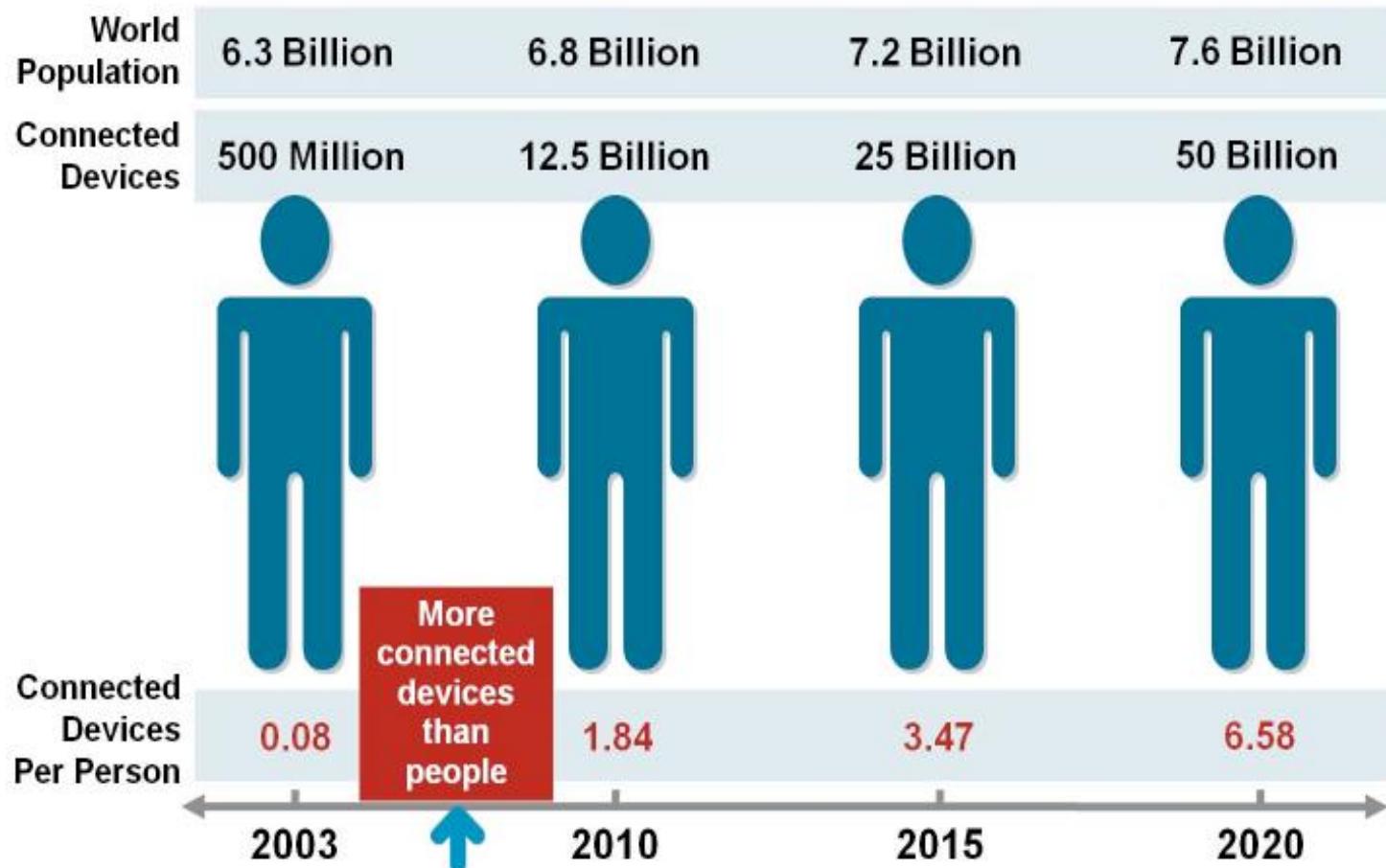
## Internet-of-Things Vision & Growth

# THE INTERNET OF THINGS

Connected devices (billions)



# Internet-of-Things Vision & Growth



Source: Cisco IBSG, April 2011

# What is IoT

- Internet of things (IoT) is the network of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these things to connect, collect and exchange data<sup>1</sup>.
- IoT refer to the connection of devices to the Internet.

<sup>1</sup>[https://en.wikipedia.org/wiki/Internet\\_of\\_things](https://en.wikipedia.org/wiki/Internet_of_things)

# Internet-of-Things (IoT)

Introduction

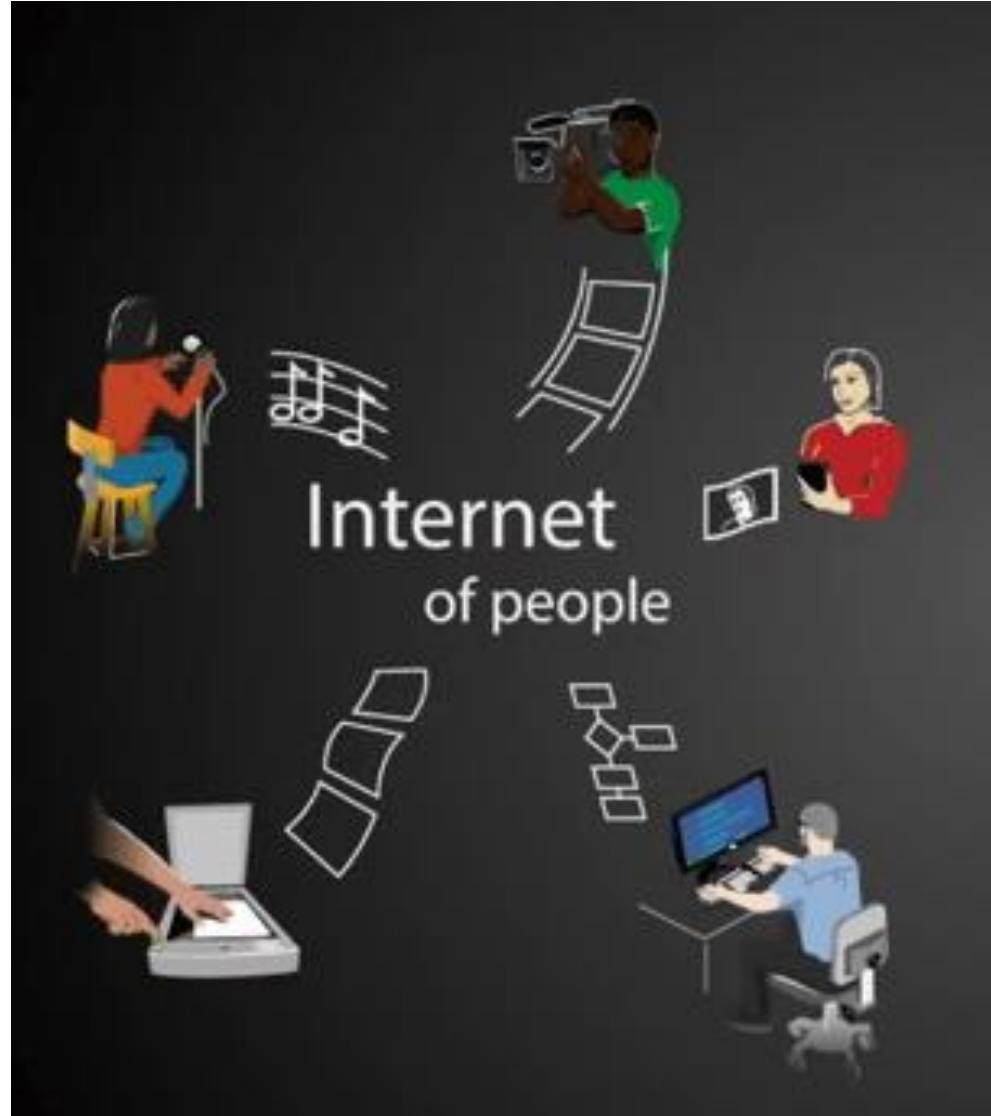
# What is IoT

- Internet of things (IoT) is the network of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these things to connect, collect and exchange data<sup>1</sup>.
- IoT refer to the connection of devices to the Internet.

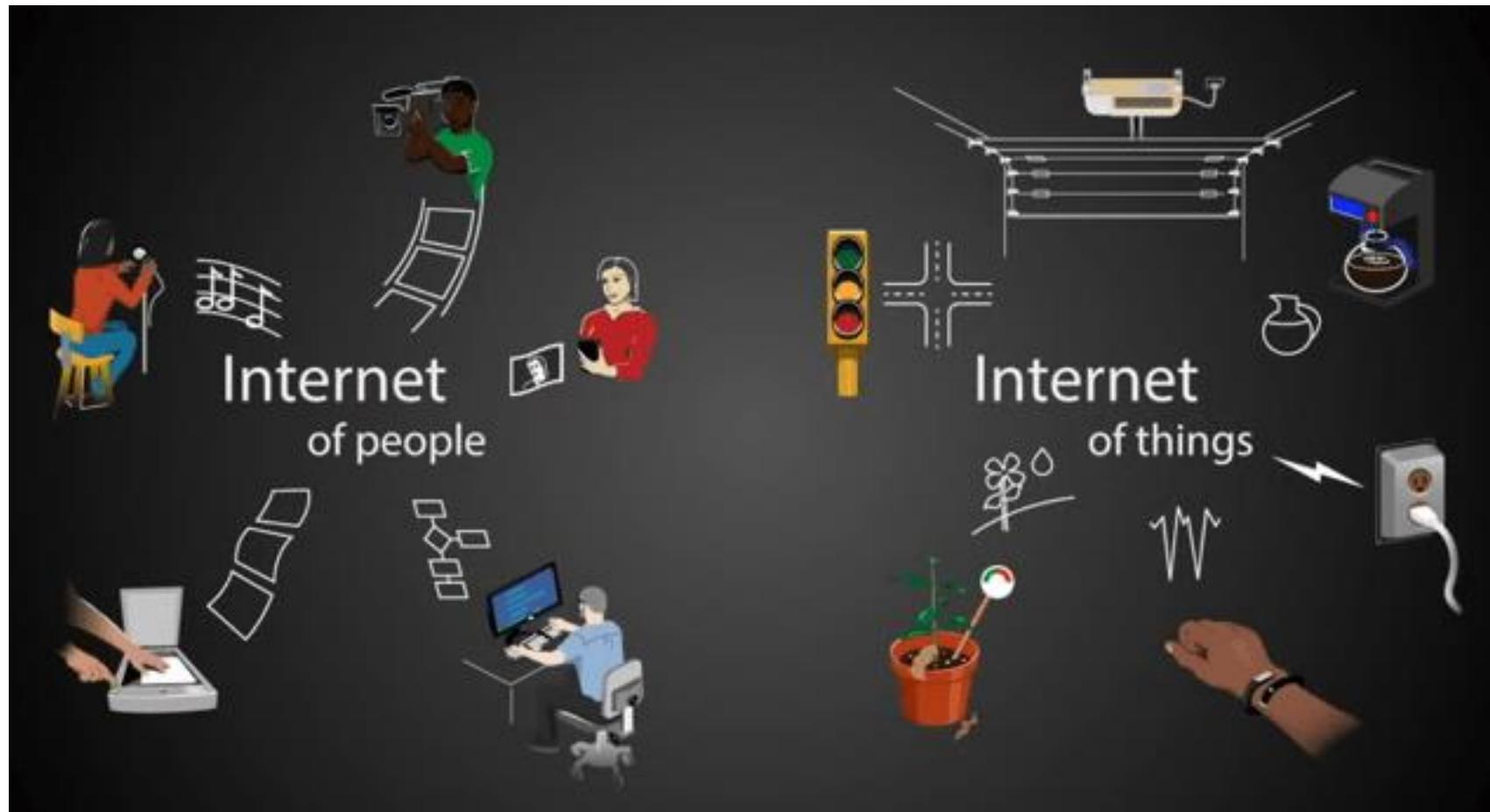
<sup>1</sup>[https://en.wikipedia.org/wiki/Internet\\_of\\_things](https://en.wikipedia.org/wiki/Internet_of_things)

## Internet of People (IOP)

- People are connected with the Internet.
- Internet is everywhere in the World.
- It is the primary connection between people.

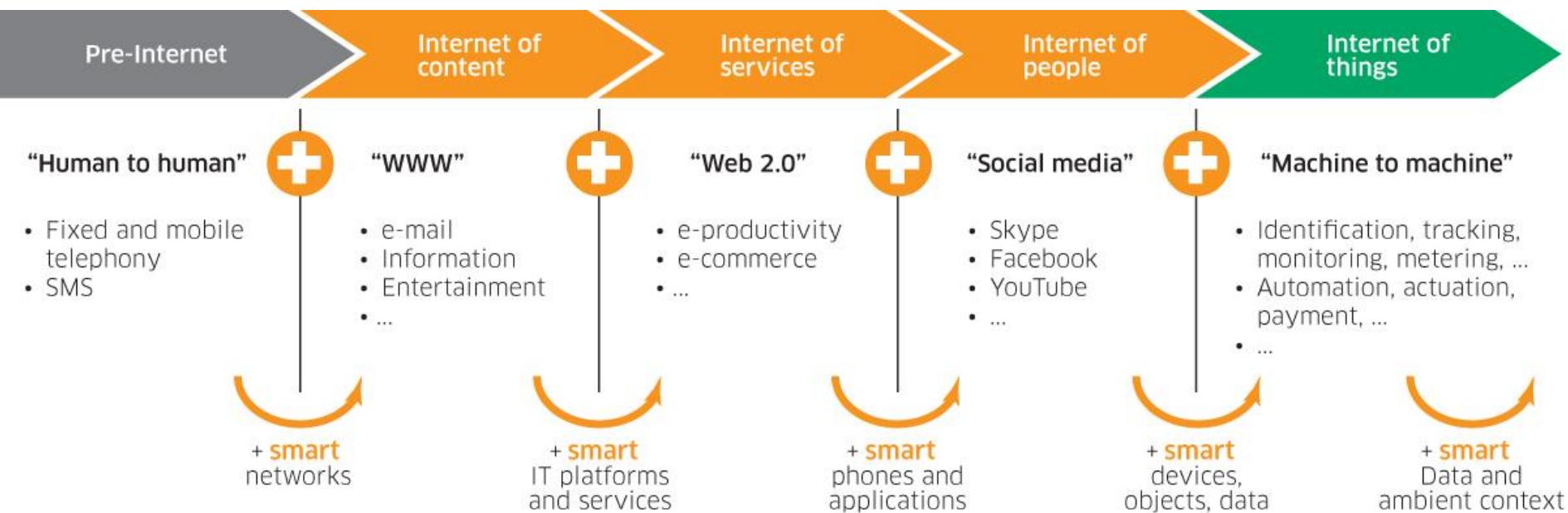


IOP

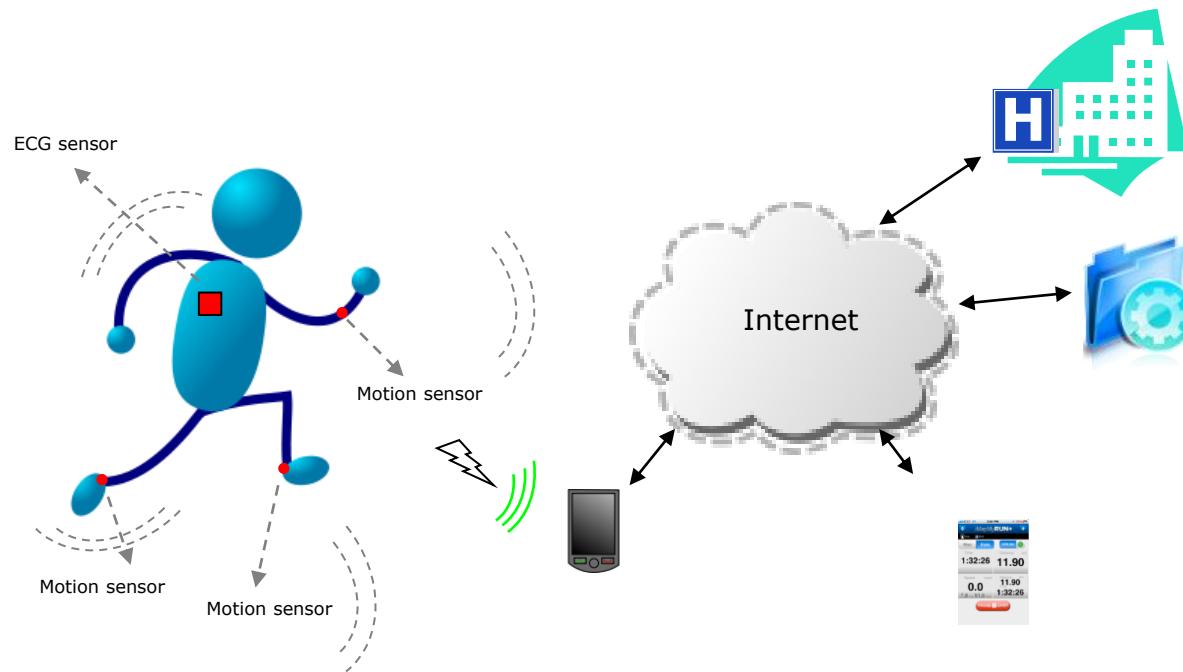


IOT

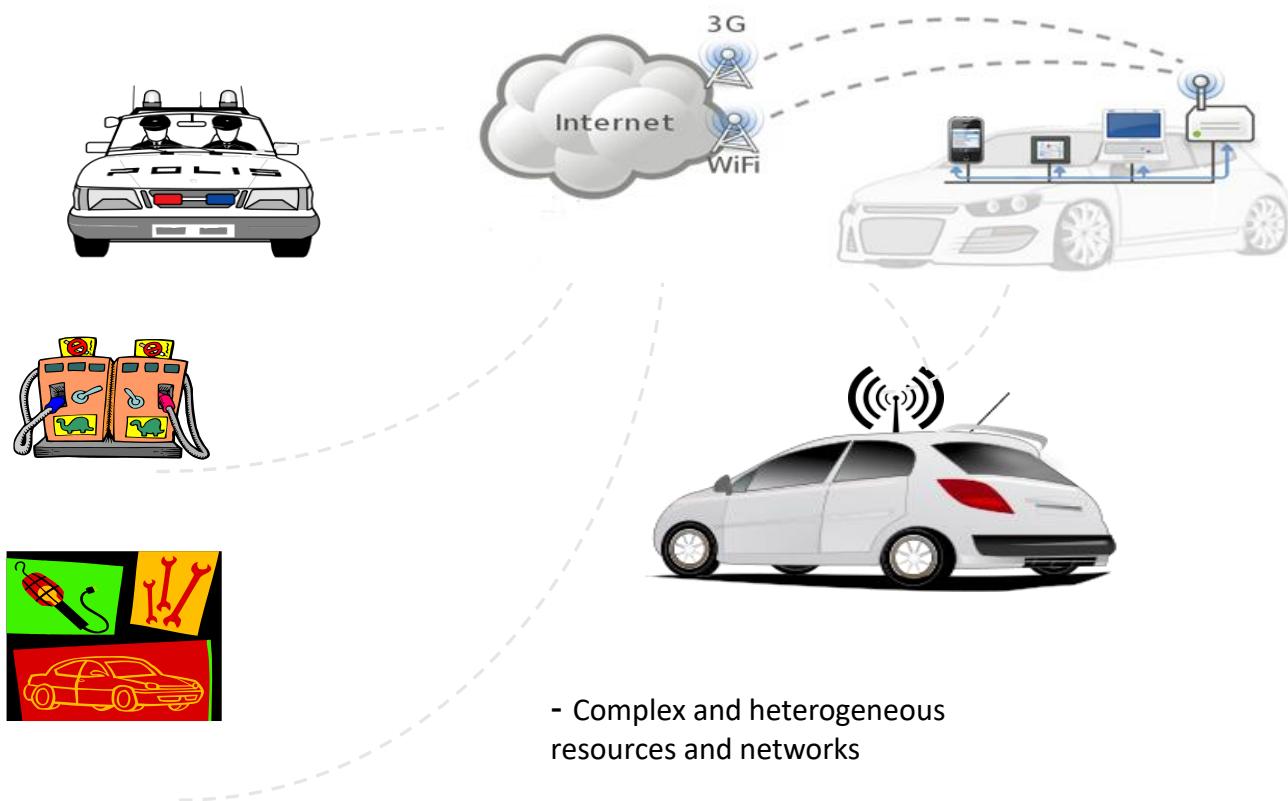
# Internet-of-Things Evolution



# People Connecting with Things



# Things Connecting with Things



# Where is IOT?

---

IOT is everywhere!

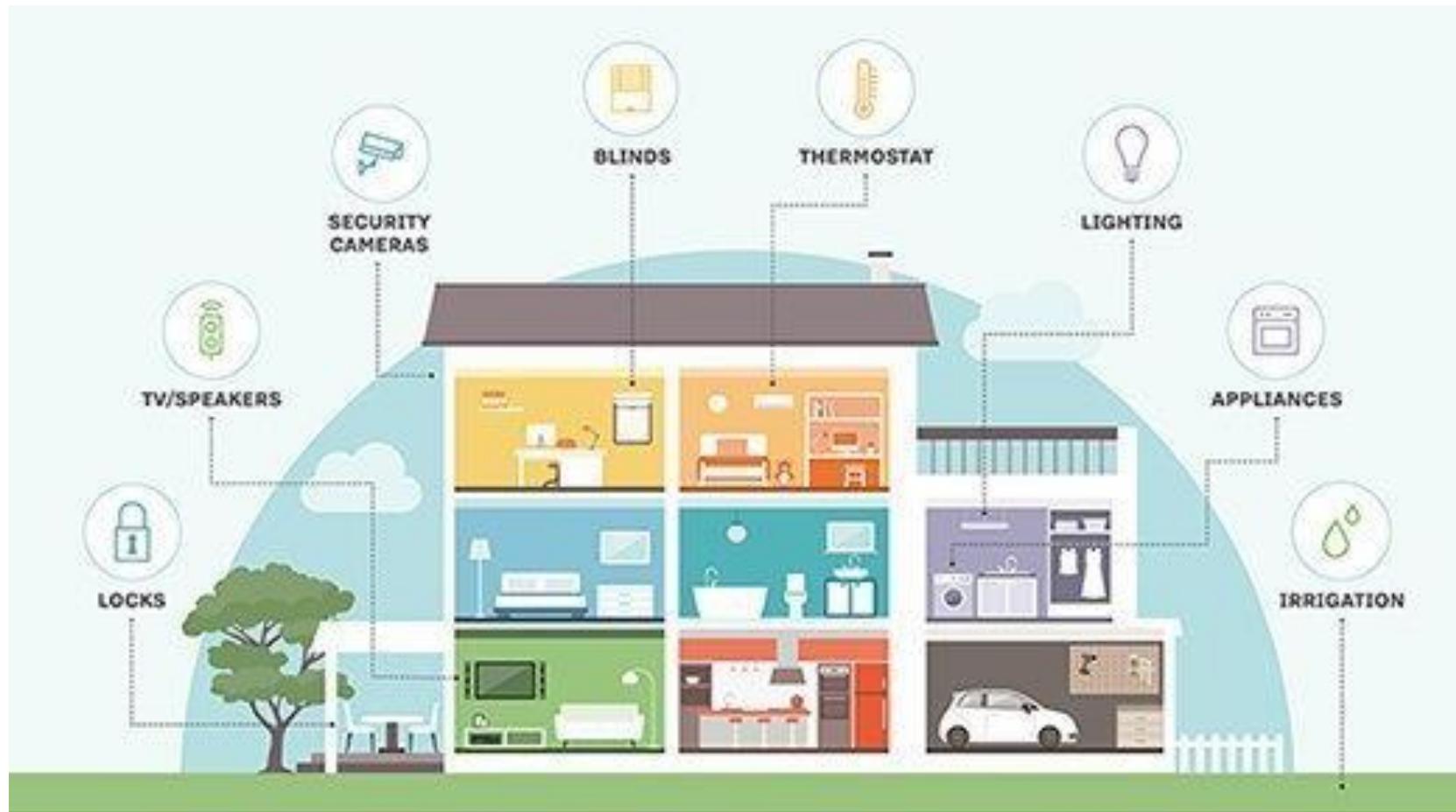




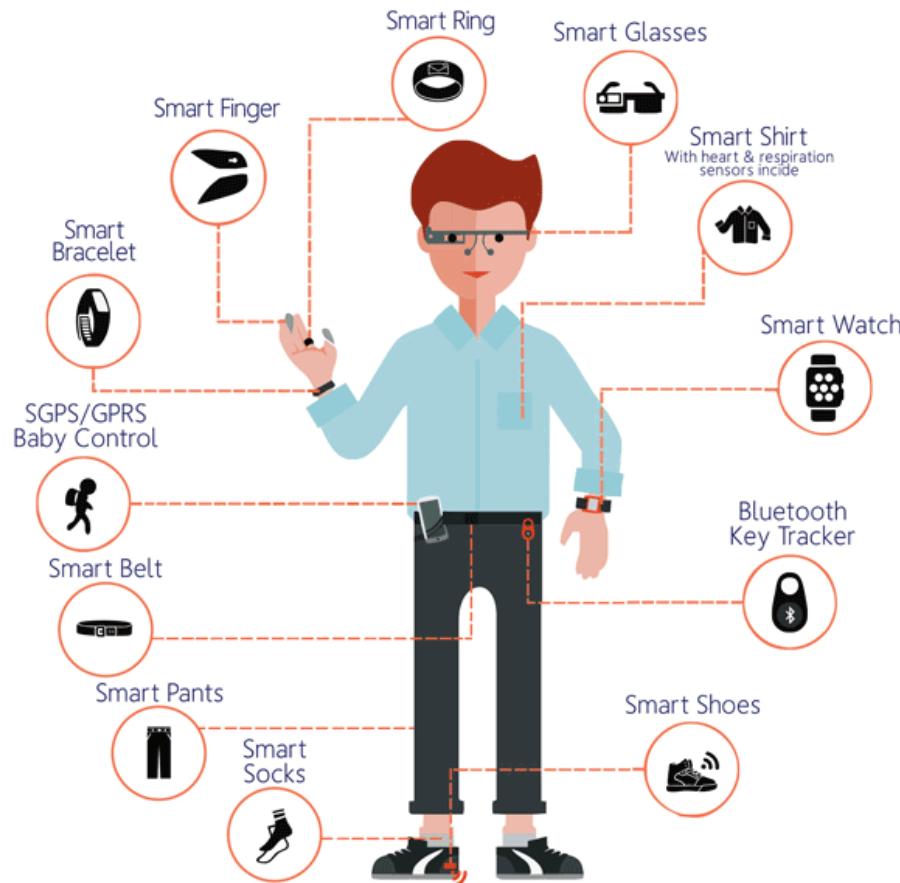
A large blue circle is positioned in the upper left area. To its right is a smaller yellow circle, and further to the right is a very small gray circle. A large dark gray shape, resembling a semi-circle or a large oval, is located in the upper right corner, partially overlapping the yellow circle.

Applications |

# Smart Homes

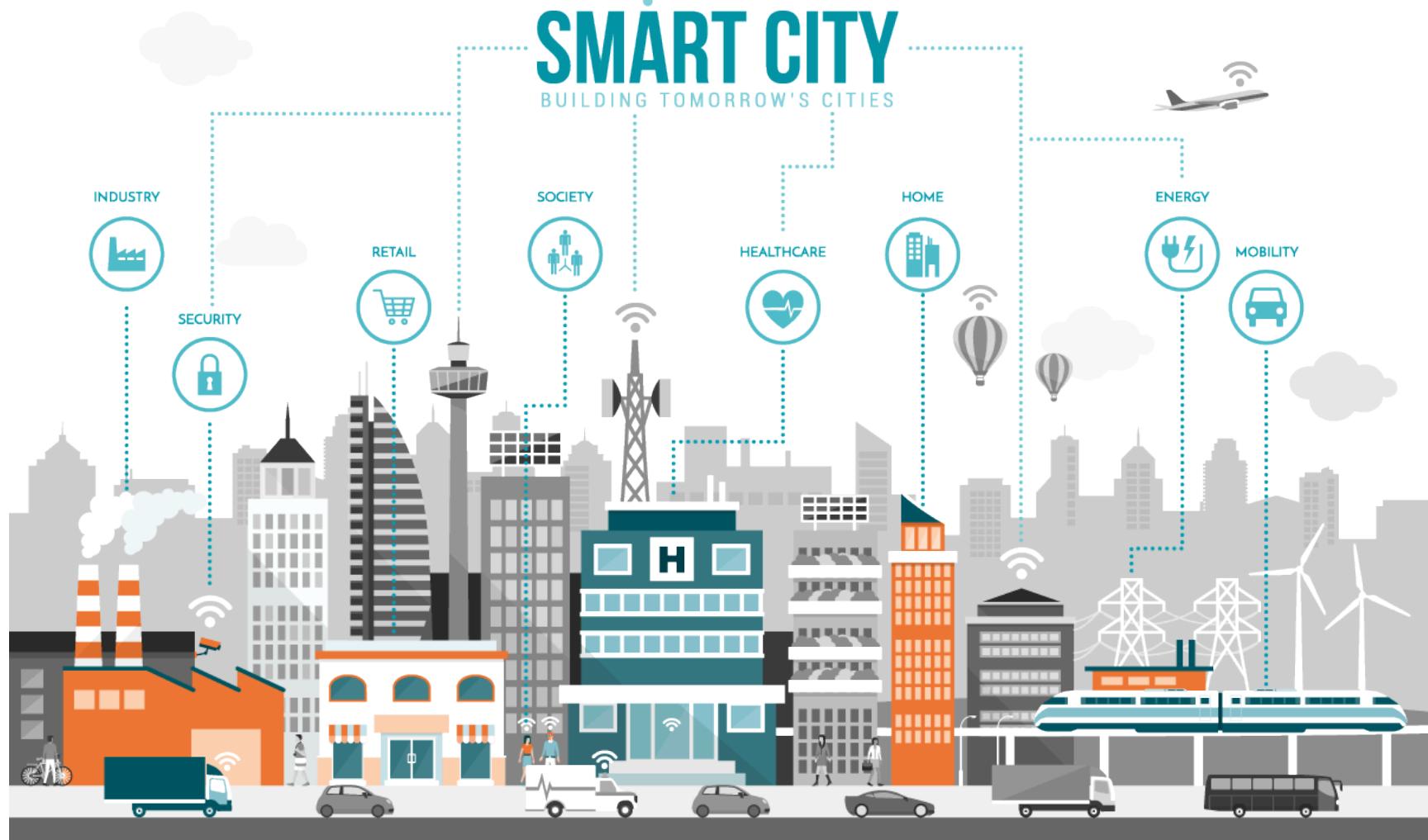


# Wearable



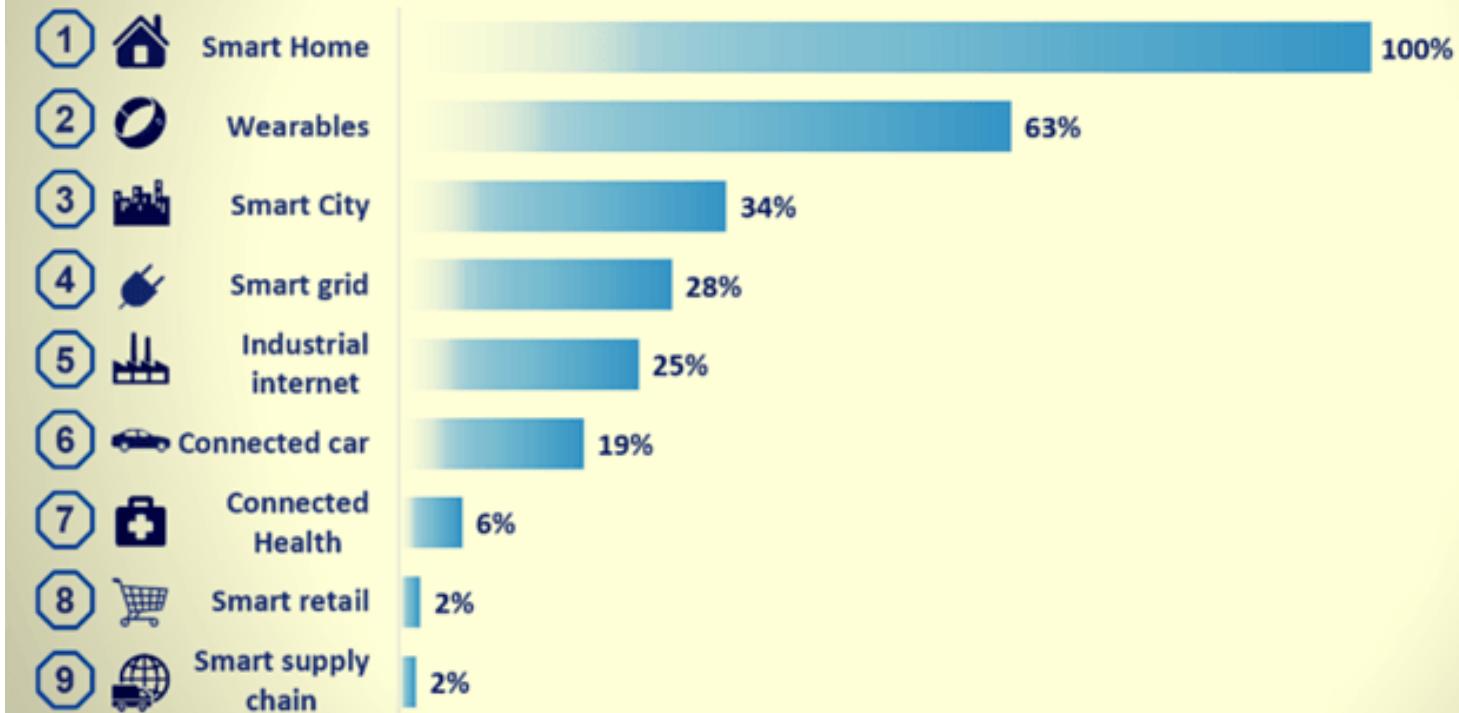
# SMART CITY

BUILDING TOMORROW'S CITIES



## The 10 most popular Internet of Things applications

A ranking based on web analytics



# Augment Existing Things

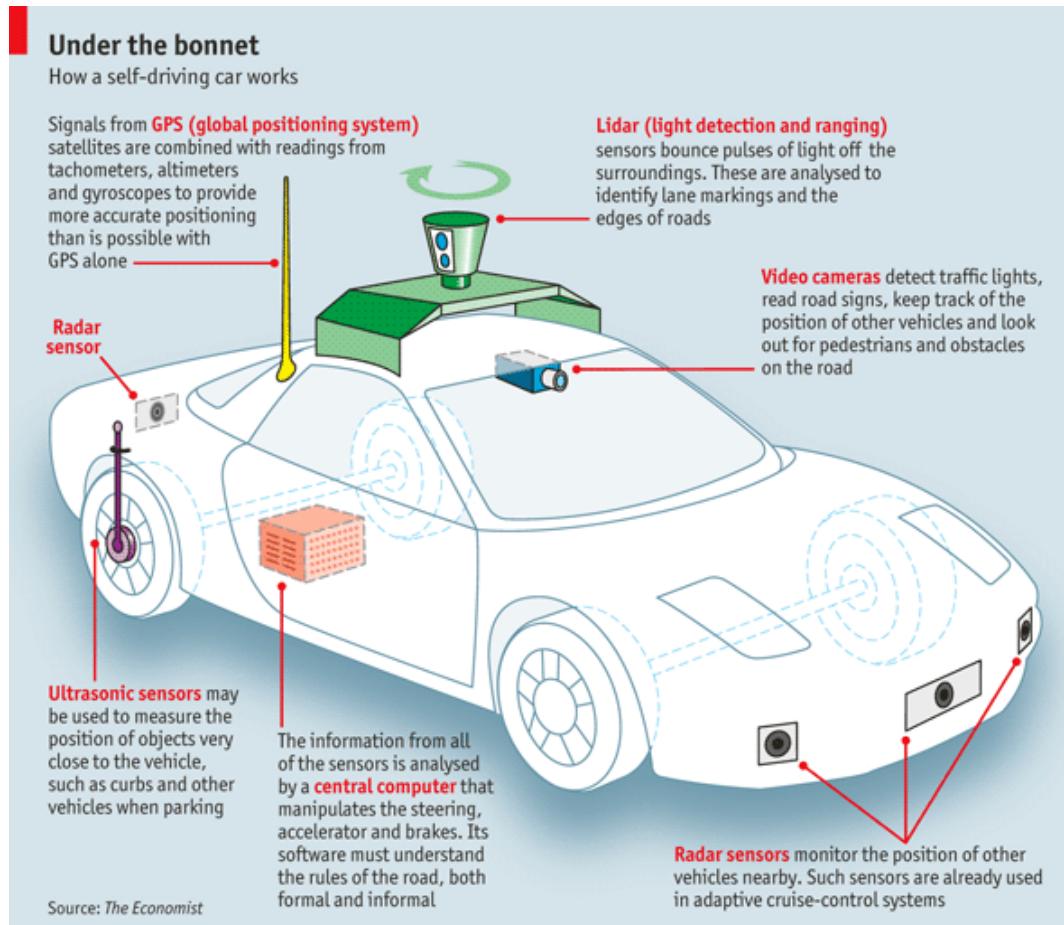




## Augmenting Life With New Things

- Smart City
- Smart Car
- Smart Me (healthcare, fitness, wellness)

# Example: Connected Roadways



# Example: Connected Roadways

[State of Self-Driving Car](#)

# The Connected Factory in Action



## INNOVATION

### TAP COMMERCIAL INNOVATION

Mobilize employees and supervisors to move across the factory floor and access data wherever they are. The iPad and other like devices are making their way into industrial settings – along with an expectation that much of the commercial innovation it brings will also apply to industrial activities.

### STATUS



"I see the fix!"



## EFFICIENCY

### LINK INFORMATION & OPERATIONAL TECHNOLOGY

Bridge the gap from data center to control room to collaborate and share best practices and common goals between manufacturing and IT.

### Systems Processes People



### OPTIMIZE ASSETS

Identify where your people, equipment, works in process and finished goods are in real-time. Adjust the schedule and inventory on the fly.



## AGILITY

### CONNECT & COLLABORATE EXTERNALLY

Extend visibility beyond your four walls. Link the extended supply chain and distribution to create dynamic workflows. Help and expertise are available in an instant.



**EXPANDABLE INFRASTRUCTURE**  
Design and build an Industrial Ethernet infrastructure to minimize cost and effort to expand or improve processes. One infrastructure for safety, control, SCADA, Physical Security, and LAN.



## RISK

### SECURE PHYSICAL & CYBER ASSETS

Traditional security devices, like keypad entry systems, call boxes and security cameras, need power from Industrial Ethernet cables, with secure networks, to protect your processes, people, and plans from cyber sabotage.



### MAXIMIZE UPTIME

Design ruggedized industrial networking infrastructure that will endure in harsh environments with redundant communications, power and configuration backup – especially for business processes under extreme conditions.

**BELDEN**  
SIMPLIFYING THE MIGHTY CONNECTION

## Example: Connected Factory

- New product and service introductions faster
- Increasing production, quality, uptime
- Mitigating unplanned downtime
- Protecting from cyber threats
- Worker productivity and safety

# Example: Smart & Connected Buildings

- Energy management
- Lighting
- Safety
- HVAC
- Building automation
- Smart spaces

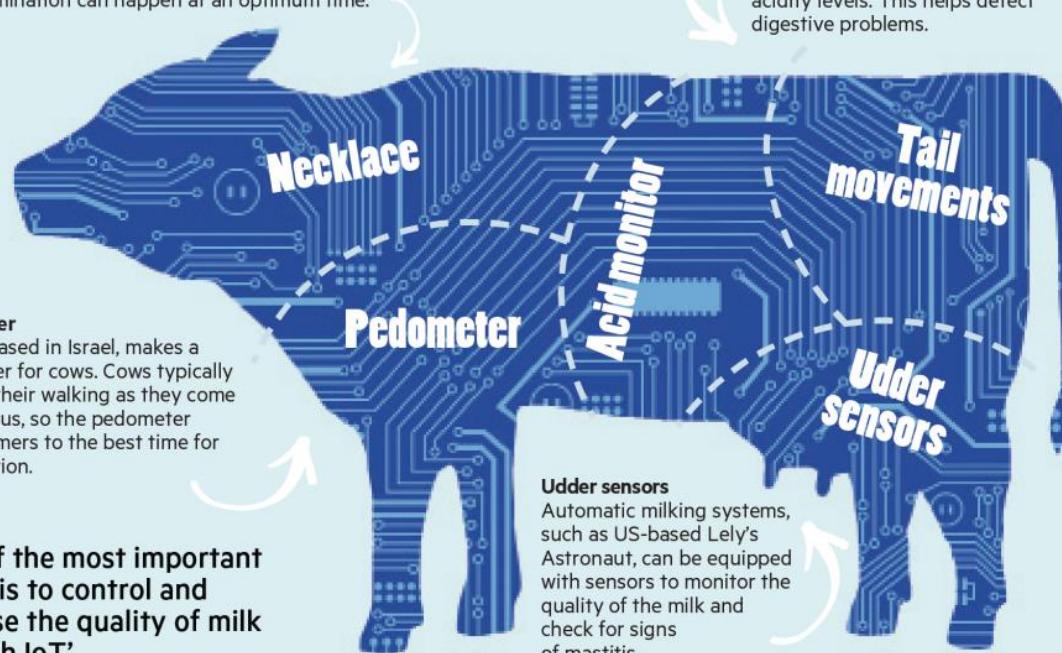


# Example: Smart Creatures

## The connected cow

### Necklace

Connecterra, a Dutch company, makes Fitbit-style necklaces that monitor a cow's movement and feeding habits. The sensor can be used to detect health problems and to tell when the cow is in heat, so that insemination can happen at an optimum time.



### Pedometer

Afimilk, based in Israel, makes a pedometer for cows. Cows typically increase their walking as they come into oestrus, so the pedometer alerts farmers to the best time for insemination.

**'One of the most important issues is to control and increase the quality of milk through IoT'**

### Acid monitor

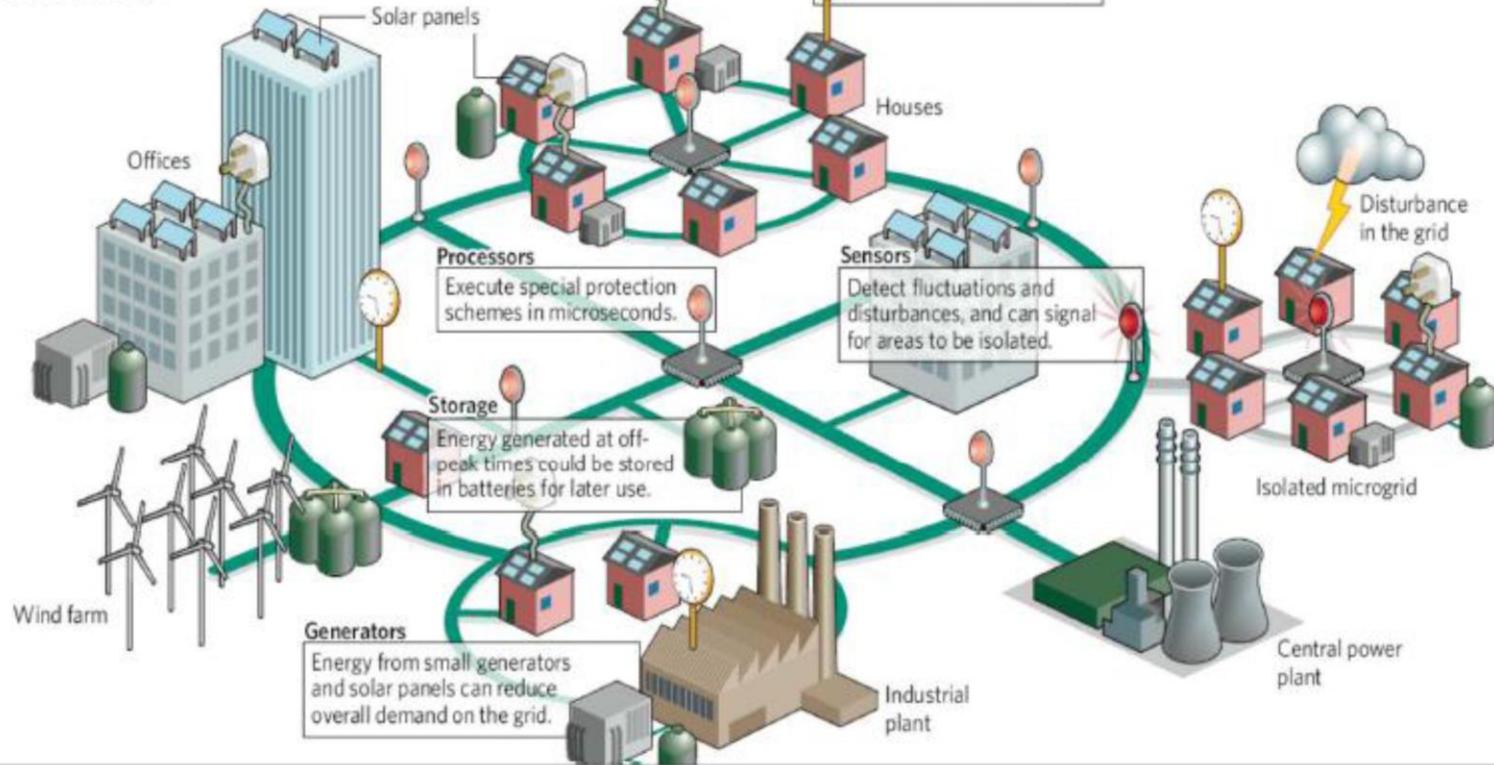
Well Cow, a British company, has developed a bolus that is inserted into the cow's rumen to monitor acidity levels. This helps detect digestive problems.

### Tail movements

Moocall, an Irish company, makes a birthing sensor that attaches to the tail. It measures tail movements triggered by labour contractions, and sends a farmer an SMS alert approximately one hour before a cow is due to calve.

## SMART GRID

A vision for the future — a network of integrated microgrids that can monitor and heal itself.



## Example: Smart Grid



## Enablers: Portability

Reducing the size of hardware to enable the creation of computers that could be physically moved around relatively easily



## Enablers: Miniaturization

Creating new and significantly smaller mobile form factors that allowed the use of personal mobile devices while on the move



50mm x 50mm

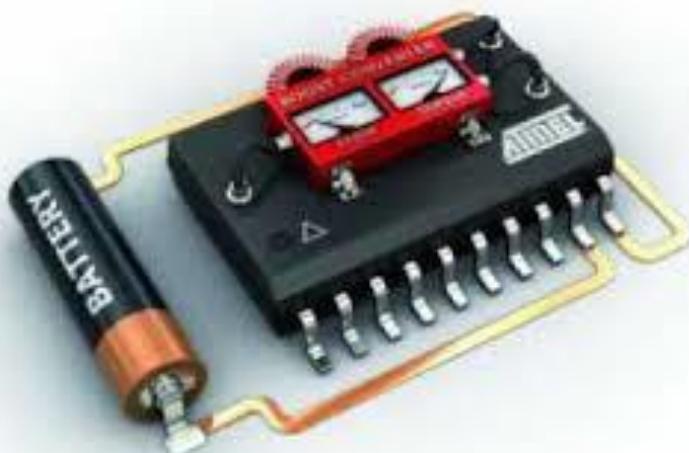


35mm x 35mm



15mm x 15mm

# Enablers: Low Power and Low Heat



- Low power architectures
- Low power radios
- Sleep modes
- Energy harvesting

# Enablers: Connectivity

- Developing devices and applications that allowed users to be online and communicate via wireless data networks while on the move



Bluetooth®

# Enablers: Convergence

---

Integrating emerging types of digital mobile devices, such as Personal Digital Assistants (PDAs), mobile phones, music players, cameras, games, etc., into hybrid devices.



# Enablers: Divergence

Opposite approach to interaction design by promoting information appliances with specialized functionality rather than generalized ones



# Enablers: Ecosystems



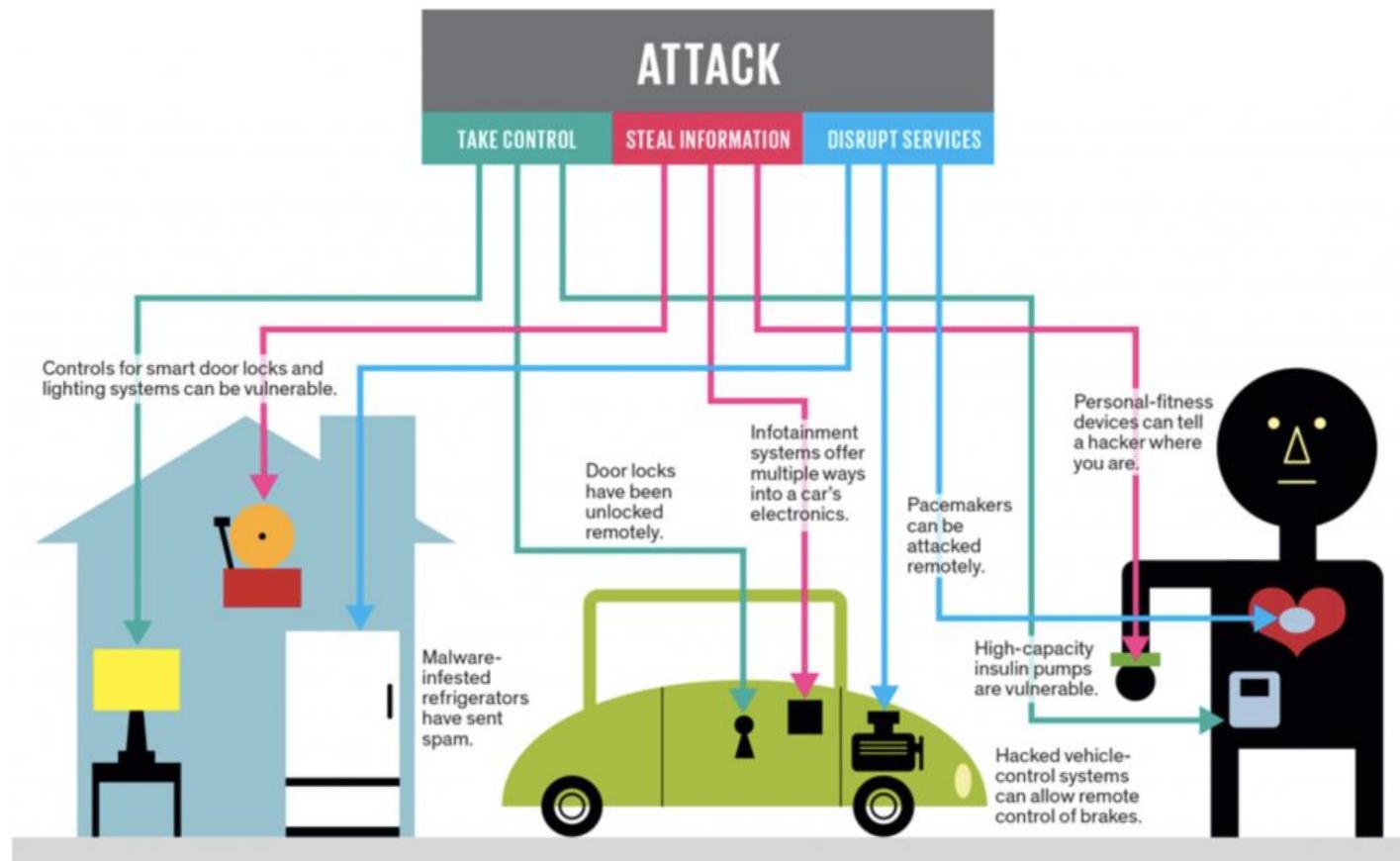
The emerging wave of *digital ecosystems* is about the larger wholes of pervasive and interrelated technologies that interactive mobile systems are increasingly becoming a part of.

# Example: Smartphone

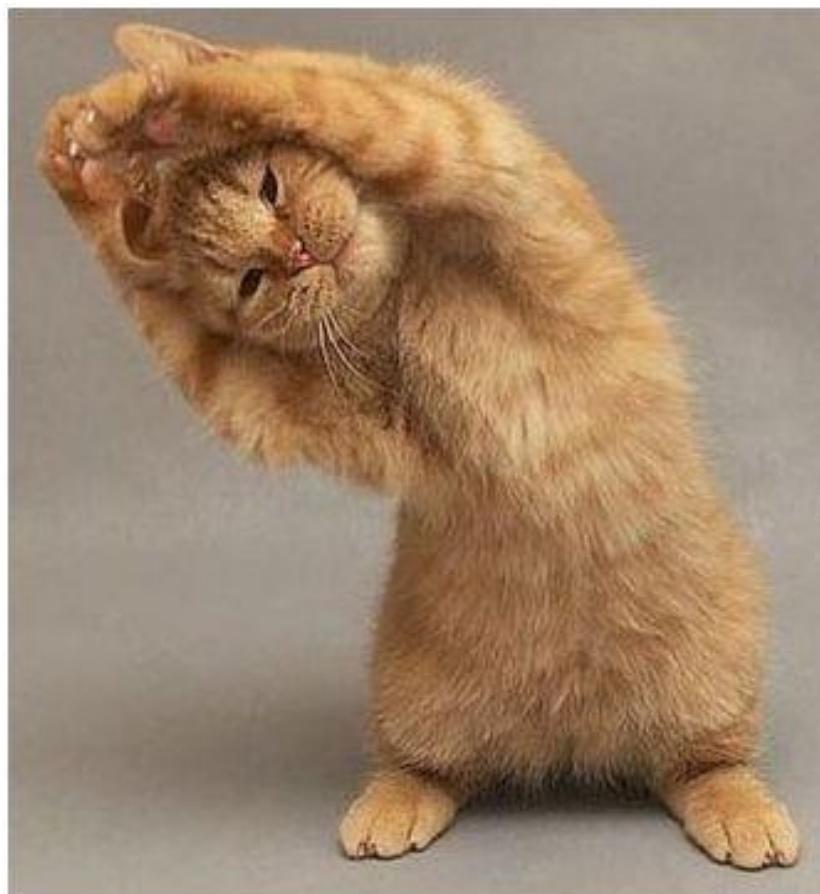
---

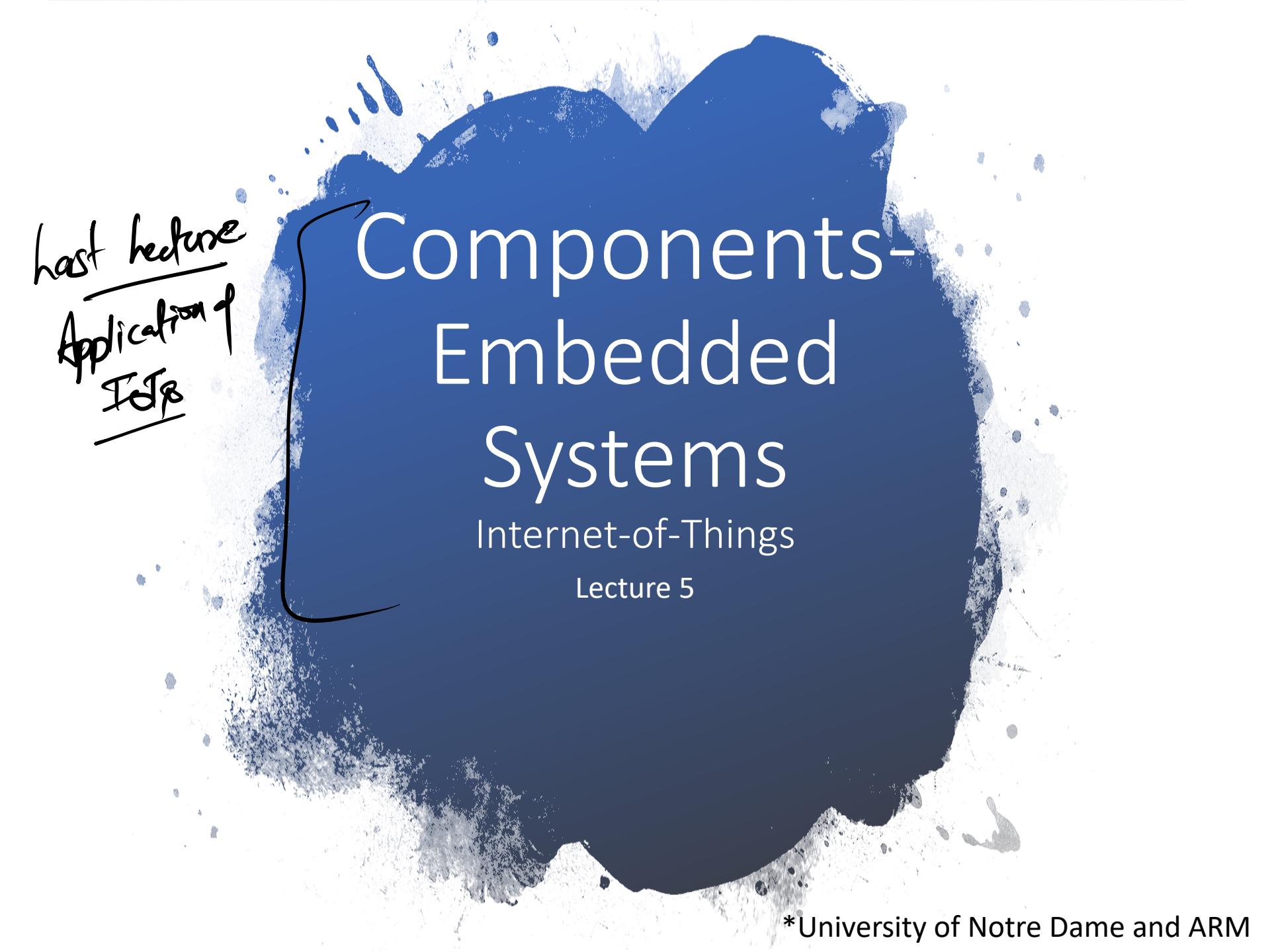
- Portability: carry it anywhere you want
- Miniaturization: make it possible to build device to fit in your pocket
- Connectivity: Wi-Fi, LTE/4G, cellular, Bluetooth
- Convergence: phone, camera, gaming device, movie streaming, music player, ...
- Digital Ecosystem: cloud, social networks, software development kits, app stores, big data, standardization ...

# IoT Issues & Challenges



# BREAK



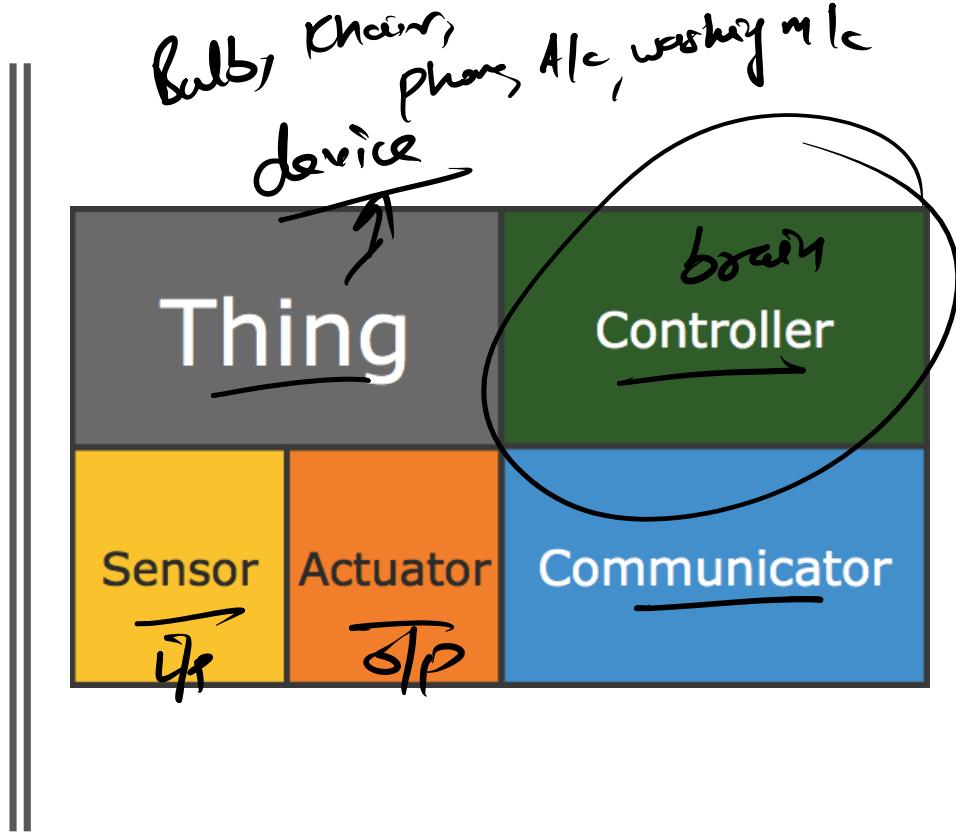


# Components- Embedded Systems

Internet-of-Things

Lecture 5

last lecture  
Application of  
IoT &



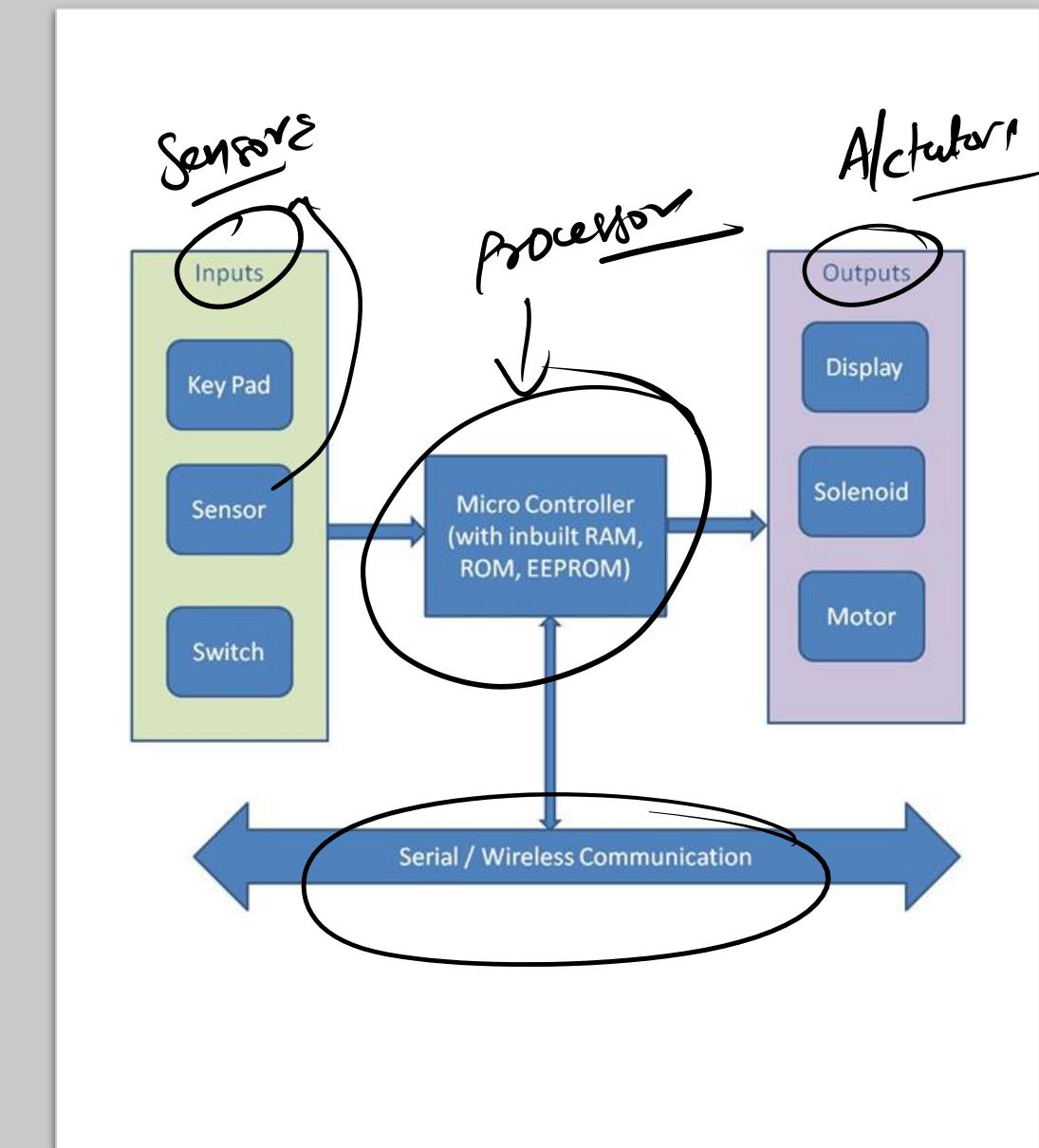
# Components of an IoT Device

# Embedded System/Computer

- “Any sort of device which includes a programmable computer but itself is not intended to be a general-purpose computer” - Wayne Wolf

*specl 2*  
to  
*some app'cns*

- General purpose
- Dedicated



# Embedded systems

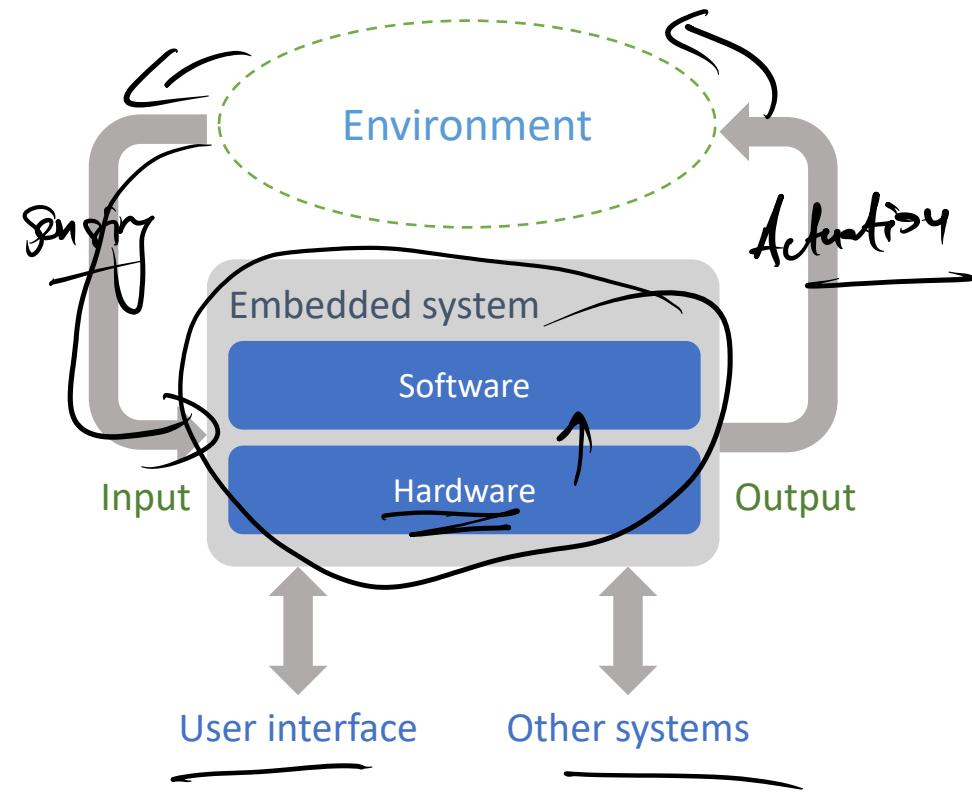
## Internet of Things (IoT)

- Application-specific computer system
- Built into a larger system
- Often with real-time computing constraints

## Adding embedded systems to larger systems

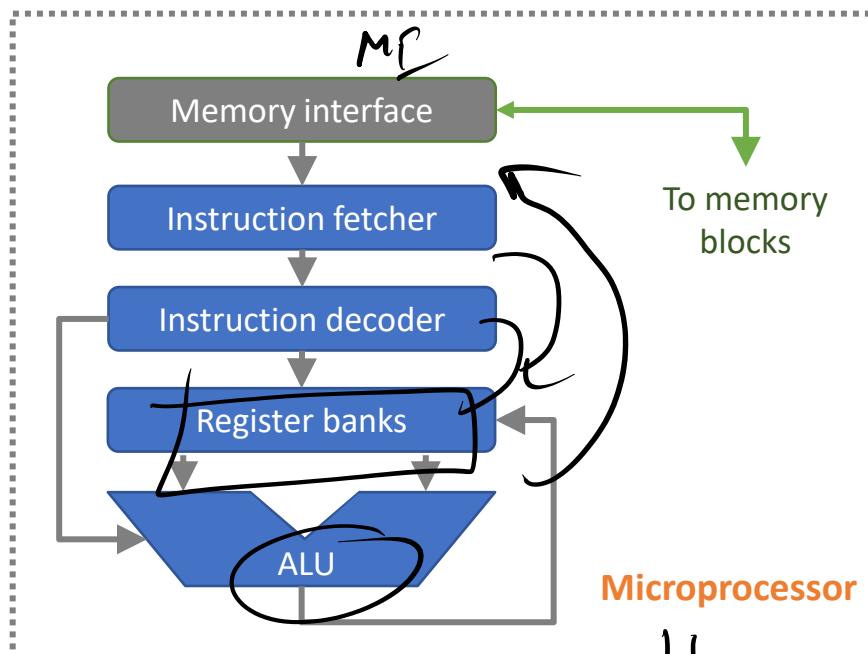
- Better performance
- More functions and features
- Lower cost, for e.g., through automation
- More dependability

**Examples:** smartphones, smart watches,  
printers, gaming consoles, wireless routers



# CPUs → MCUs → Embedded Systems

~~Microprocessor or Central Processing unit (CPU)~~



Defined typically as a single processor core that supports at least instruction fetching, decoding, and executing

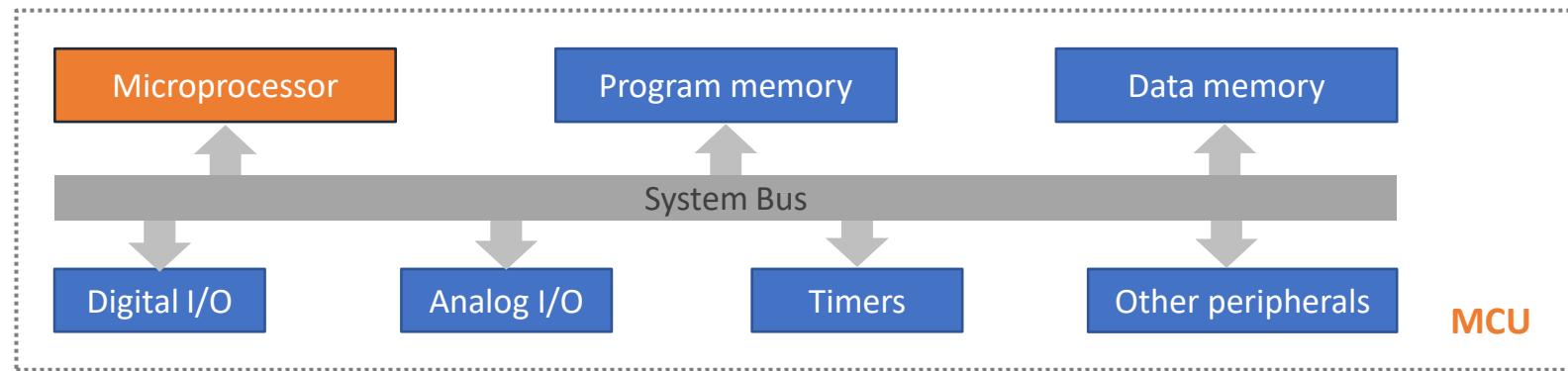


Used for general purpose computing, but needs to be supported with memory and Input/Output (I/O) interfaces



# CPUs → MCUs → Embedded Systems

## Microcontroller Unit (MCU)



- Typically has a single processor core
- Has memory blocks, digital I/Os, analog I/Os, and other basic peripherals
- ⚙️ Used for basic control purposes, such as embedded applications

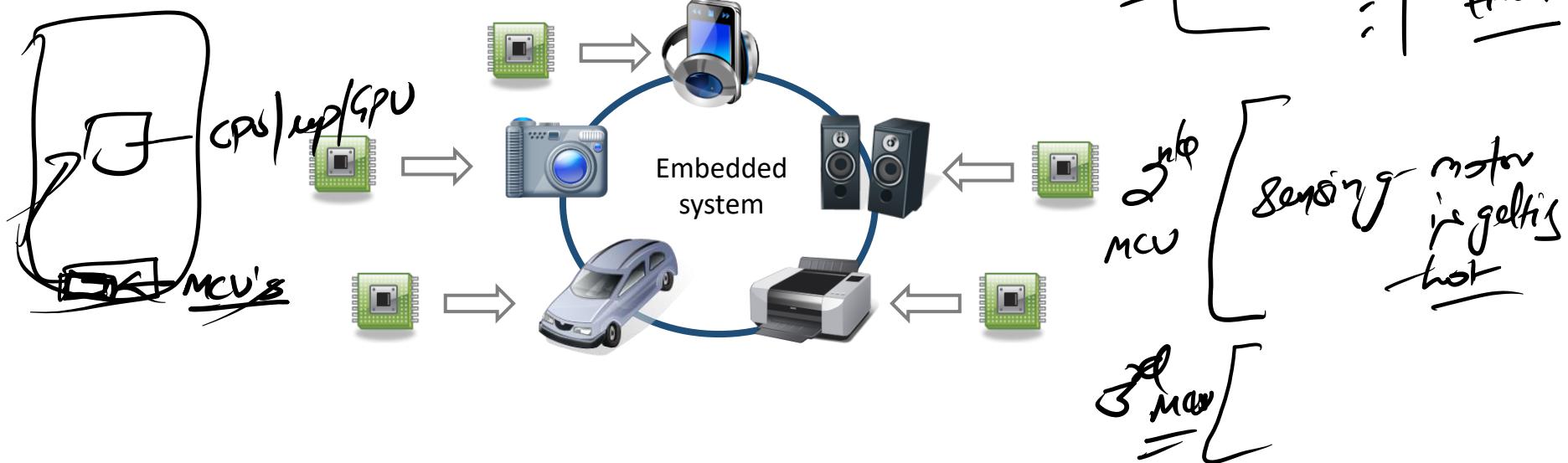
# CPUs → MCUs → Embedded Systems

## Embedded system

Typically implemented using MCUs

Often integrated into a larger mechanical or electrical system

Usually has real-time constraints



# Embedded system example: Bike computer

only  
one  
MCU

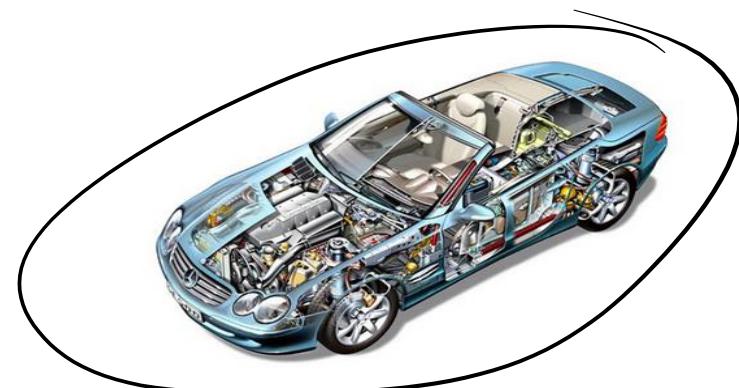
- Functions
  - Speed, cadence, distance, heart rate (HR) measurements
- Constraints
  - Size, weight, and cost; power and energy
- Inputs (Sensor)
  - Wheel rotation sensor and mode key
- Output
  - Liquid crystal display (LCD), BLE interface to smartphone
- Uses low performance microcontroller



# Embedded system example: Gasoline engine control unit

- Functions
  - Fuel injection ✓
  - Air intake setting ✓
  - Spark timing ✓
  - Exhaust gas circulation ✓
  - Electronic throttle control ↗
  - Knock control ✓
- Constraints
  - Reliability in a harsh environment
  - Cost
  - Weight

- Many inputs and outputs
  - Discrete sensors and actuators (MCU)
  - Network interface to rest of the car (MCU)
- Uses high performance microcontroller
  - E.g., 32-bit, 3MB flash memory, 150–300MHz

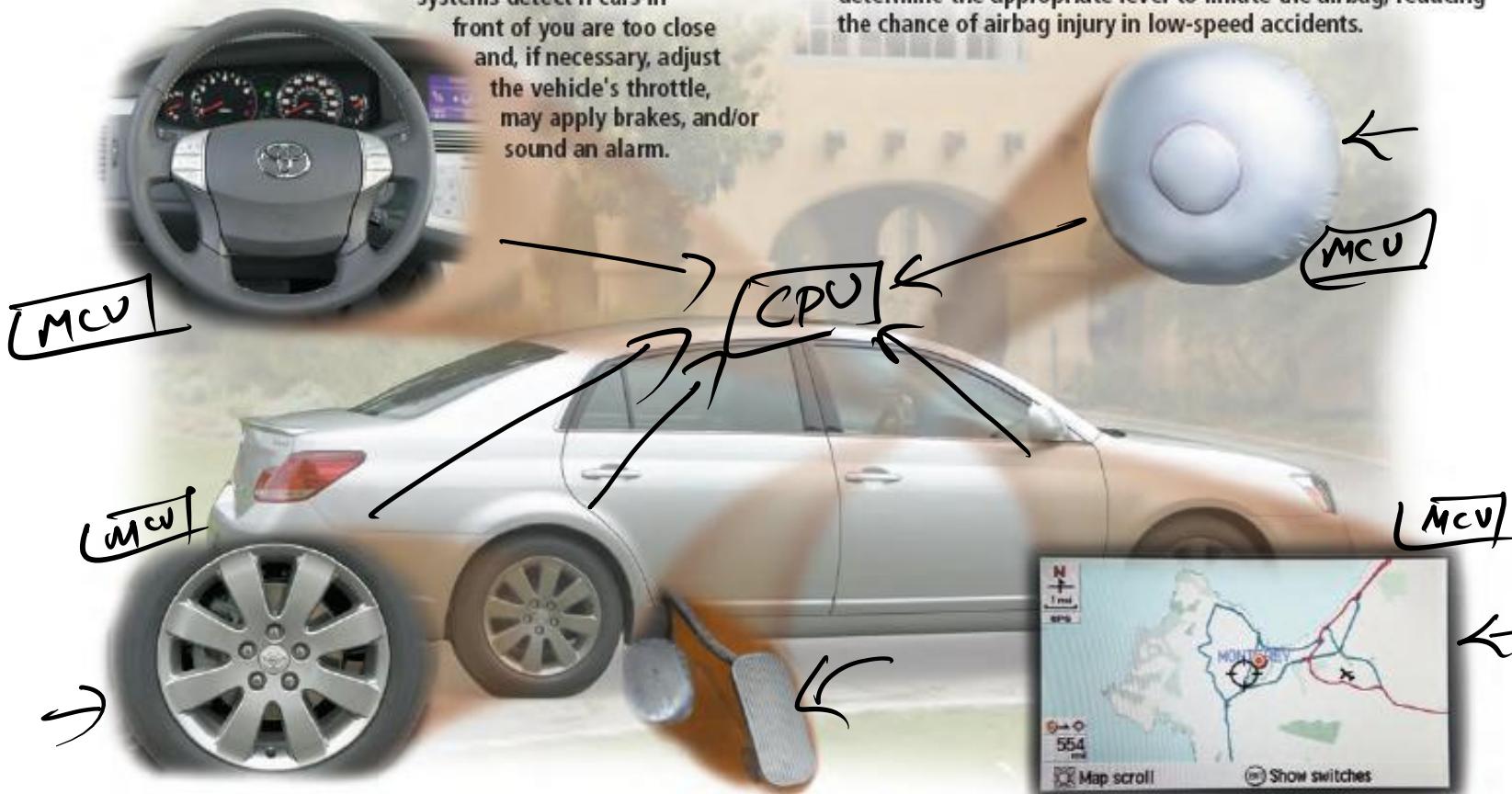


# Automotive Embedded Systems

(Self-drive car)

Adaptive cruise control systems detect if cars in front of you are too close and, if necessary, adjust the vehicle's throttle, may apply brakes, and/or sound an alarm.

Advanced airbag systems have crash-severity sensors that determine the appropriate level to inflate the airbag, reducing the chance of airbag injury in low-speed accidents.



Tire pressure monitoring systems send warning signals if tire pressure is insufficient.

Drive-by-wire systems sense pressure on the gas pedal and communicate electronically to the engine how much and how fast to accelerate.

Cars equipped with wireless communications capabilities, called telematics, include such features as navigation systems, remote diagnosis and alerts, and Internet access.

# Automotive Embedded Systems

- Today's high-end automobile may have 100+ microprocessors:  
Seat belt; dashboard devices; engine control; ABS; automatic stability control; navigation system; infotainment system; collision avoidance system; tire pressure monitoring; lane warning; adaptive cruise control; climate control; airbag control unit; electric window and central locking; parking aid; automatic wiper control; alarm and immobilizer; power seat; electric power steering; electronic transmission; active suspension

Gr.  
Project E

, pedestrian  
Peds  
Stoed L.  
Red L.  
Zebra Cross  
Other cars

side mirror  
Doft mirr-  
A/c

## Embedded Processor Market

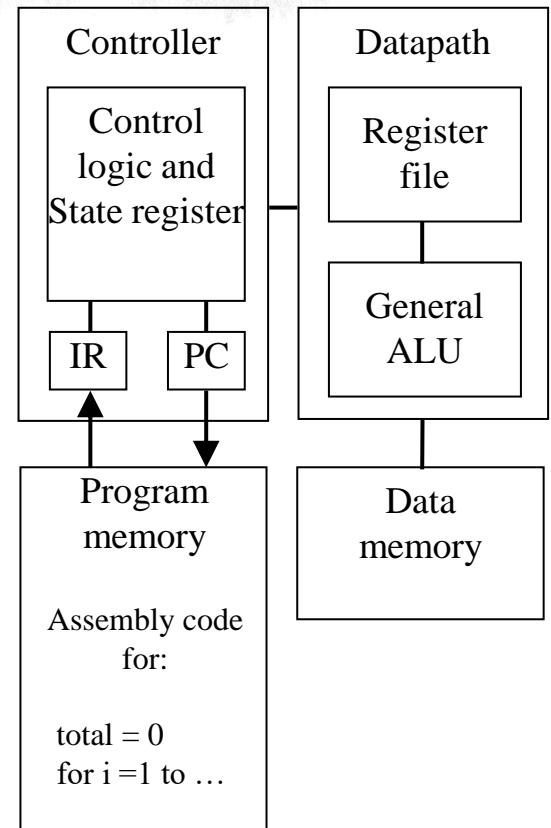
80 million PCs every year

3 billion embedded CPUs  
every year

Embedded systems  
market growing, while PC  
market mostly saturated

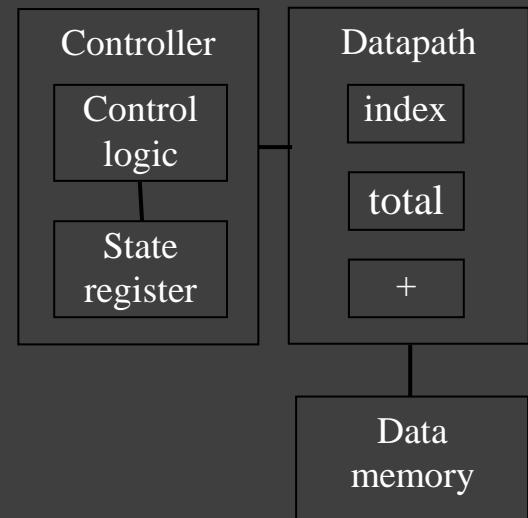
# General-Purpose Processor

- Programmable device, “microprocessor”
- Features
  - Program memory
  - General data path with large register file and general ALU
- User benefits
  - Low time-to-market and NRE costs
  - High flexibility
- Examples: Intel Core i7, AMD Ryzen 5, etc.



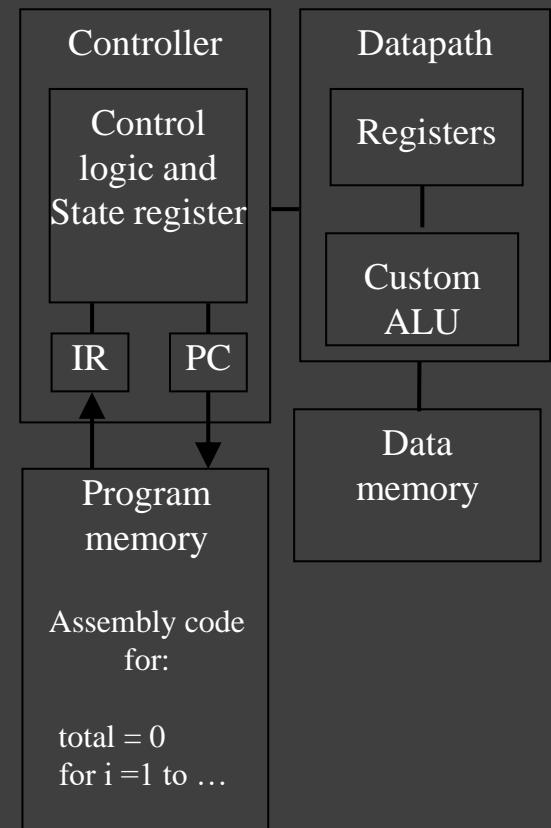
# Dedicated Processor

- Digital circuit designed specifically for one purpose
- Features
  - Contains only the components needed to execute a single program
  - No program memory
- Benefits
  - Fast
  - Low power
  - Small size



# Application-Specific Processor (ASIC)

- Programmable processor optimized for a particular class of applications that have common characteristics.
- Features
  - Program memory
  - Optimized data path
  - Special functional units
- Benefits
  - Some flexibility, good performance, size, and power, “reusable”

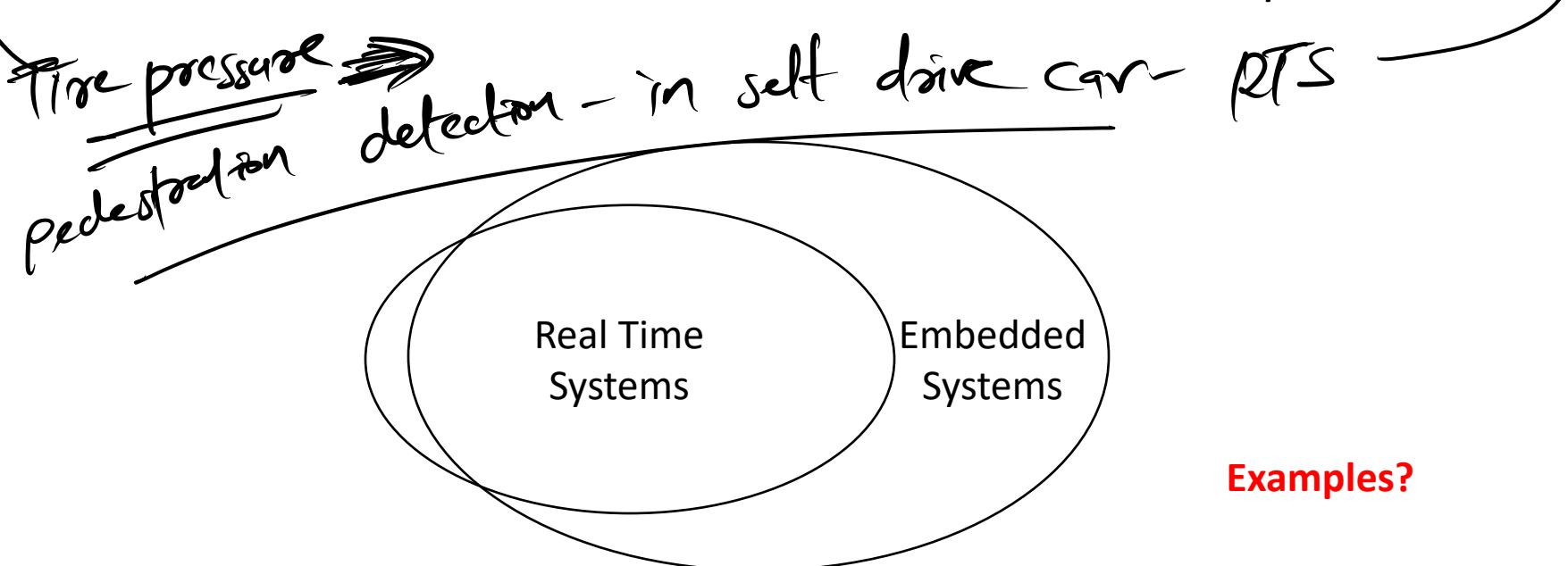


# Characteristics of Embedded Systems

- Dedicated functionality ↗
- Real-time operation ↗
- Small size and low weight ↗
- Low power ↗
- Harsh environments ↗
- Safety-critical operation ↗
- Cost sensitive ↗

# Embedded vs. Real Time Systems

- Embedded system: is a computer system that performs a limited set of specific functions; it often interacts with its environment
- RTS: Correctness of the system depends not only on the logical results, but also on the **time** in which the results are produced



## Examples

- Real Time Embedded:
    - Nuclear reactor control ✓
    - Flight control ✓
    - Basically, any safety critical system ✓
    - GPS ✓
    - MP3 player ✓
    - Mobile phone ✓
  - Real Time, but not Embedded:
    - Stock trading system
    - Skype
    - Pandora, Netflix
  - Embedded, but not Real Time:
    - Home temperature control ↗
    - Sprinkler system
    - Washing machine, refrigerator, etc.
- Software

# Benefits of embedded systems



## Greater performance and efficiency

More sophisticated control through software



## Lower cost

Cheaper components  
Reduced manufacturing costs  
Reduced operating and maintenance costs



## More features

Many not possible or impractical using other approaches

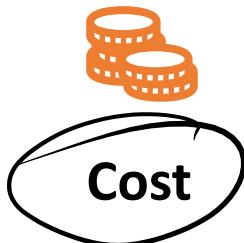


## Better dependability

Adaptive systems that can compensate for failures

Better diagnostics to improve repair time

# Constraints specific to embedded devices



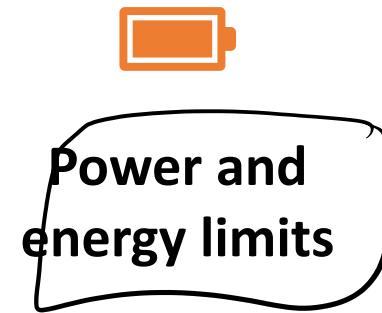
**Cost**

Competitive markets penalize products that do not deliver adequate value for money



**Size and weight limits**

Mobile (aviation, automotive) and portable (e.g., handheld, wearable) systems



**Power and energy limits**

Battery capacity, cooling limits



**Environment**

Temperatures may range from -40 degrees C to 125 degrees C, or even more.

# Functions of embedded systems



## Closed-loop control system

Monitor a process, adjust an output to maintain the desired set point of operation (temperature, speed, direction, etc.)



## Sequencing

Step through different stages based on environment and system conditions



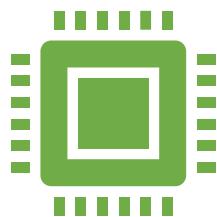
## Signal processing

Remove noise, select desired signal features



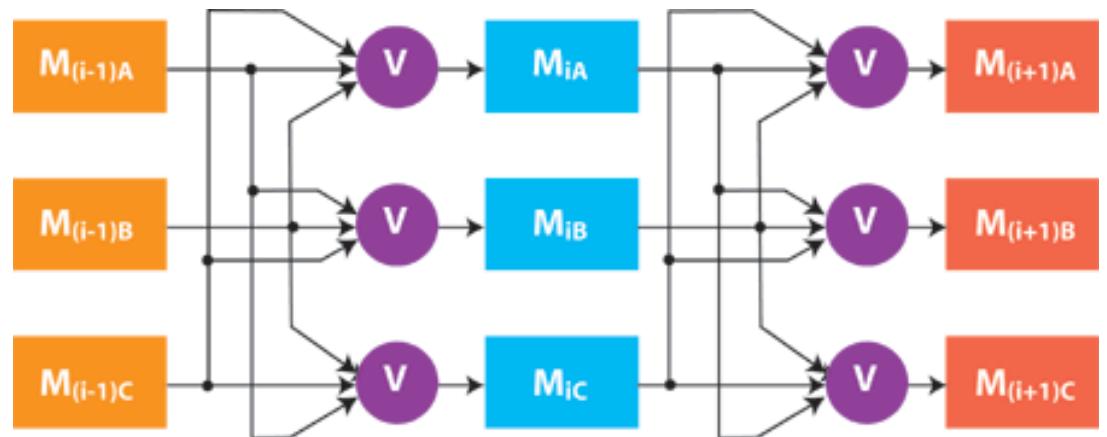
## Communications and networking

Exchange information reliably and quickly

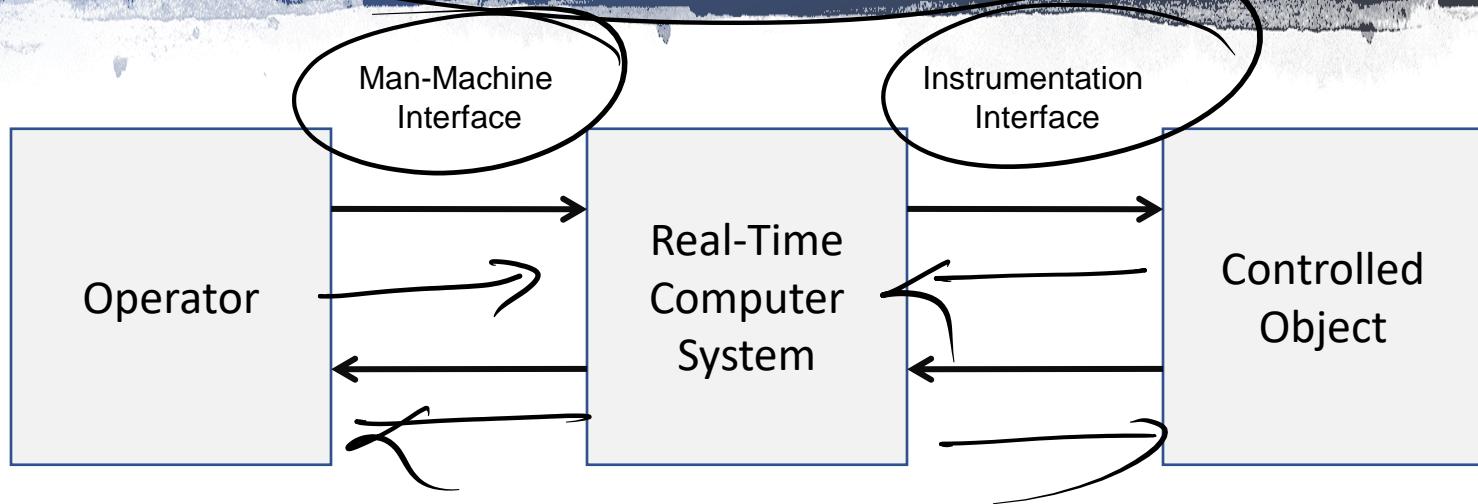


## Characteristics of RTS

- Event-driven (reactive) vs. time-driven
- Reliability/fault-tolerance requirements (example: triple modular redundancy)
- Predictability
- Priorities in multi-programmed systems



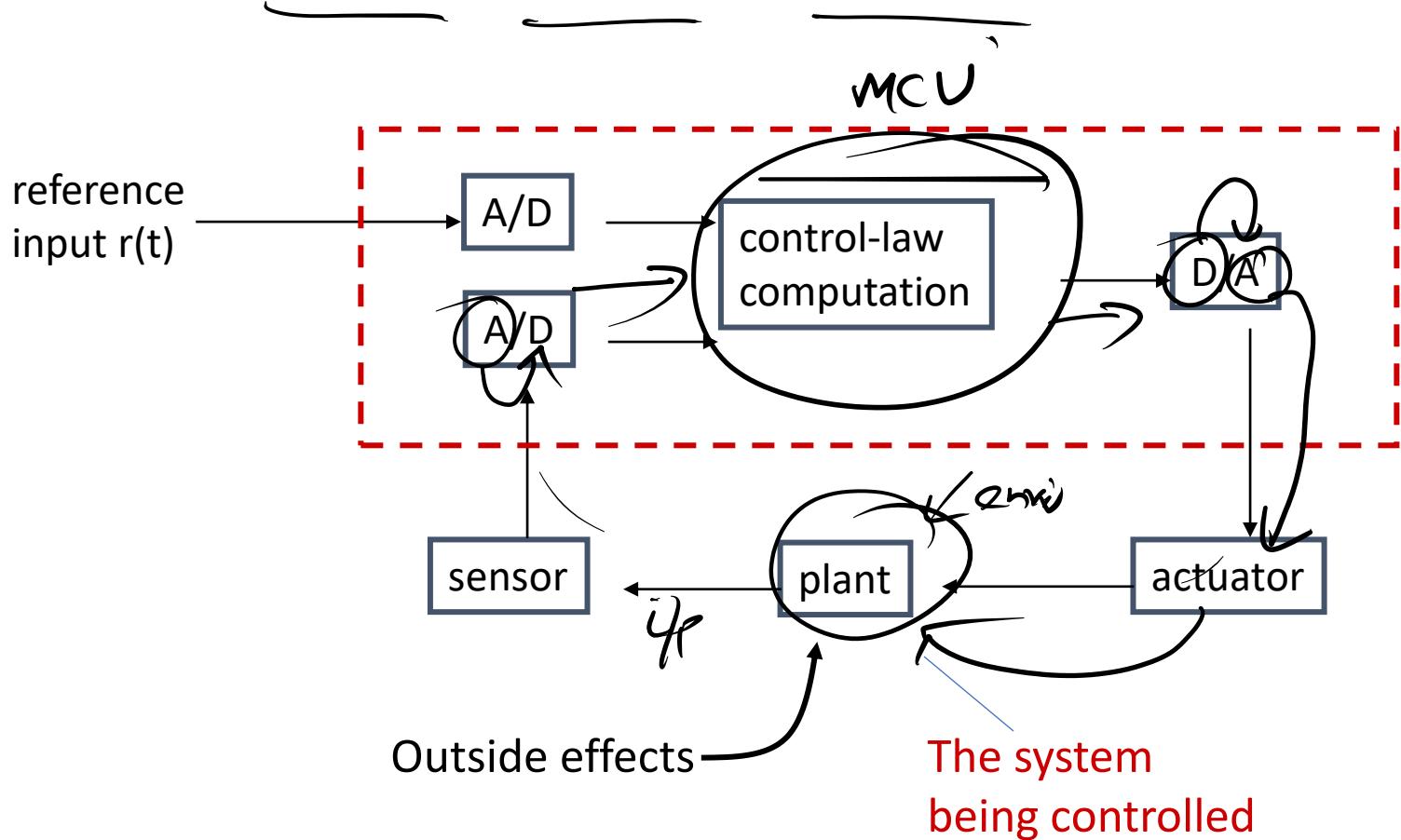
# Control Systems



- **Man-machine interface:** input devices, e.g., keyboard and output devices, e.g., display
- **Instrumentation interface:** sensors and actuators that transform physical signals and digital data.

# Control System Example

**Example:** A simple one-sensor, one-actuator control system.



# Control Systems Cont'd.

## Pseudo-code for this system:

```
set timer to interrupt periodically with period  $T$ ;  
at each timer interrupt do  
    do analog-to-digital conversion to get  $y$ ;  
    compute control output  $u$ ;  
    output  $u$  and do digital-to-analog conversion;  
end do
```

process it  
decide it

$T$  is called the sampling period.  $T$  is a key design choice. Typical range for  $T$ : seconds to milliseconds.

## Sensors and Actuators

### Sensors:

- They are mainly input components
- They sense and collect surrounding information

### Actuators:

- They are mainly output components
- They alter the surrounding

# Communications

- Connects devices with each other & the cloud
- Communication type:
  - Wireline (e.g., copper wires, optical fibers)
  - Wireless (e.g., RF, IR); RF-based communication is the most popular choice
- Popular RF-based communication solutions:
  - IEEE 802.15.4 (LR-WPANs) (Zigbee, 6LoWPAN) LoRaWAN
  - IEEE 802.11 (or Wifi)
  - Bluetooth (BLE)
  - Near Field Communication (NFC), e.g., RFID



Thank You!!

Next Session:  
**Human  
Computer  
Interaction**

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn:

<https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# Human Computer Interaction

Internet-of-Things (IoT)

---

COCOS20

# Smart Objects



Smart Refrigerator

- Objects that are able to **sense** the environment, **interpret** the environment, **self-configure**, **interact** with other objects and exchange information with people

# Traditional Computing System: HCI



***"Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them."*** -- Association for Computing Machinery.



(Bad) Examples of User Interfaces

TOO COOL TO DO DRUGS

COOL TO DO DRUGS

DO DRUGS

DRUGS



(Bad) Examples of User Interfaces



(Bad) Examples of User Interfaces



resume aborted transfer?

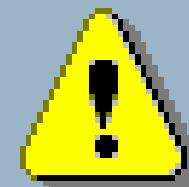
A file with this name already exists.  
Do you want to resume an aborted transfer?  
(Click No to overwrite your existing file  
and start a new transfer.)

Yes

No

Cancel

Microsoft Access



**Wrong button!**

This button doesn't work

**Solution**

Try another.

OK

(Bad) Examples of User Interfaces

# Why is HCI Important?

- It can affect
  - Effectiveness
  - Productivity
  - Morale
  - Safety
- Bad interfaces:
  - Confusing
  - Cumbersome
  - Time-consuming
  - Uninformative
  - Lead to errors
  - ...

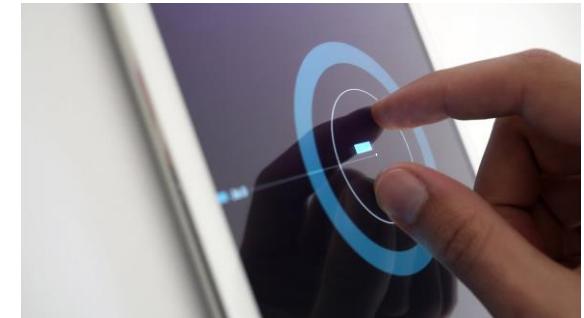
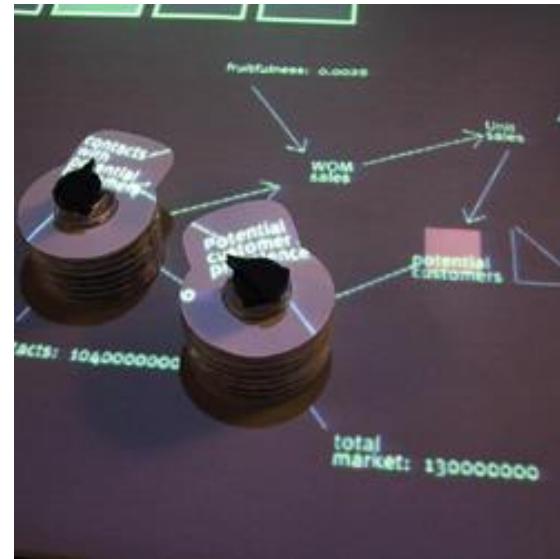


# Interfaces

- **Keyboard/mouse/screen/speakers**
- Pen input
- Touch
- Speech/audio/sound
- Gesture, eye movement
- Tangible interfaces
- Virtual/augmented reality (VR, AR)
- Wearable computing
- **Multi-modal** interactive interfaces: more than just one input/output channel

## Interface Discussion

- Ease-of-Use?
- Flexibility?
- Accuracy?
- Safety?
- Privacy?



# Touch as Input



Gesture/Motion as Input

**1**

An eye tracker consists of cameras, projectors and algorithms.

**2**

The projectors create a pattern of near-infrared light on the eyes.

**3**

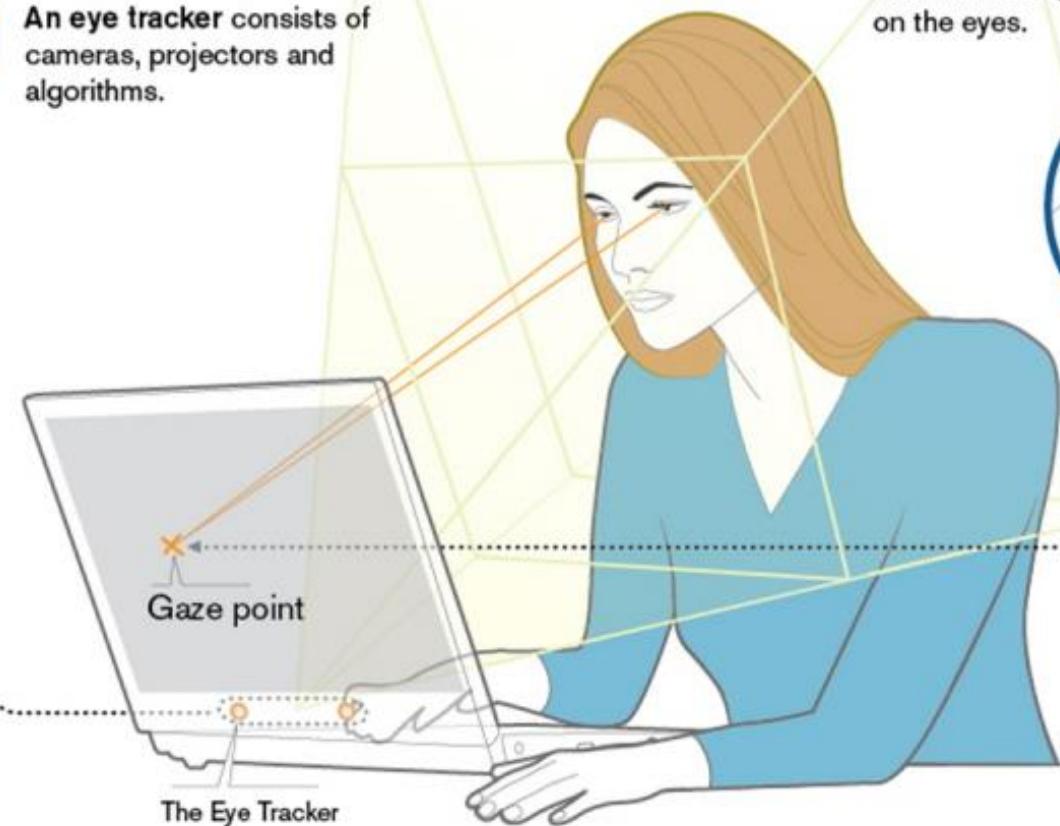
The cameras take high-frame-rate images of the user's eyes and the patterns.

**4**

The image processing algorithms find specific details in the user's eyes and reflections patterns.

**5**

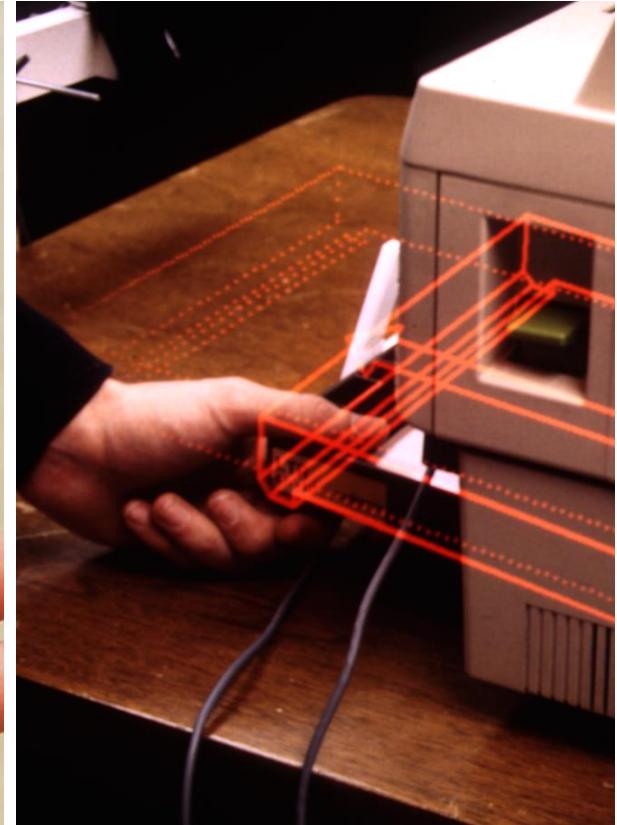
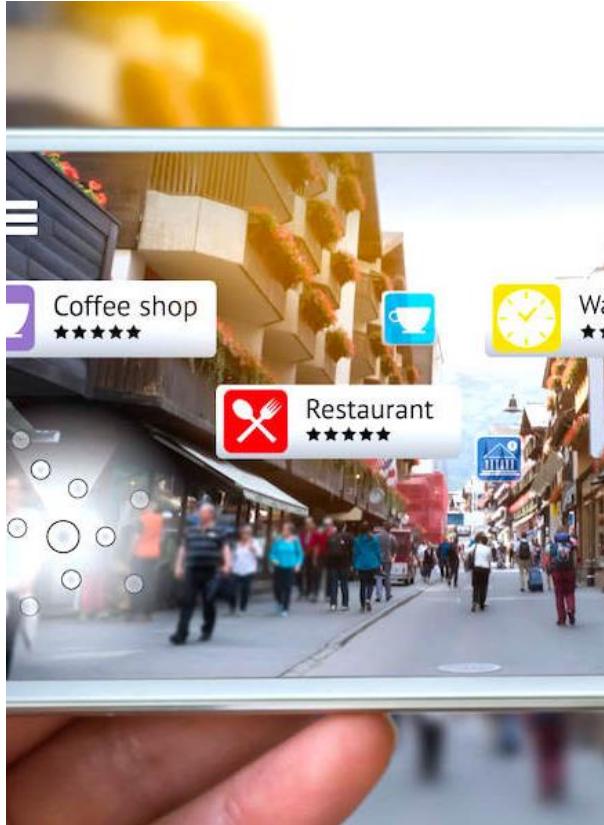
Based on these details, mathematical algorithms calculate the eyes' position and gaze point, for instance on a computer monitor.



## Eye Movement as Input

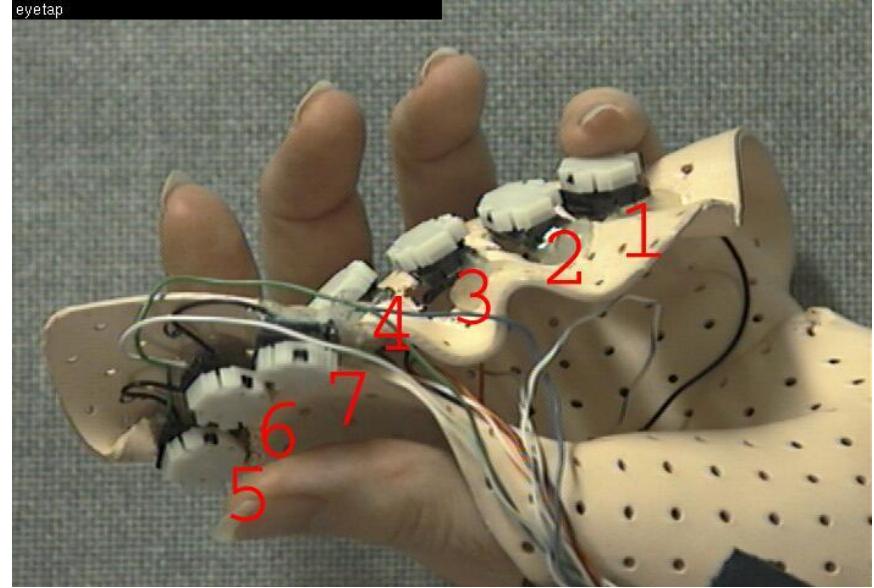
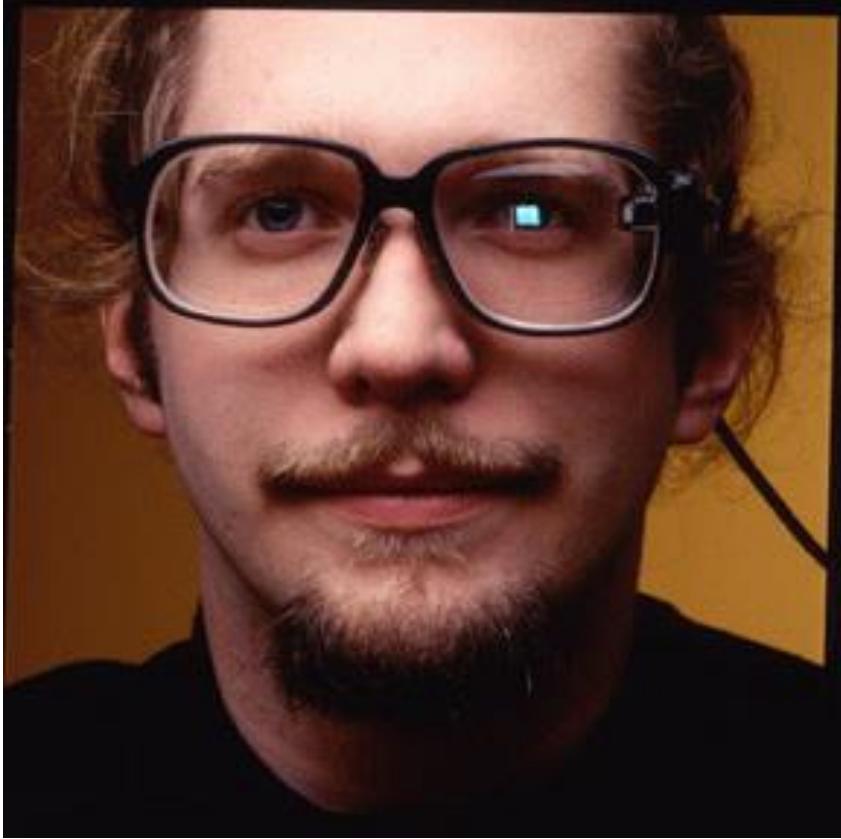
# Haptic Interfaces





# Augmented Reality

---



# Wearable Computing

---

Computation devices accompany you, rather than you seeking them out



"Hey Siri, what's the best  
sushi place in town?"



## Speech Input

- Human beings have a great and natural mastery of speech
  - makes it difficult to appreciate the complexities
  - but it's an easy medium for communication

# Windows Speech Recognition

- Supplied with every Windows machine
  - From '98 on
  - Almost no one used it
- What was the problem?
  - Need to “train” users to use early virtual assistants (VAs)
  - Microphone expense determines quality
  - No app buy-in.

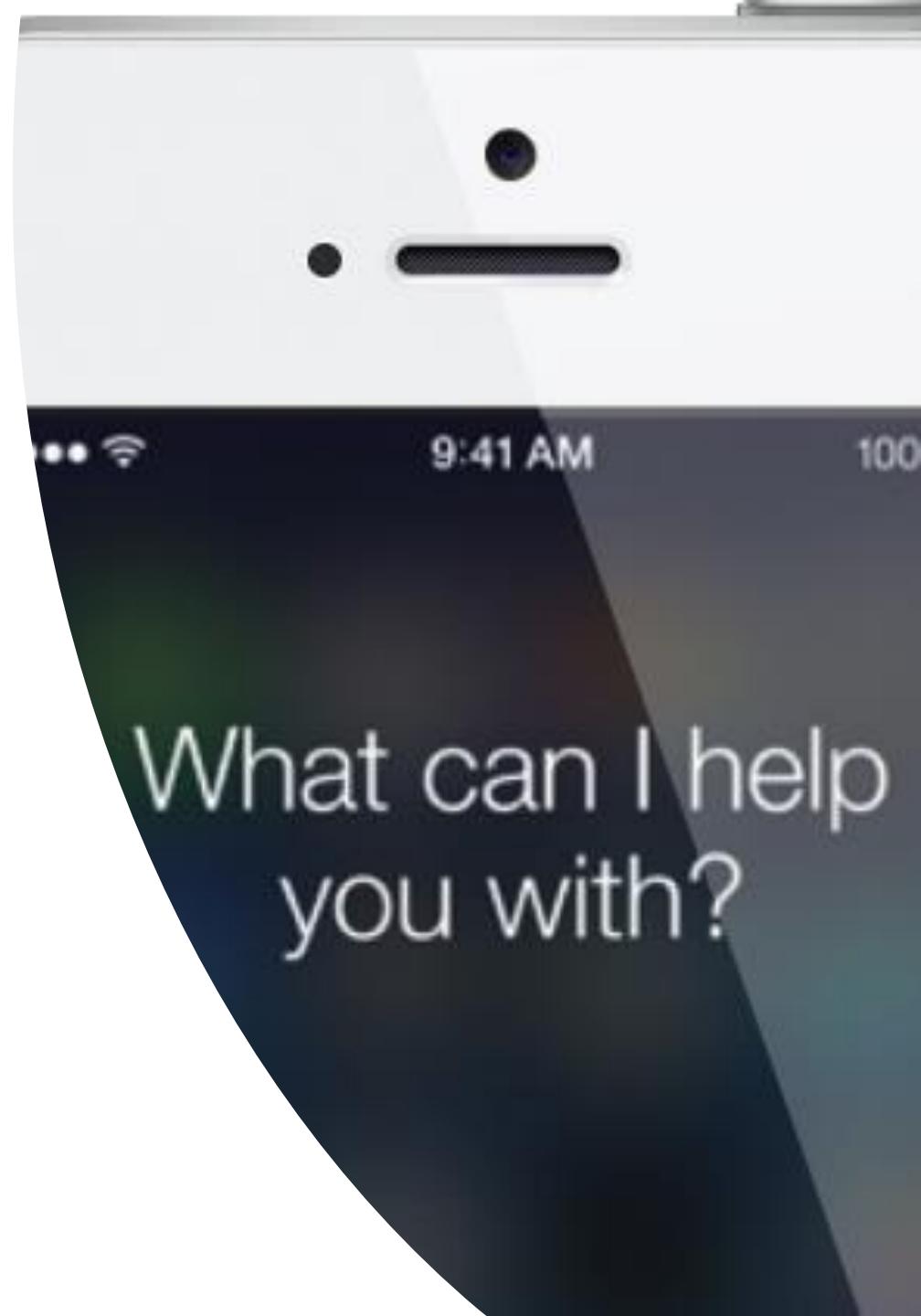


# And Then There Was Siri

---

## A Technical Success

- Consistent microphone gives predictable quality
- Inclusion on every iPhone made it mainstream



# And Then There Was Siri

- Misunderstandings
- Limited skills
- What Apple wants isn't always what users want
- No 3<sup>rd</sup> parties; limited innovation and evolution

“ I need to hide a body ”

What kind of place are you looking for?

reservoirs

metal foundries

mines

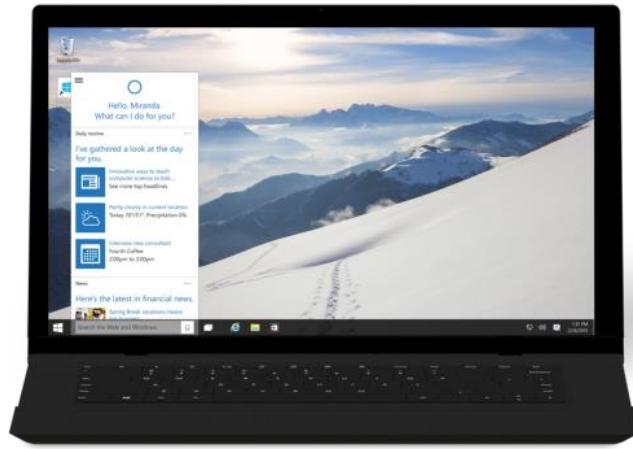
dumps

swamps



# Current Incarnations

- What these look like now
  - Specialized hardware
  - Domestic setting
  - Initially aimed at home automation
  - Mostly used for home entertainment
  - All open to 3<sup>rd</sup> parties



# Voice “Explodes” into Mainstream



IBM Watson™



# Seven Design Principles

## 1. **Equitable use**

- same means for all users, do not segregate/stigmatize users, make design appealing

## 2. **Flexibility in use**

- provide choice of methods & adapt to user's pace

## 3. **Simplicity and intuitiveness of use**

- support user's expectations
- accommodate different languages and literacy skills
- provide prompting and feedback

# Seven Design Principles

## 4. Perceptible information

- redundancy of information: use different forms/modes
- emphasize essential information

## 5. Tolerance for error

- minimize impact caused by mistakes
- remove potentially dangerous situations
- hazards should be shielded by warnings

# Seven Design Principles

## **6. Low physical effort**

- comfort; minimize fatigue and effort
- repetitive or sustained actions should be avoided

## **7. Size and space for approach and use**

- placement of system should be reachable by all users
- consider line of sight for standing and sitting user
- allow for variation in hand size
- provide room for assistive devices

# Disabilities

---

- Federal law to ensure access to IT, including computers and web sites.
  - Vision (low vision, blind, color blind)
  - Hearing (deaf, limited hearing)
  - Mobility
  - Learning (dyslexia, attention deficit)

# Disabilities

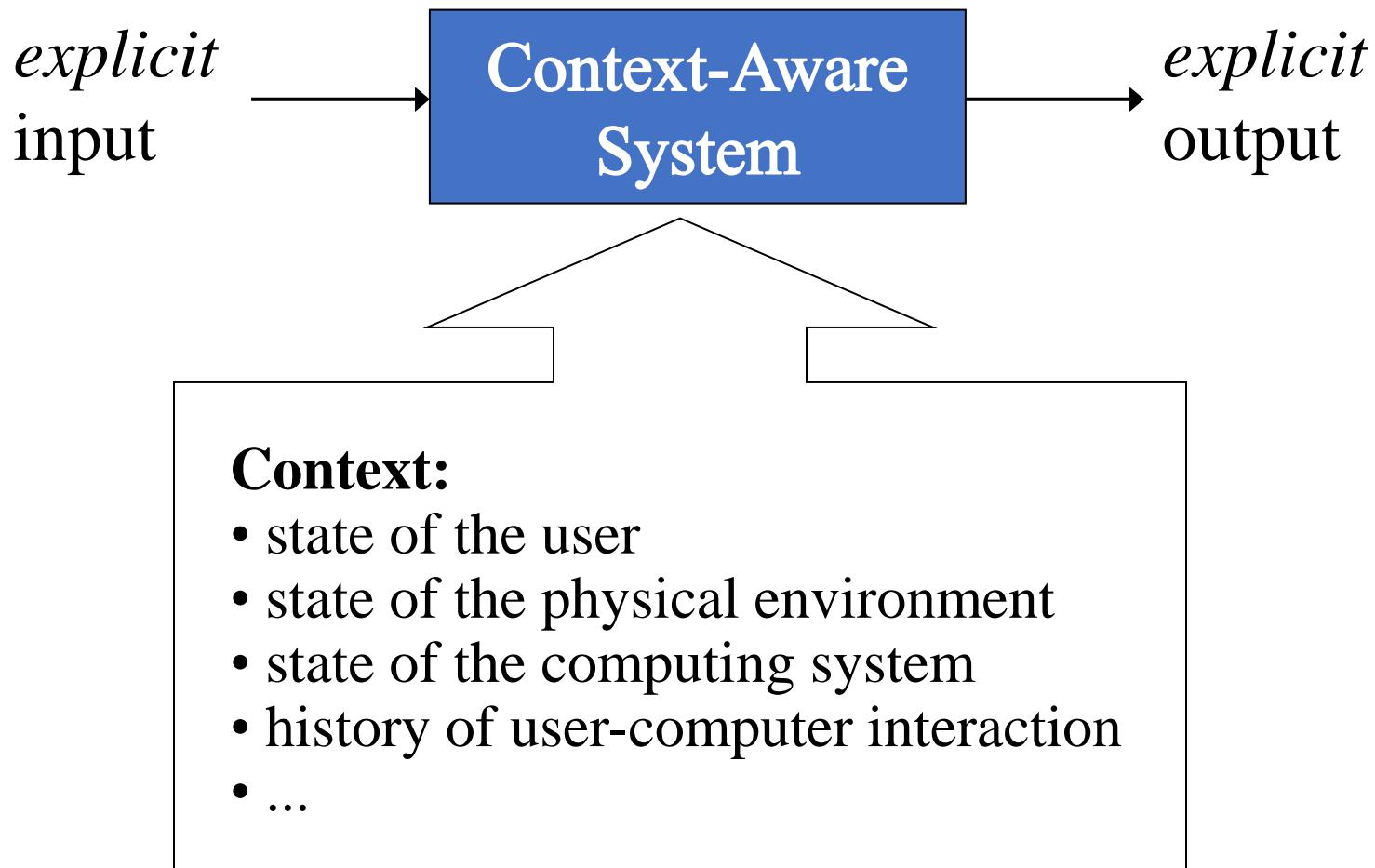
---

- Keyboard and mouse alternatives
- Color coding
- Font size
- Contrast
- Text descriptors for web images
- Magnification
- Text-to-speech; speech recognition
- Head-mounted optical mice
- Eye gaze control

# System Structure



# Context as Implicit Input



# What is Context?



# Examples of Context

- Identity (user, others, objects)
- Location
- Date/Time
- Environment
- Emotional state
- Focus of attention
- Orientation
- User preferences
- Calendar (events)
- Browsing history
- Behavioral patterns
- Relationships (phonebook, call history)
- ... the elements of the user's environment that the computer knows about...

# Relevance of Context Information

- Trying to arrange lunch meeting
- Going to a job interview
- Going home after work and making evening plans
- Shopping
- Tourist
- ...

# Definitions of Context

- “Context is **any information that can be used to characterize the situation of an entity**. An entity is a person, place, or object that is considered **relevant** to the interaction between a user and an application, including the user and applications themselves” [Dey et al. 2001]

# Classification

## External (physical)

- Context that can be measured by hardware sensors
- Examples: location, light, sound, movement, touch, temperature, air pressure, etc.

## Internal (logical)

- Mostly specified by the user or captured monitoring the user's interaction
- Examples: the user's goal, tasks, work context, business processes, the user's emotional state, etc.

# Context?



# Context?



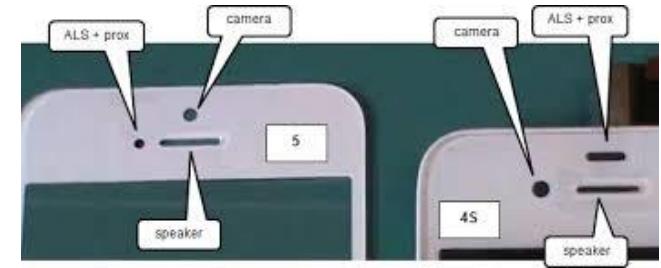
# Simple Everyday Examples

- Smartphone adjusts the screen to the orientation of the device
- Apple Watch turns on display if arm lifted/rotated
- Orientation is determined by using both a gyroscope and an accelerometer



# Simple Everyday Examples

- Phone display adjusts the brightness of the display based on the surrounding area
- Uses a light sensor



# Simple Everyday Examples

- Device displays user's location, shows route to a desired destination, find nearby stores, geotag images on social media, etc.
- Uses location sensor



# Simple Everyday Examples

- The time is displayed on the phone
  - Time zone change
  - Daylight savings time



# Simple Everyday Examples

- Device disables touch screen when the user speaks on the phone
- Uses a proximity sensor (infrared signal travel time)



# Challenges

- Lack of self-awareness
  - Knowing when to do or not to do something is hard
- Complexity
  - More rules do not necessarily yield more intelligence
  - But will become harder to maintain and understand
- Human-in-the-loop vs. automation
  - Loss of control vs. risk of human error
- Development
  - Sensing, aggregation, rules, etc., are complex issues
- Privacy
- User preferences
- Information overload

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn:

<https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# Sensors

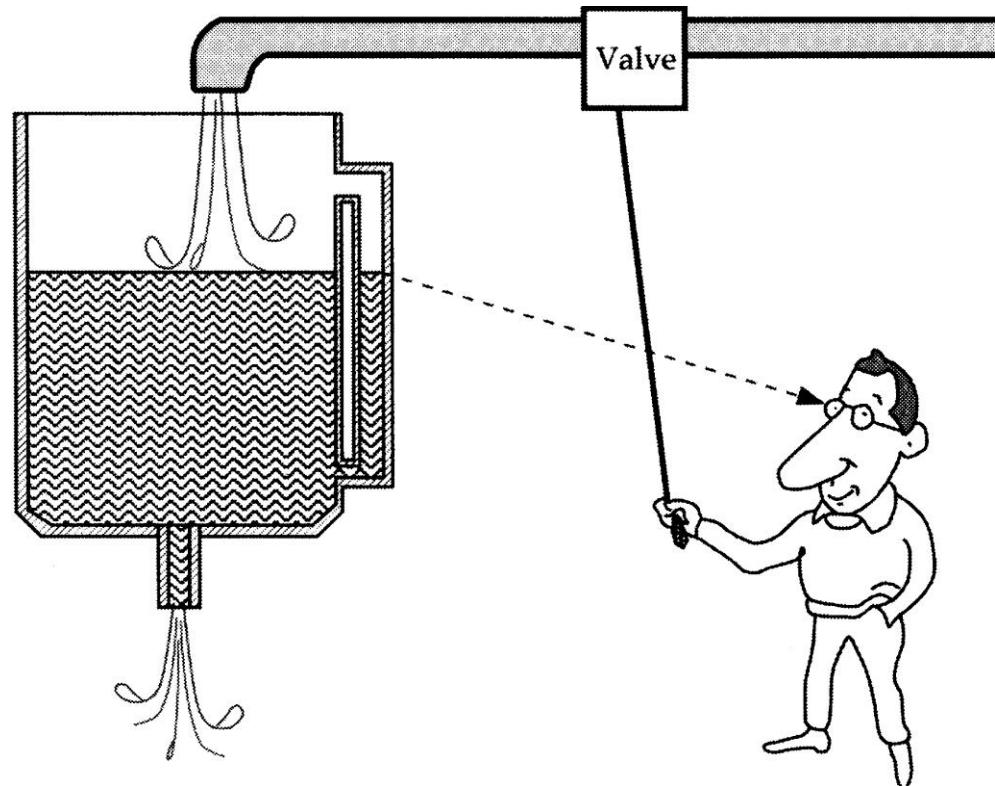
Internet-of-Things (IoT)

---

COCOS20

# Sensors

- A sensor is a device that receives a stimulus and responds with an electrical signal.



# Sensors

Sensors are devices that measure a particular property of the environment and convert that into an electrical output

Typically, sensors have a linear function between the physical property measured and the electrical output

The sensitivity of a sensor indicates the minimum value of the measured input that can produce a certain output signal

An analog-to-digital converter employed to turn a sensor output into signals that can be processed by MCUs

Different physical phenomena underpin the operation of sensors, e.g., the piezoelectric effect (accumulation of electric charge when mechanical pressure is applied)



## Computer-Process Interface

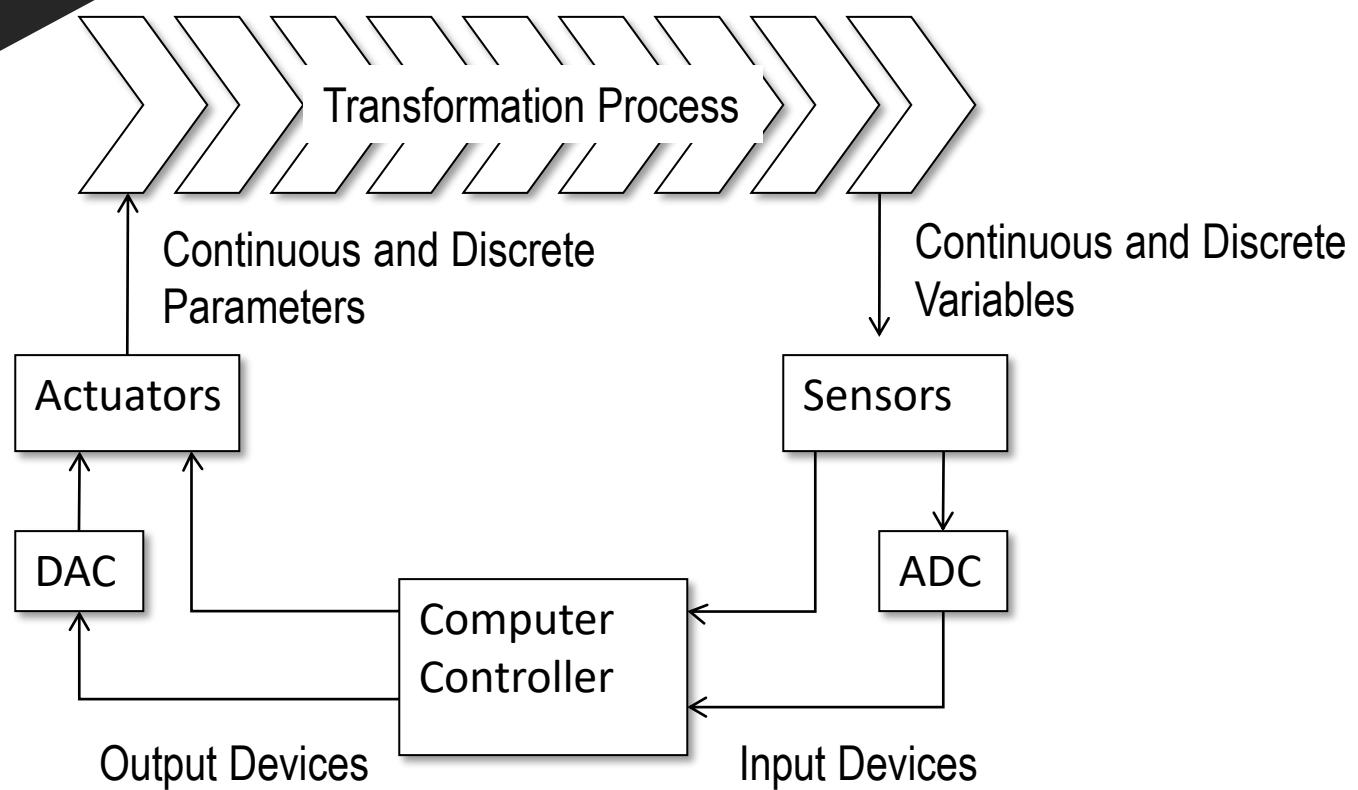
---

- To implement process control, the computer must collect data from and transmit signals to the production process
- Components required to implement the interface:
  - Sensors to measure continuous and discrete process variables
  - Actuators to drive continuous and discrete process parameters
  - Devices for ADC and DAC
  - I/O devices for discrete data

# Need for Sensors

- Sensors are omnipresent. They embedded in our bodies, automobiles, airplanes, cellular telephones, radios, chemical plants, industrial plants and countless other applications.
- Without the use of sensors, there would be no automation!!

# Computer Process Control System



# What is a Stimulus?

- Motion, position, displacement
- Velocity and acceleration
- Force, strain
- Pressure
- Flow
- Sound
- Moisture
- Light
- Radiation
- Temperature
- Chemical presence



Visual Sensor



Ultrasound Sensor



Infrared Sensor

# Types of sensors

- Push-button/switch



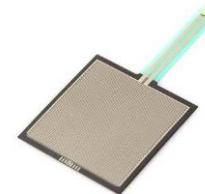
- Temperature



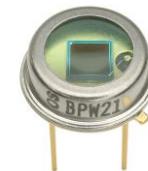
- Acceleration



- Pressure



- Optical/photodiode



- Humidity



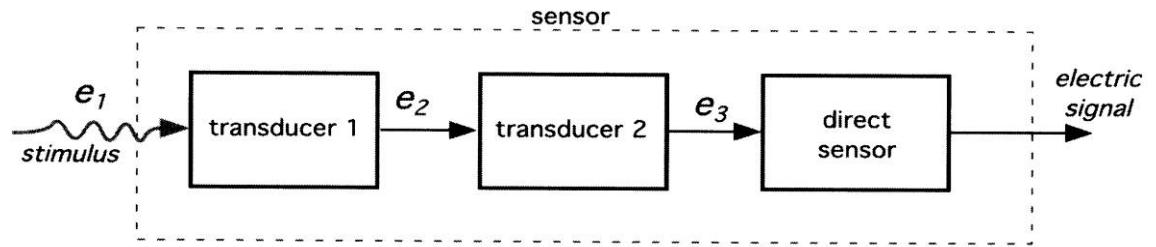
# What is a Response?

When we say electrical, we mean a signal which can be channeled, amplified, and modified by electronic devices:

- **Voltage**
- **Current**
- **Charge**

# Sensor as Energy Converter

- This conversion can be direct or it may require transducers



- Example:

A chemical sensor may have a part which converts the energy of a chemical reaction into heat (transducer) and another part, a thermopile, which converts heat into an electrical signal.

Microphone, Loud Speaker, Biological Senses (e.g. touch, sight,..., etc)

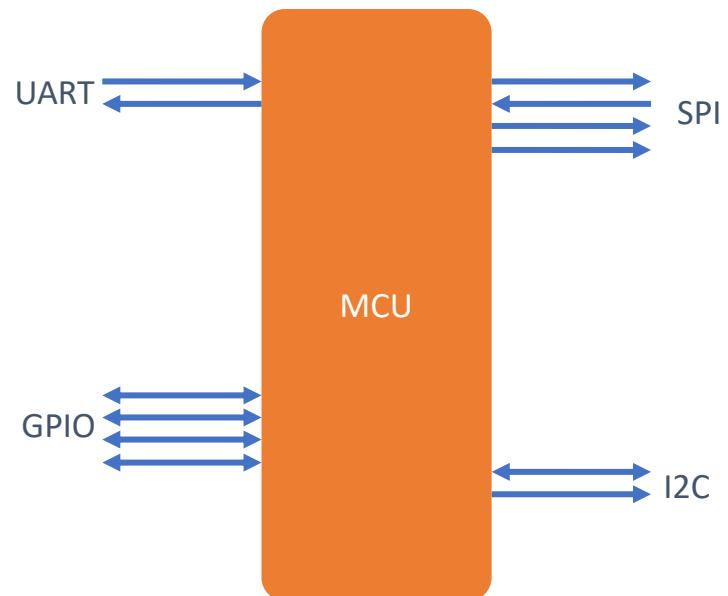
# Physical Principles of Sensing

- Charges, fields, & potentials
- Capacitance
- Magnetism
- Induction
- Resistance
- Piezoelectric effect
- Seebeck and Peltier effects
- Thermal properties of materials
- Heat transfer
- Light

# Input/Output

I/O defines how an MCU can interact with the environment

- Different I/O protocols are available
- Universal Asynchronous Receiver/Transmitter (UART) – two wires to send/receive data between devices
- Serial Peripheral Interface (SPI) – any number of bits can be sent/received without interruption
- Inter-Integrated Circuit (I<sup>2</sup>C) – combines features of UART and SPI
- General purpose I/O (GPIO) – controllable by user at run time.



# GPIO



Can be set up to accept or source different logic voltage levels, through which MCU can control peripherals or receive external input/interrupts.



If pins are configured for interrupts, they can be used to move wake-up from low-power/sleep modes.



Can be grouped into a GPIO port and controlled as such.



Pulse-width modulation (PWM) employed when the linear processes must be controlled (e.g., fans)



Limited to low-current applications → transistors and relays used to help drive higher current loads.

# Pulse-width modulation

Reducing average power output by switching on/off at high rates.

- Duty Cycle – the fraction of one period  $T$  during which a signal is active:

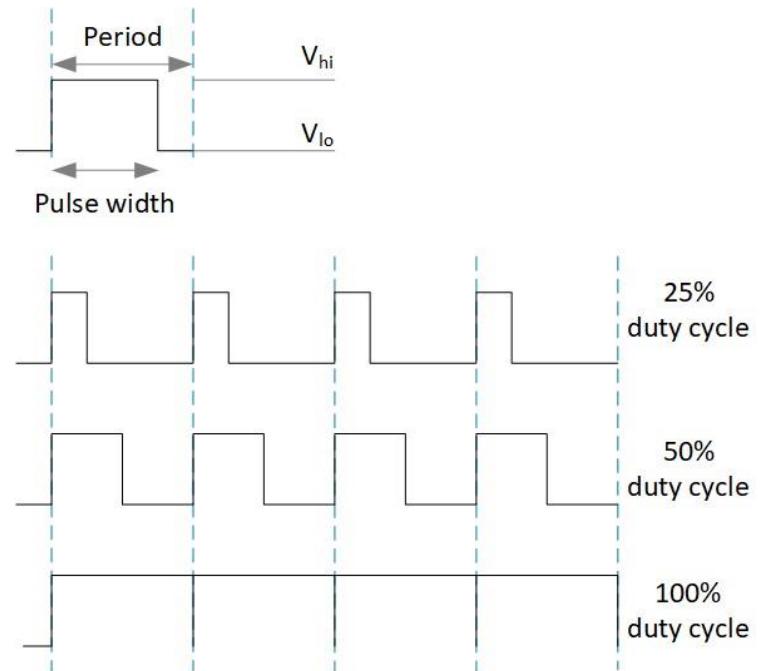
$$D = \frac{PW}{T} \times 100 [\%]$$

- Frequency – rate of periods:

$$f = \frac{1}{T} [\text{Hz}]$$

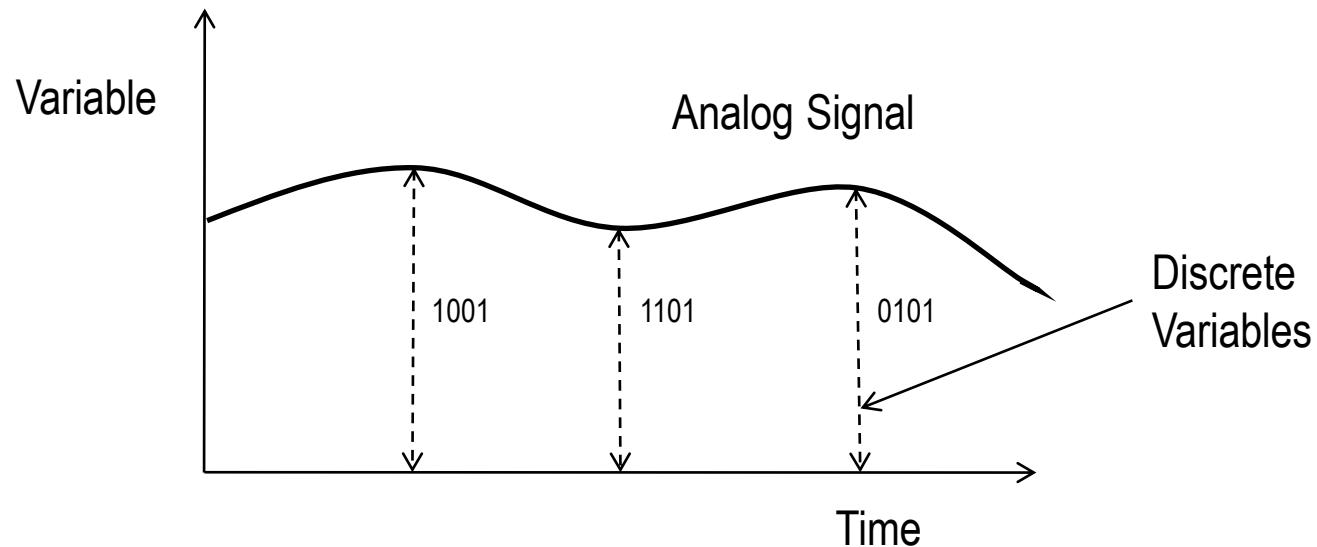
- Average voltage:

$$V_{avg} = V_{hi} \times \frac{D}{100}$$



# Analog-to-Digital Conversion

- **Sampling** – converts the continuous signal into a series of discrete analog signals at periodic intervals
- **Quantization** – each discrete analog is converted into one of a finite number of (previously defined) discrete amplitude levels
- **Encoding** – discrete amplitude levels are converted into digital code



# Features of an ADC

- **Sampling rate** – rate at which continuous analog signal is polled (e.g., 1000 samples/sec)
- **Quantization** – divide analog signal into discrete levels
- **Resolution** – depends on number of quantization levels
- **Conversion time** – how long it takes to convert the sampled signal to digital code
- **Conversion method** – means by which analog signal is encoded into digital equivalent
  - Example: Successive approximation method

# Successive Approximation Method

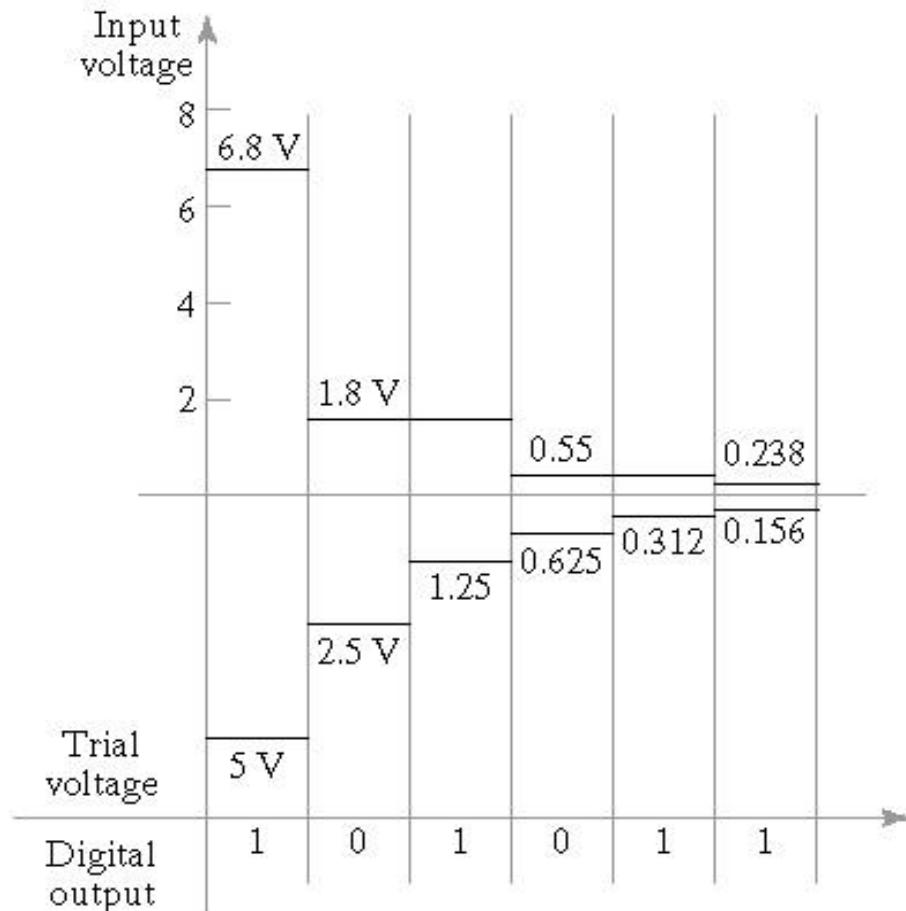
- A successive approximation ADC is a type of ADC that converts a continuous analog waveform into a discrete digital representation via a binary search through all possible quantization levels before finally converging upon a digital output for each conversion

# Algorithm

- A series of trial voltages are successively compared to the input signal whose value is unknown
- Number of trial voltages = number of bits used to encode the signal
- First trial voltage is  $1/2$  the full scale range of the ADC
- If the remainder of the input voltage exceeds the trial voltage, then a bit value of 1 is entered, if less than trial voltage then a bit value of zero is entered
- The successive bit values, multiplied by their respective trial voltages and added, becomes the encoded value of the input signal

# Example

- Analogue signal is 6.8 volts. Encode, using SAM, the signal for a 6 bit register with a full scale range of 10 volts.



For six digit precision,  
the resulting binary  
digital value is 101011,  
which is interrupted as:

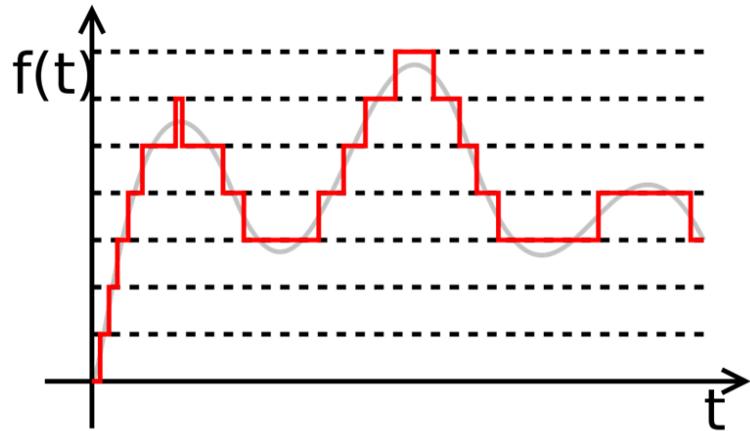
$$\begin{aligned} & 1 \times 5.0 \text{ V} \\ & 0 \times 2.5 \text{ V} \\ & 1 \times 1.25 \text{ V} \\ & 0 \times 0.625 \text{ V} \\ & 1 \times 0.312 \text{ V} \\ & 1 \times 0.156 \text{ V} \end{aligned}$$

$$\text{Total} = 6.718 \text{ V}$$

# Example

$$Q = \frac{E_{FSR}}{2^M} = \frac{E_{FSR}}{N}$$

- Q = resolution in volts per step
- M = resolution in bits
- N = Number of intervals (steps, quantization levels)
- $E_{FSR}$  = Full scale voltage range
- Quantization error =  $\frac{1}{2}$  of interval
  
- Voltage range 0 – 10V; M = 12 bits
- N = 4096 intervals (steps)
- Q = 2.44 mV/code



## Example

- Using an analogue-to-digital converter, a continuous voltage signal is to be converted into its digital counterpart. The ADC has a 16-bit capacity, and full scale range of 60 V. Determine:
  - number of quantization levels
  - resolution
  - quantization error

# Solution

(1) Number of quantization levels:

$$= 2^{16} = 65,536$$

(2) Resolution:

$$= 60 / 65,536 = \sim 0.00092 \text{ Volts}$$

(3) Quantization error:

$$= 0.00092/2 = 0.00046 \text{ Volts}$$

# Digital-to-Analog Conversion

- Convert digital values into continuous analogue signal
  - Decoding digital value to an analogue value at discrete moments in time based on value within register

$$E_0 = E_{ref} \left\{ 0.5B_1 + 0.25B_2 + \dots + (2^n)^{-1} B_n \right\}$$

Where  $E_0$  is output voltage;  $E_{ref}$  is reference voltage;  $B_n$  is status of successive bits in the binary register

# Example

- A DAC has a reference voltage of 100 V and has 6-bit precision. Three successive sampling instances 0.5 sec apart have the following data in the data register:

Instant	Binary Data
1	101000
2	101010
3	101101

- Output Values:

$$E_{01} = 100\{0.5(1)+0.25(0)+0.125(1)+0.0625(0)+0.03125(0)+0.015625(0)\}$$

$$E_{01} = 62.50V$$

$$E_{02} = 100\{0.5(1)+0.25(0)+0.125(1)+0.0625(0)+0.03125(0)+0.015625(0)\}$$

$$E_{02} = 65.63V$$

$$E_{03} = 100\{0.5(1)+0.25(0)+0.125(1)+0.0625(0)+0.03125(0)+0.015625(0)\}$$

$$E_{03} = 70.31V$$

# Sensor Types: HW & SW

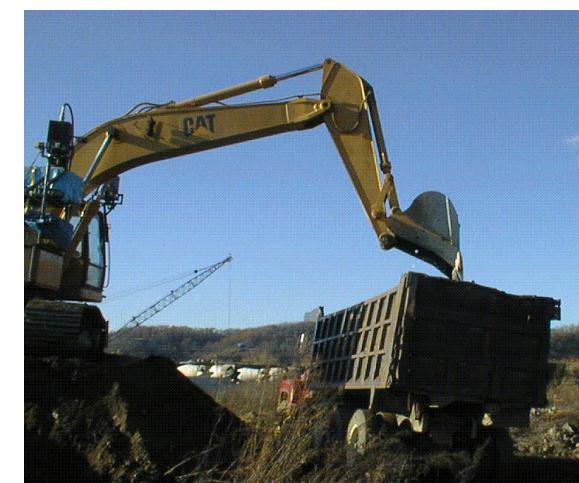
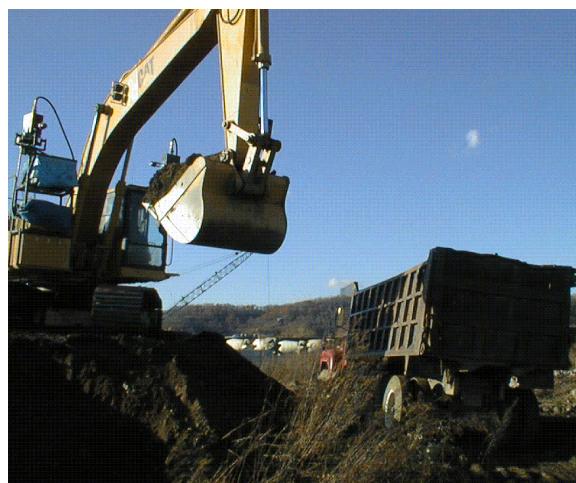
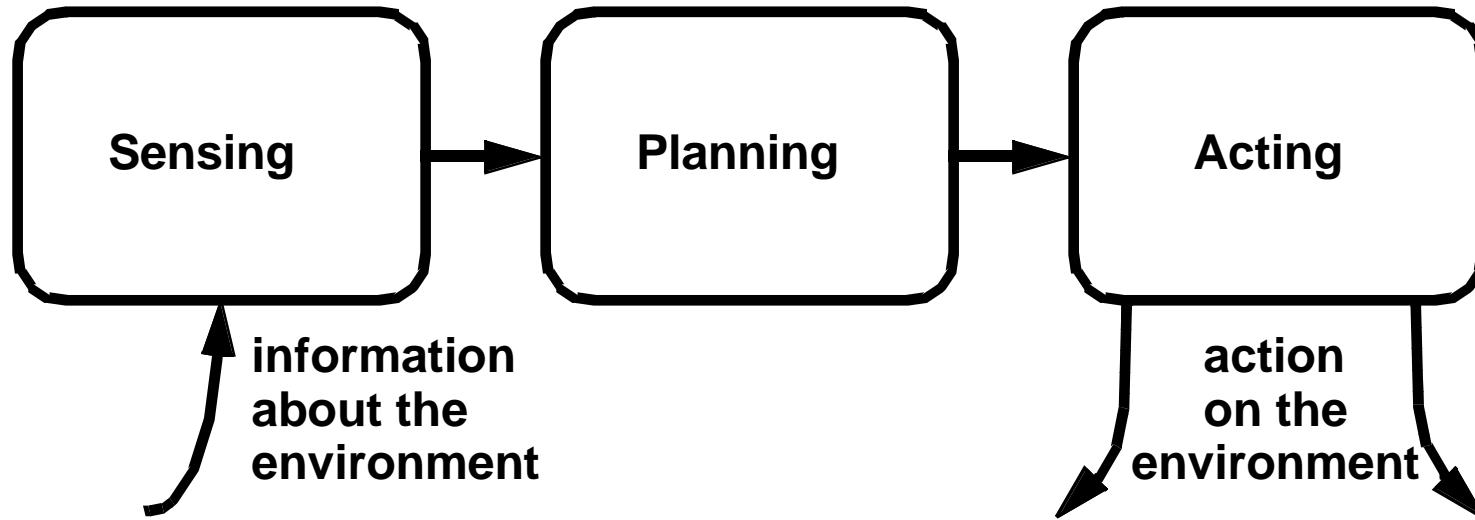
- **Hardware-based sensors**
  - Physical components built into a device
  - They derive their data by directly measuring specific environmental properties
- **Software-based sensors**
  - Not physical devices, although they mimic hardware-based sensors
  - They derive their data from one or more hardware-based sensors

# Sensor List of Smartphone

Sensor	Function Type	Software-based or Hardware-based
Accelerometer	Motion Sensor	Hardware-based
Gyroscope	Motion Sensor	Hardware-based
Gravity	Motion Sensor	Software-based
Rotation Vector	Motion Sensor	Software-based
Magnetic Field	Position Sensor	Hardware-based
Proximity	Position Sensor	Hardware-based
GPS	Position Sensor	Hardware-based
Orientation	Position Sensor	Software-based
Light	Environmental Sensor	Hardware-based
Thermometer	Environmental Sensor	Hardware-based
Barometer	Environmental Sensor	Hardware-based
Humidity	Environmental Sensor	Hardware-based

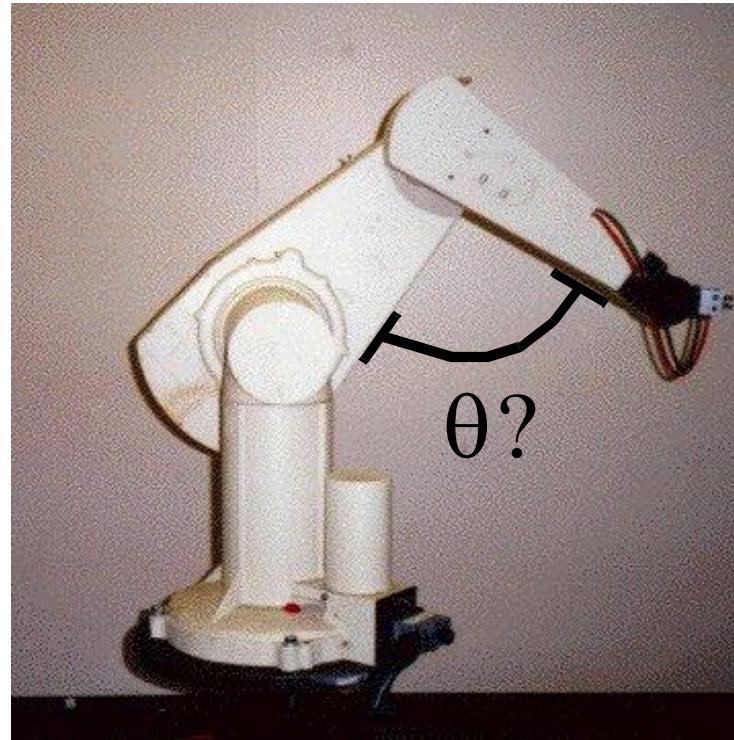
# **Overview of Our Sensors For Robotics**

# What makes a machine a robot?



Why do robots need sensors?

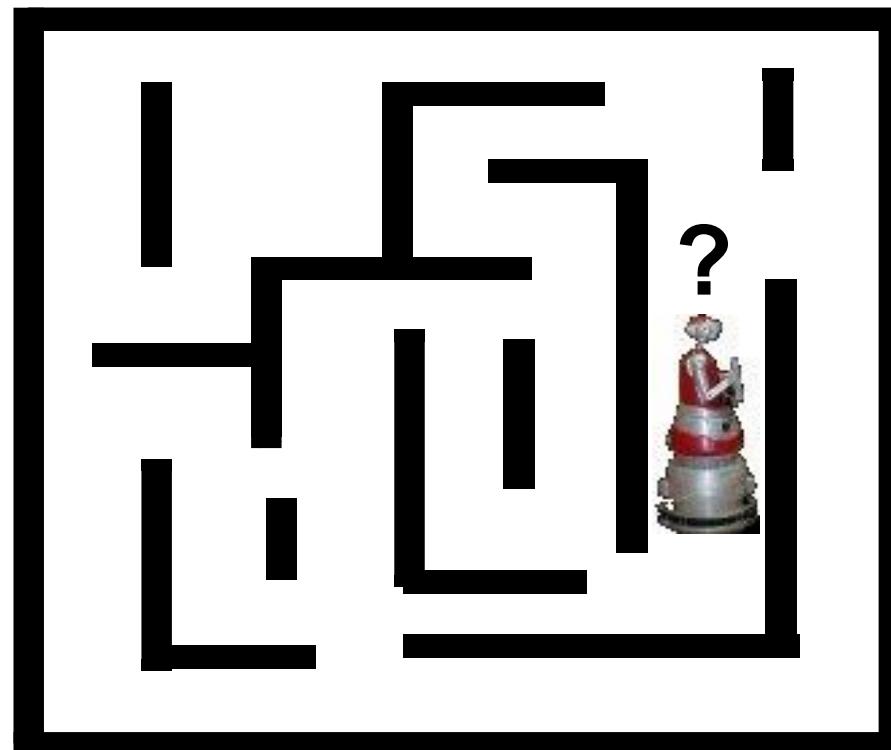
**What is the angle of my arm?**



**internal information**

# Why do robots need sensors?

**Where am I?**



**localization**

# Why do robots need sensors?

**Will I hit anything?**



**obstacle detection**

Sensing for specific tasks

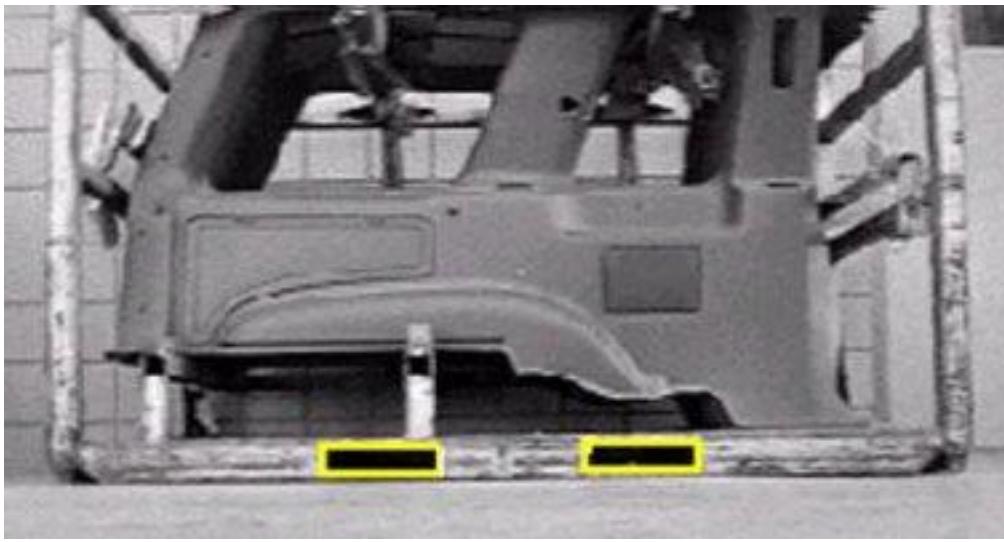
**Where is the cropline?**



**Autonomous  
harvesting**

Sensing for specific tasks

**Where are the forkholes?**



**Autonomous material handling**

Sensing for specific tasks

**Where is the face?**



**Face detection & tracking**

# Types of Sensors

- Active

- send signal into environment and measure interaction of signal w/ environment
- e.g. radar, sonar

- Passive

- record signals already present in environment
- e.g. video cameras

# Actuators

- Hardware devices that convert a controller command signal into a change in a physical parameter
  - The change is usually mechanical (e.g., position or velocity)
  - An actuator is also a transducer because it changes one type of physical quantity into some alternative form
  - An actuator is usually activated by a low-level command signal, so an amplifier may be required to provide sufficient power to drive the actuator

# Types of Actuators



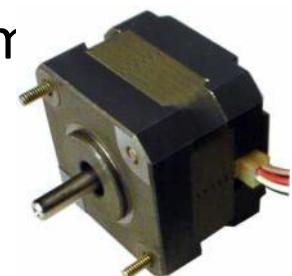
## 1. Electrical actuators

- Electric motors
  - DC servomotors
  - AC motors
  - Stepper motors
- Solenoids



## 2. Hydraulic actuators

- Use hydraulic fluid to amplify the controller command signal



## 3. Pneumatic actuators

- Use compressed air as the driving force



Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn:

<https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# IoT System Architectures and Standards

COCSC20

# Syllabus

This module will cover the following aspects

- Key considerations for IoT architectures
- Cloud, fog, and edge paradigms
- The role of gateways in IoT
- IoT internetworking approaches
- Standards that enable practical IoT deployment and interoperability

# The IoT architectural landscape



- Thousands of new applications exist, spanning countless domains (verticals).
- Each application has its unique requirements → combining these leads to systems that are complex, difficult to manage, and often proprietary.
- Defining a unified architecture is challenging and interoperability problematic, if there are too many standards to choose from.
- Efforts by multiple entities to define common frameworks, including international standardization bodies, multi-national collaborative research projects, industry consortiums, and large commercial actors.
- Device/protocol documentation is scattered and often difficult to navigate.
- We will focus on the key principles different architectural patterns share and examine some examples.

# Key considerations for IoT architectures



What application domains should be covered?



Where to place the “intelligence”?



What networking structure should be employed?



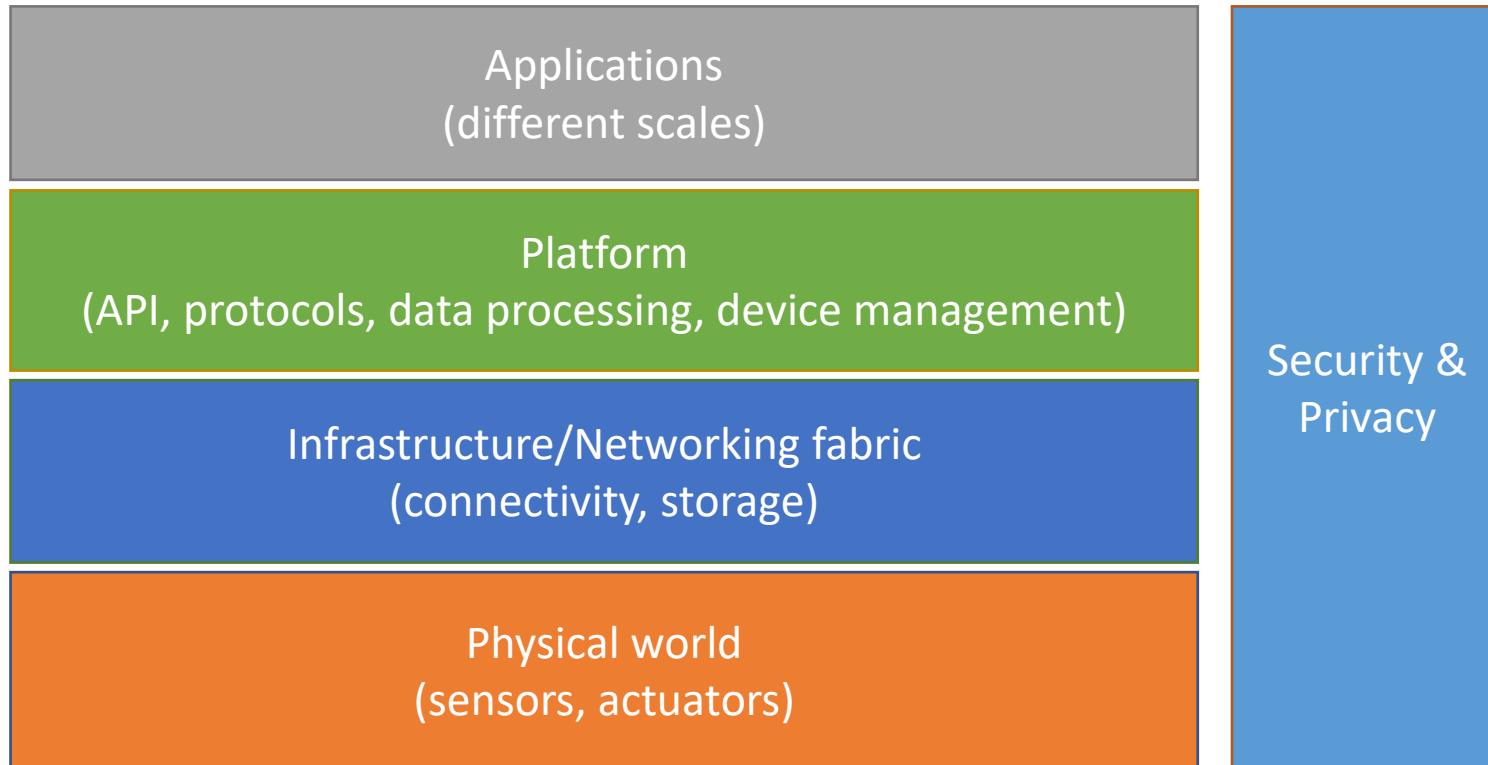
How to modularize systems, so as to manage complexity and enable programmability?



What are the cost and scalability implications?

# A layered view to IoT architecture

At a high level, stakeholders may converge to a shared vision



This approach enables to break up complexity, share resources more easily, and promote interoperability

# Advantages of the layer approach

- Allows IoT device manufacturers to focus strictly on improving their performance, power consumption, etc. – expose only well-defined interfaces to software platforms.
- Easier to share and partition strictly the network and computing resources (slicing); reducing the burden on service providers to build and manage networks – Infrastructure/Network as a Service (IaaS/NaaS)
- Enables software/app developers to build applications without having to understand the specifics of a device – Platform as a Service (PaaS)

# Security challenges

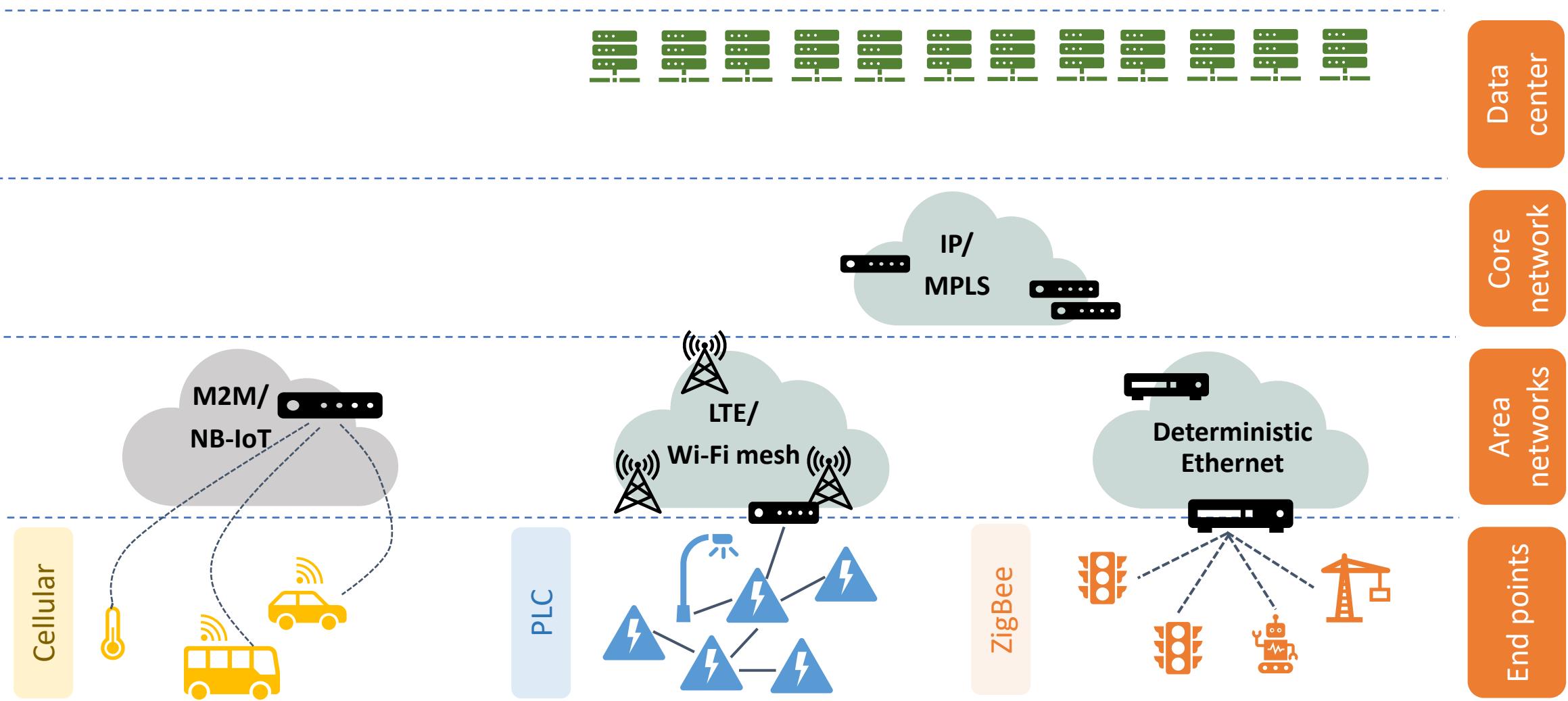
How to secure the entire ecosystems, from hardware to application?

- Hardware isolation (Arm TrustZone)
- Middleware (Speculative Store Bypass Barrier – SSBB)
- Network isolation (Software-defined Networking – SDN)
- Data confidentiality in transit (Transport Layer Security – TLS)
- Software isolation (containers)



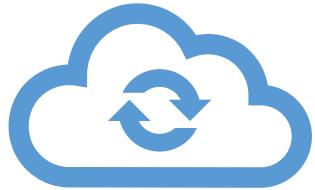
End-to-end security  
not straightforward

# A practical network-centric view



# Cloud vs. Fog vs. Edge

The information processing view



## Cloud computing



Cloud dominated the networked systems landscape until recently



All intelligence on powerful servers, including relational databases, control functions, data analytics engines, web interfaces, etc.

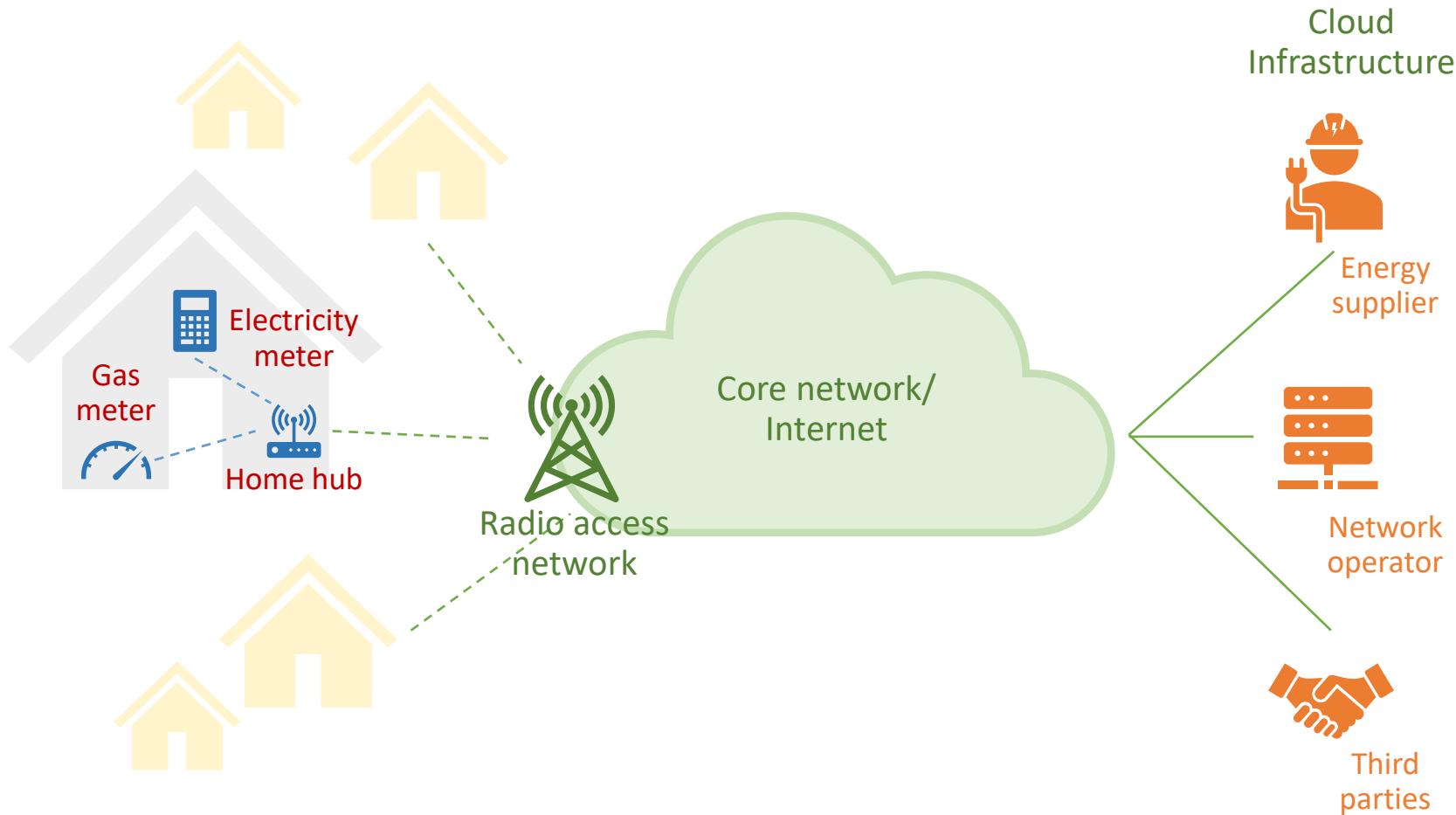


End devices merely information gatherers



Might not scale as the number of IoT devices grows, and applications continue to diversify and generate more data

# Example: Smart metering



- Sensing performed by simple sensors
- Information relayed by home hub over cellular network
- Data processed in the cloud by different stakeholders

# Cloud vs. Fog vs. Edge

The information processing view



Pushing some of the intelligence closer to the device, for e.g., to access networks or gateways



This includes data aggregation, compression, (partial) processing, making localized decisions



IoT devices kept simple, no direct communication with end servers, still battery powered



Resource management implemented across different network layers – management could be regarded as an application



**Fog computing**

# The role of gateways in fog architectures



Data filtering and processing (for e.g., aggregation of summaries, compression, etc.)



Protocol translation and interfacing between different connectivity technologies



Data flow multiplexing, packet routing



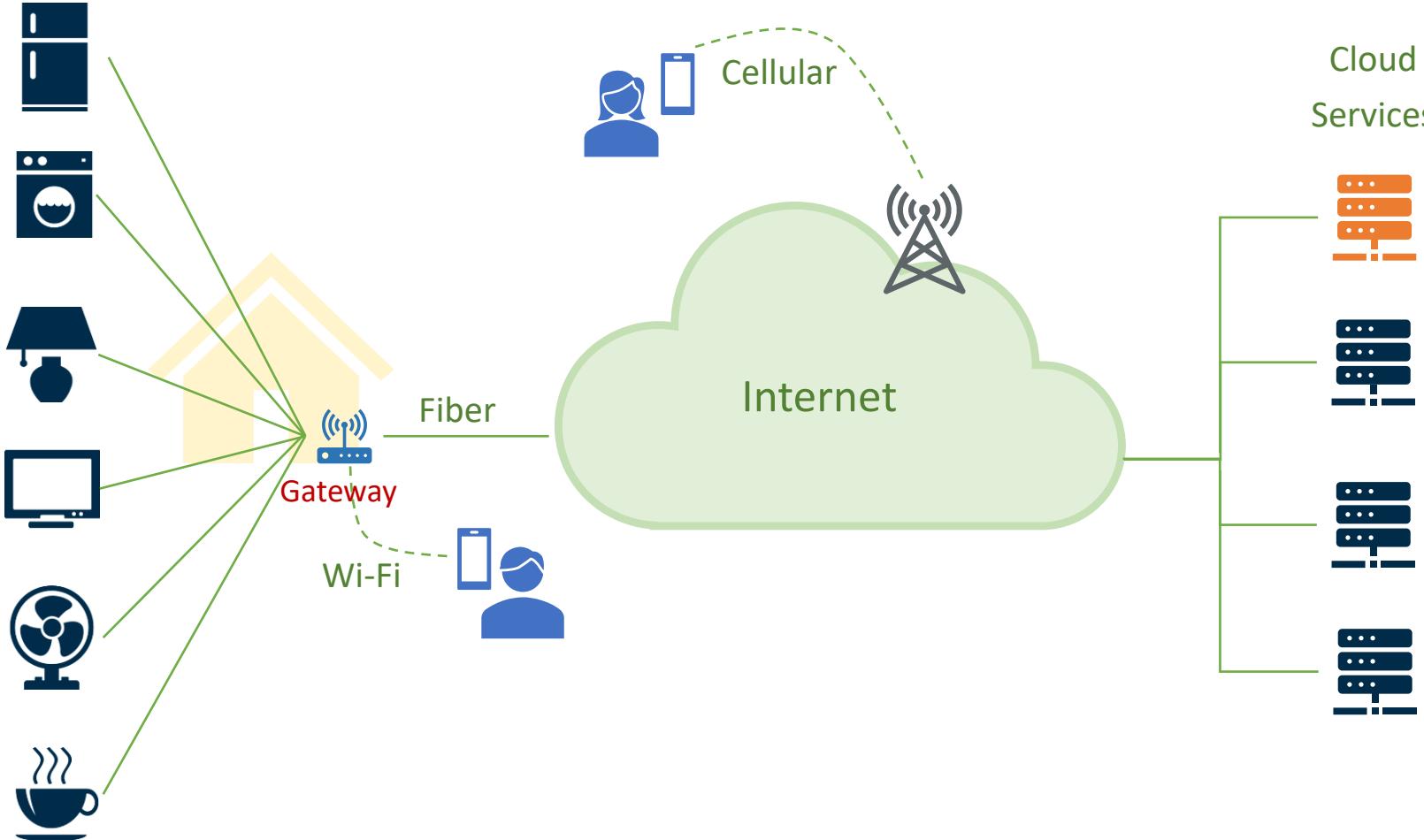
Security (for e.g., data encryption, firewalling)



Scalability problem: as the number of devices grows, so will the number of gateways that are required

# Example: Home automation

## Home appliances



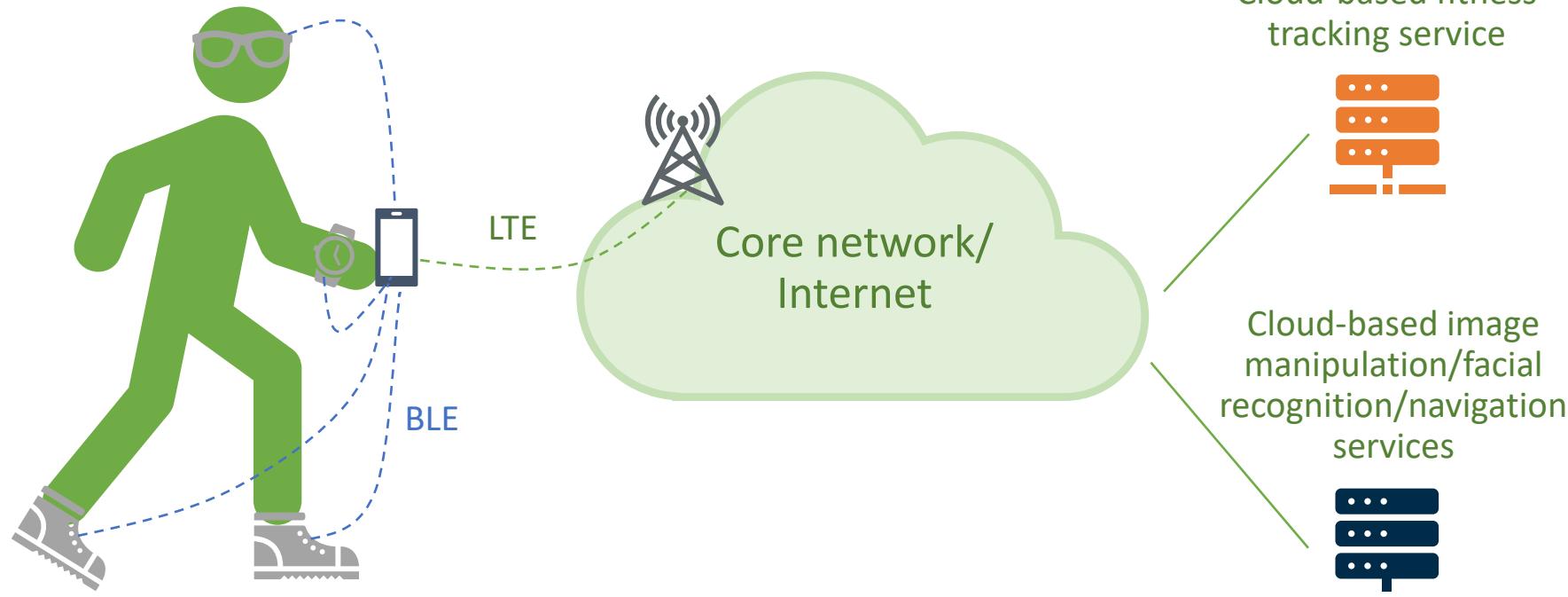
- Gateway performs protocol translation
- Incorporates basic network intrusion detection system
- Cloud services continue to perform analytics

# Smartphones as gateways

The fitness and healthcare domain

- Embed multiple networking technologies (Wi-Fi, 3G/4G, Bluetooth/BLE, NFC, etc.)
- Run full TCP/IP stacks, thus maintain end-to-end connectivity with cloud
- Can connect to multiple devices within close proximity simultaneously
- Ability to enforce secure transport (e.g., TLS/HTTPS)
- Sufficient computing power to pre-process/augment collected data

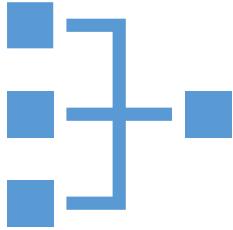
# Example: Wearables



- Smartphone communicates over BLE with wearable devices
- Performs minimal information pre-processing
- Relays data to cloud-based services

# Cloud vs. Fog vs. Edge

The information processing view



## Edge computing



Pushing compute power, communication capabilities, intelligence down at device level



Processing as much as possible where data is collected

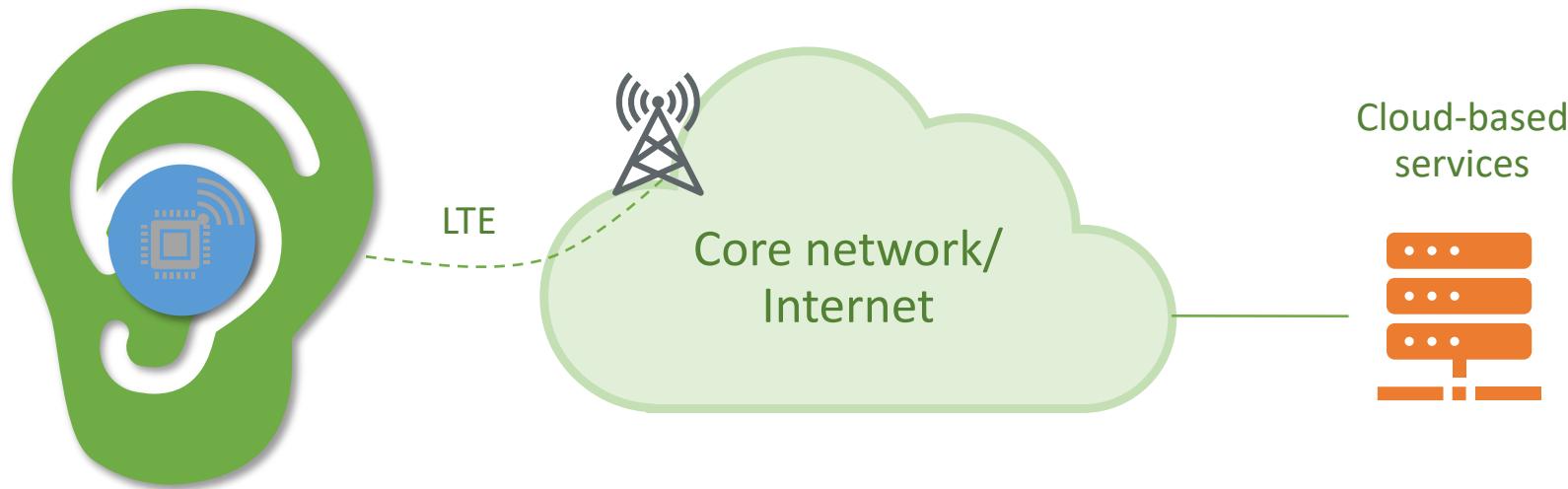


Transmitting only key information or summaries



Enabling new applications: automotive IoT, virtual/augmented reality, in-ear computing

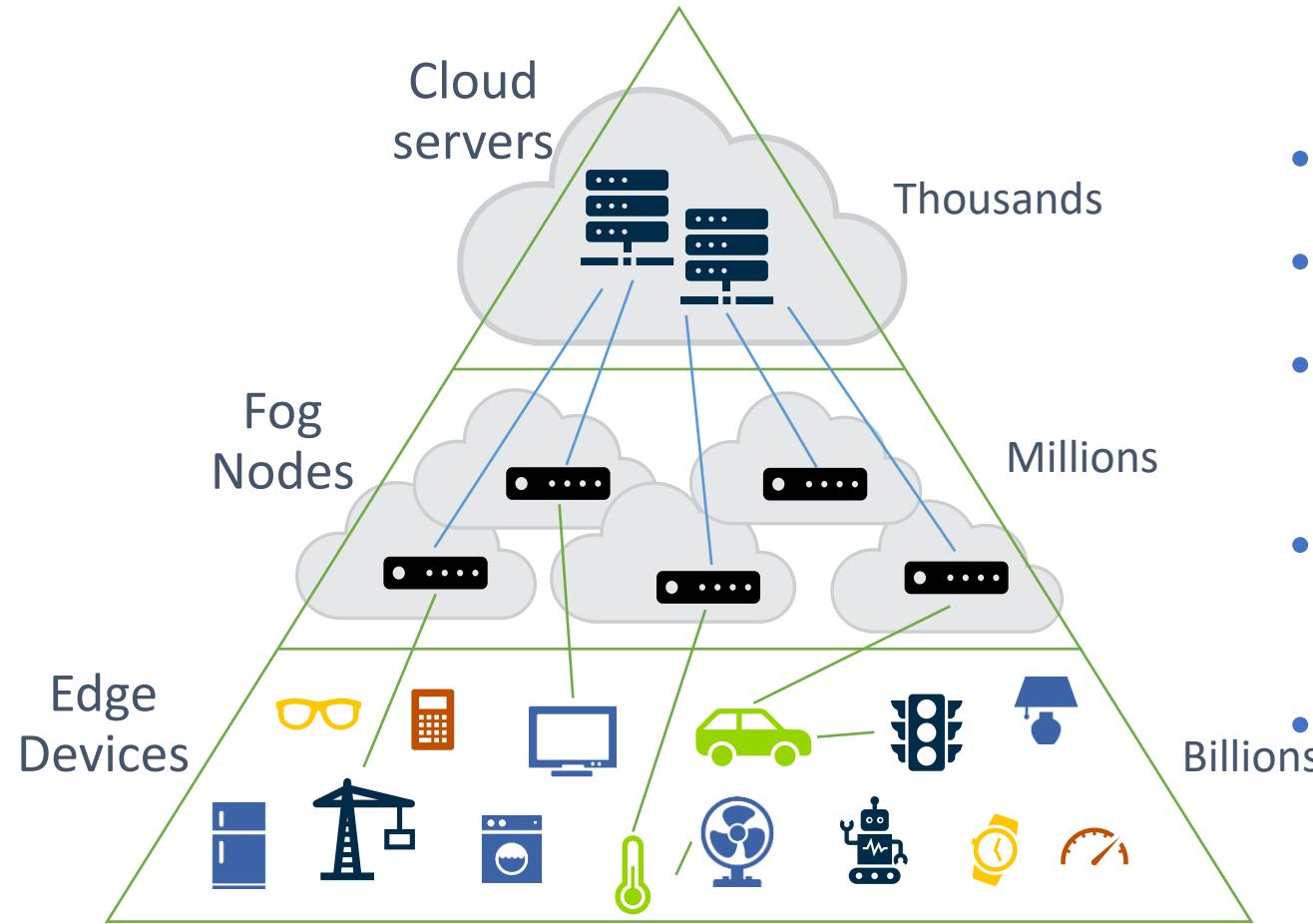
# Example: Hearables



- **Hardware:** Low-power chips specialized in computationally intensive tasks (Arm Ethos)
- **Software:** AI libraries optimized for constrained devices (uTensor)
- **Neural networks:** compressed/pruned models

# Choosing the right IoT architecture

Performance and cost remain the dominant architectural drivers



- What are the application requirements?
- What data needs to be acted on locally?
- Where is most of the computing power?
- How much networking infrastructure should be deployed/used?
- Where are the trust boundaries?

# Standards for IoT

Multiple regulation bodies and industry alliances are standardizing the means by which devices can interact with each other and with gateways or cloud services.

## The Institute of Electrical and Electronics Engineers (IEEE)

- Primarily dealing with defining protocols for (wireless) access networks
- Targeting the Industrial, Scientific, and Medical (ISM) bands (e.g., 2.4GHz, 5GHz, 900MHz in some regions, etc.)
- From an IoT perspective, the most relevant technologies include
  - IEEE 802.15.4, on which ZigBee builds, and
  - IEEE 802.11ah (HaLow) that is an amendment to the IEEE 802.11 specification (typically used for Wi-Fi) that enables low-power wide-area networking

# Standards for IoT

Multiple regulation bodies and industry alliances are standardizing the means by which devices can interact with each other and with gateways or cloud services.

## The 3<sup>rd</sup> Generation Partnership Project (3GPP)

- Focuses on specifying cellular network architectures and protocols (e.g., GSM, 3G, 4G-LTE, etc.)
- Developing standards for cellular communications tailored to IoT applications
  - LTE-M – compatible with existing LTE networks, easy to roll out, limited to 1Mb/s speeds
  - NB-IoT – deployed in same or different frequency bands, lower capacity (200Kb/s), different modulation and coding schemes, and does not require gateways.

# Standards for IoT

## The Internet Engineering Task Force (IETF)

- Focuses on specifying protocols that are used across the Internet; these standards are known as Requests for Comments (RFCs)
- IoT relevant standards include
  - Addressing/internetworking for low power devices (IPv6 over Low-Power Wireless Personal Area Networks – 6LoWPAN)
  - Routing (Routing Over Low-power and Lossy networks – ROLL)
  - End-to-end communications (Constrained Application Protocol – CoAP)
  - Security (Datagram Transport Layer Security – DTLS)
  - Software updating (Software Updates for Internet of Things – SUIT)
- Also offers experience-based guidance
  - Example: The JavaScript Object Notation (JSON) Data Interchange Format – RFC 8259

# Standards for IoT

## Industry alliances

- Bluetooth – wireless personal area networks (WPANs); defines application profiles
- ZigBee – WPANs building on IEEE 802.15.4; inexpensive consumer/industrial applications
- LoRaWAN – LPWAN based on chirp spread spectrum technology

## Collaborative associations

- The Alliance for IoT Innovation (AIoTI) – European Commission framework supporting interaction between IoT players to drive innovation, standardization, and policy.
- Open Connectivity Foundation (OCF) – Industry-led framework aiming to develop IoT standards, interoperability guidelines, and provide a device certification program.

# Standards for IoT

## Other standardization bodies relevant to IoT

- National Institute of Standards and Technology (NIST) – works on a range of science, technology, and engineering topics
  - Example: Advanced Encryption Standard (AES)
- International Organization for Standardization (ISO) – promotes a broad range of proprietary, industrial, and commercial standards
  - Example: Internet of Things (IoT) – Reference Architecture (ISO/IEC 30141:2018)
- International Telecommunication Union (ITU) – recommendations, reference models
  - Example: ITU-T Y.4000/Y.2060 - Overview of the Internet of things

# Choosing among IoT standards is not straightforward



# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# Computer Network Recap

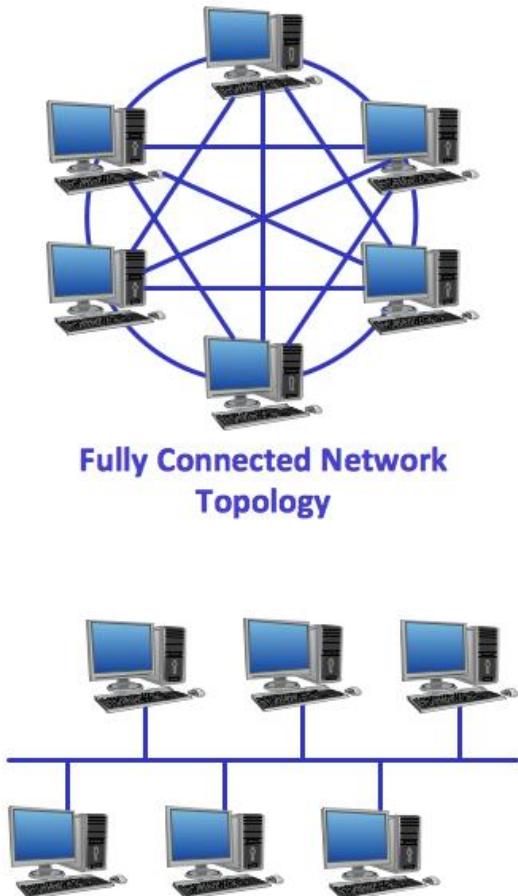
Internet-of-Things (IoT)

COCOS20

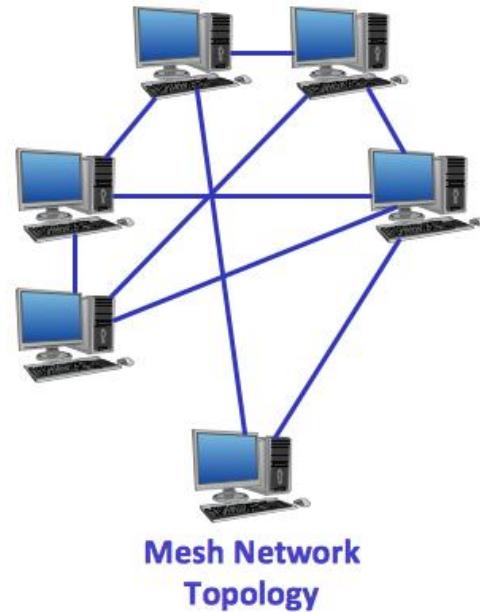
# Computer Network Terminology

- **Network:** group of computers and associated devices that are connected by communication facilities
- **Wide Area Network (WAN):** world-wide (Internet)
- **Metropolitan Area Network (MAN):** city-scale.
- **Local Area Network (LAN):** laboratory/office-scale (Ethernet).
  - **WLAN:** wireless LAN (Wi-Fi).
  - **WPAN:** wireless personal area network (Bluetooth).
  - **WBAN:** wireless body area network.

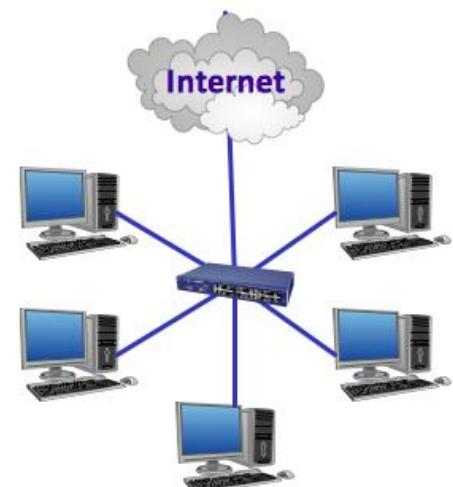
# Network Topologies



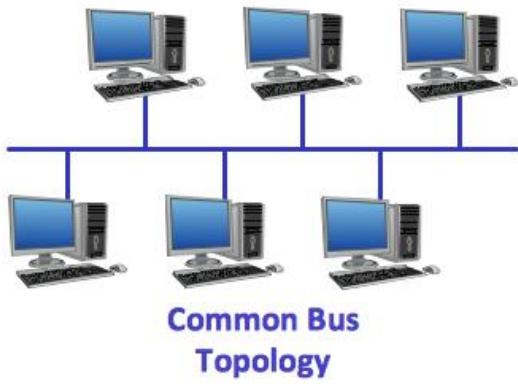
Fully Connected Network  
Topology



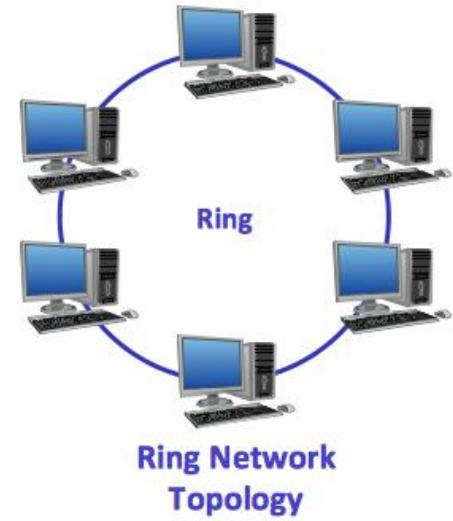
Mesh Network  
Topology



Star Network  
Topology



Common Bus  
Topology



Ring  
Topology

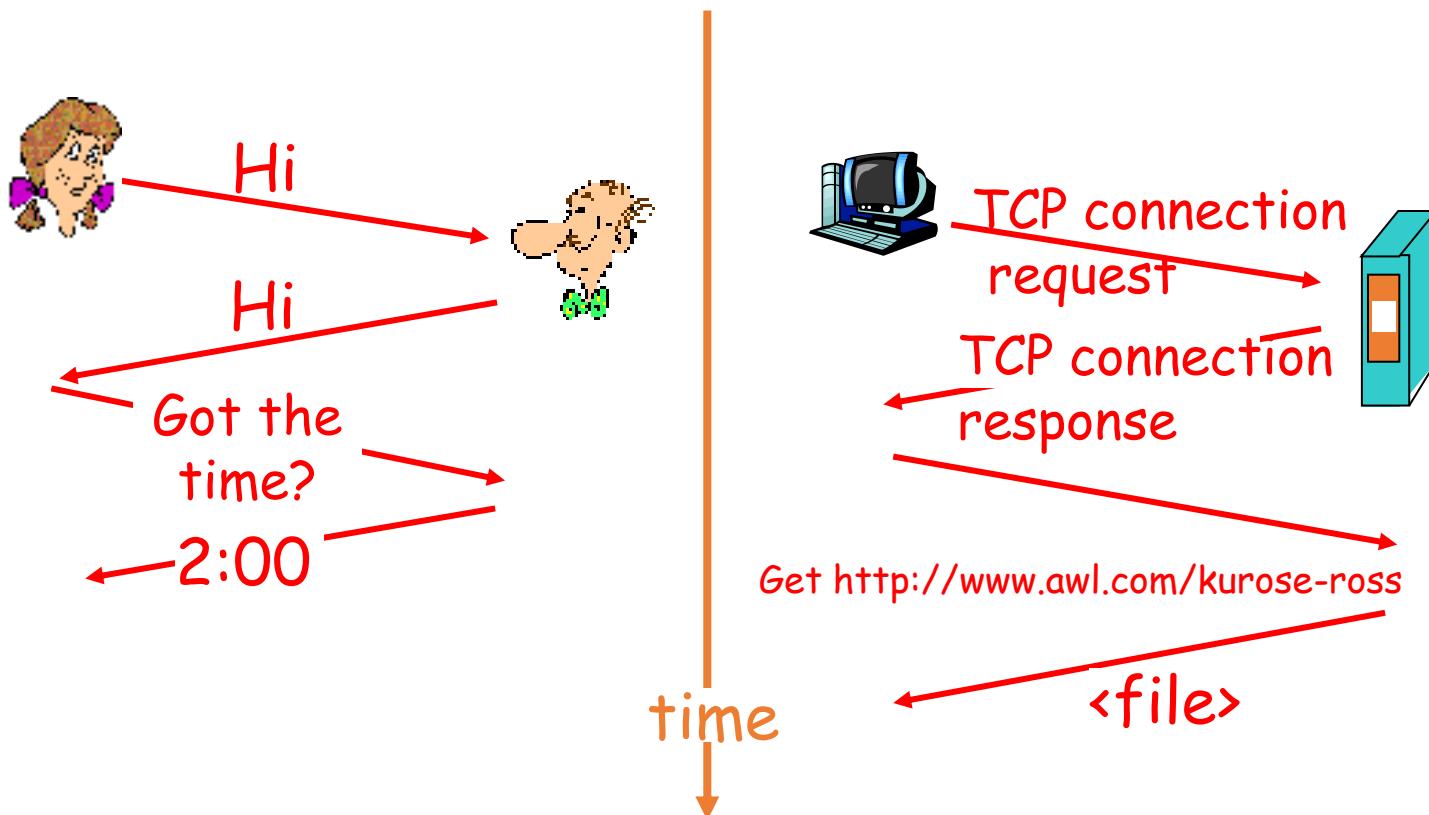
# Network Protocols

- Protocols are the **building blocks** of a network architecture.
- Formal standards and policies enabling communication.
- IEEE (Institute of Electrical and Electronics Engineers): standardization
  - Example: Project 802
    - 802.3: Ethernet
    - 802.11: WLAN
    - 802.15: WPAN

# Communication

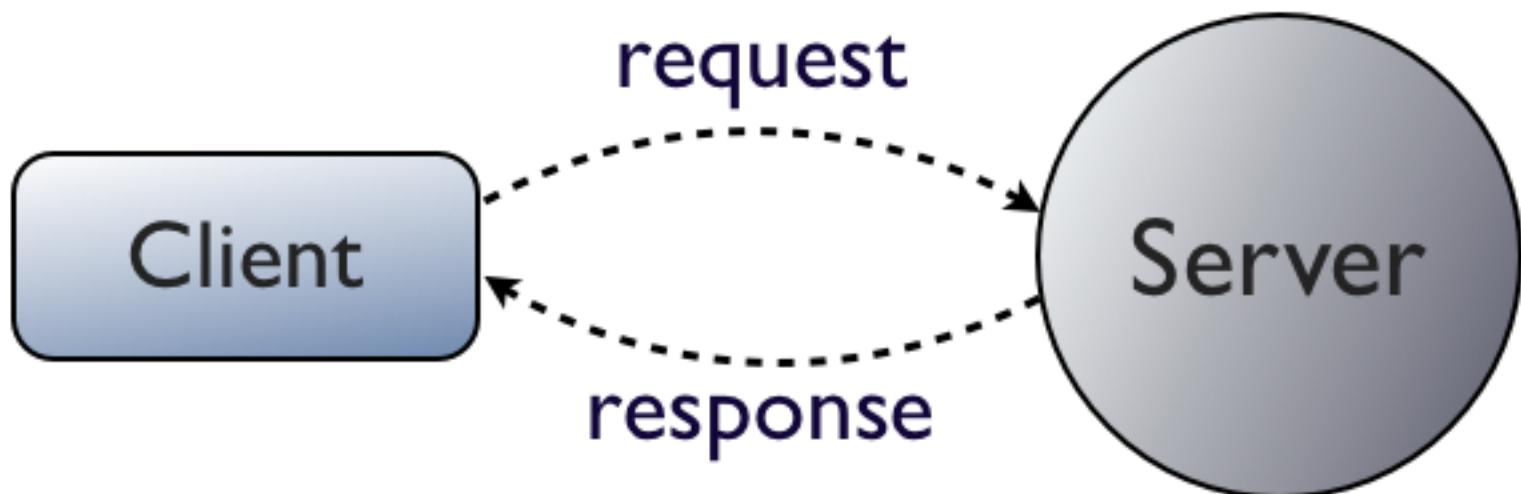
- Who initiates communication?
- Order of communication?
- How long can I talk?
- How loud can I speak?
- Do I have to say something specific at beginning or end?
- Do I have to add meta information?
- What do I do if I get interrupted?
- What do I do if I was not understood?

# Protocols



# Client/Server Model

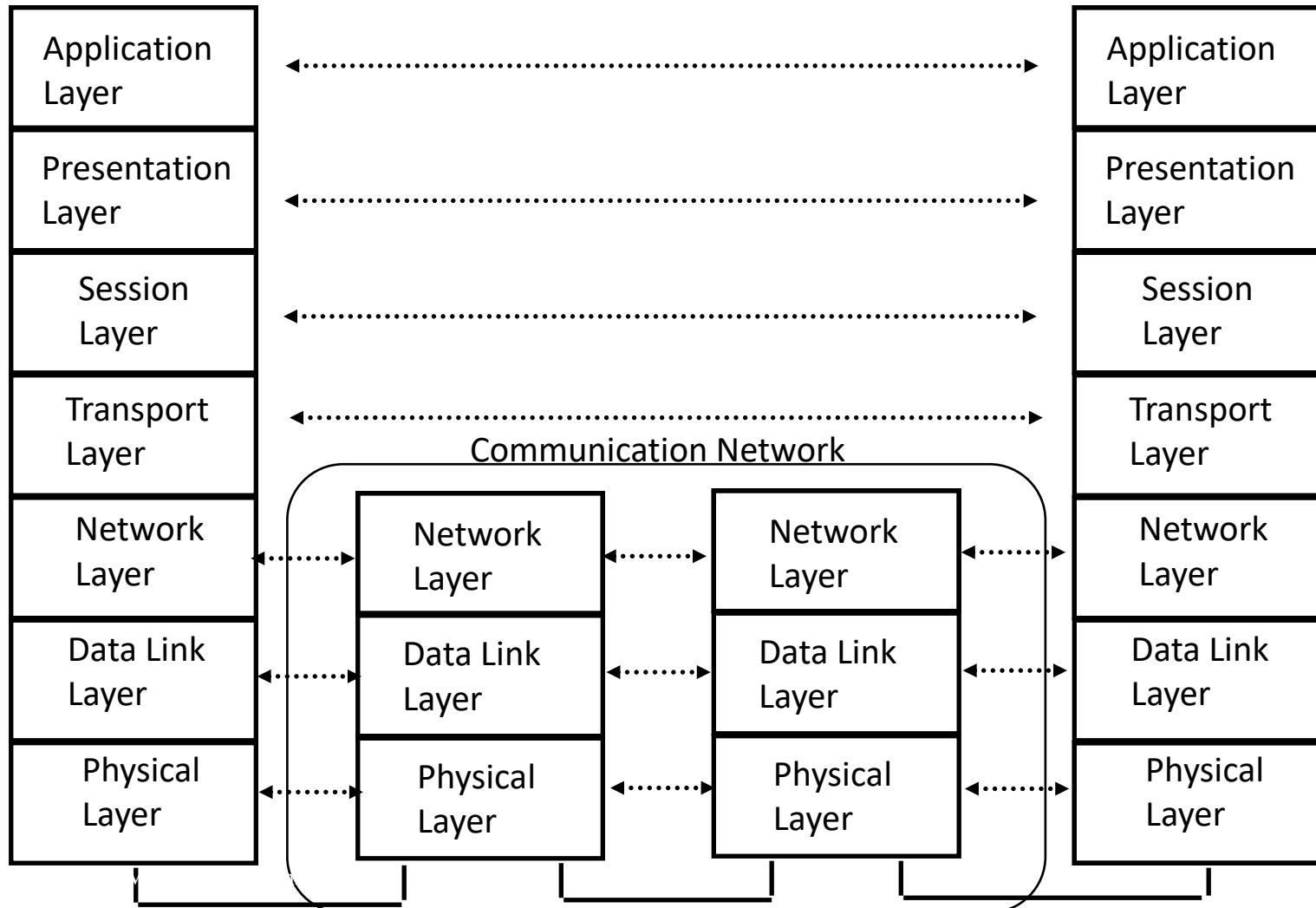
- Client: “active” (initiates communication)
- Server: “passive” (listens and responds)



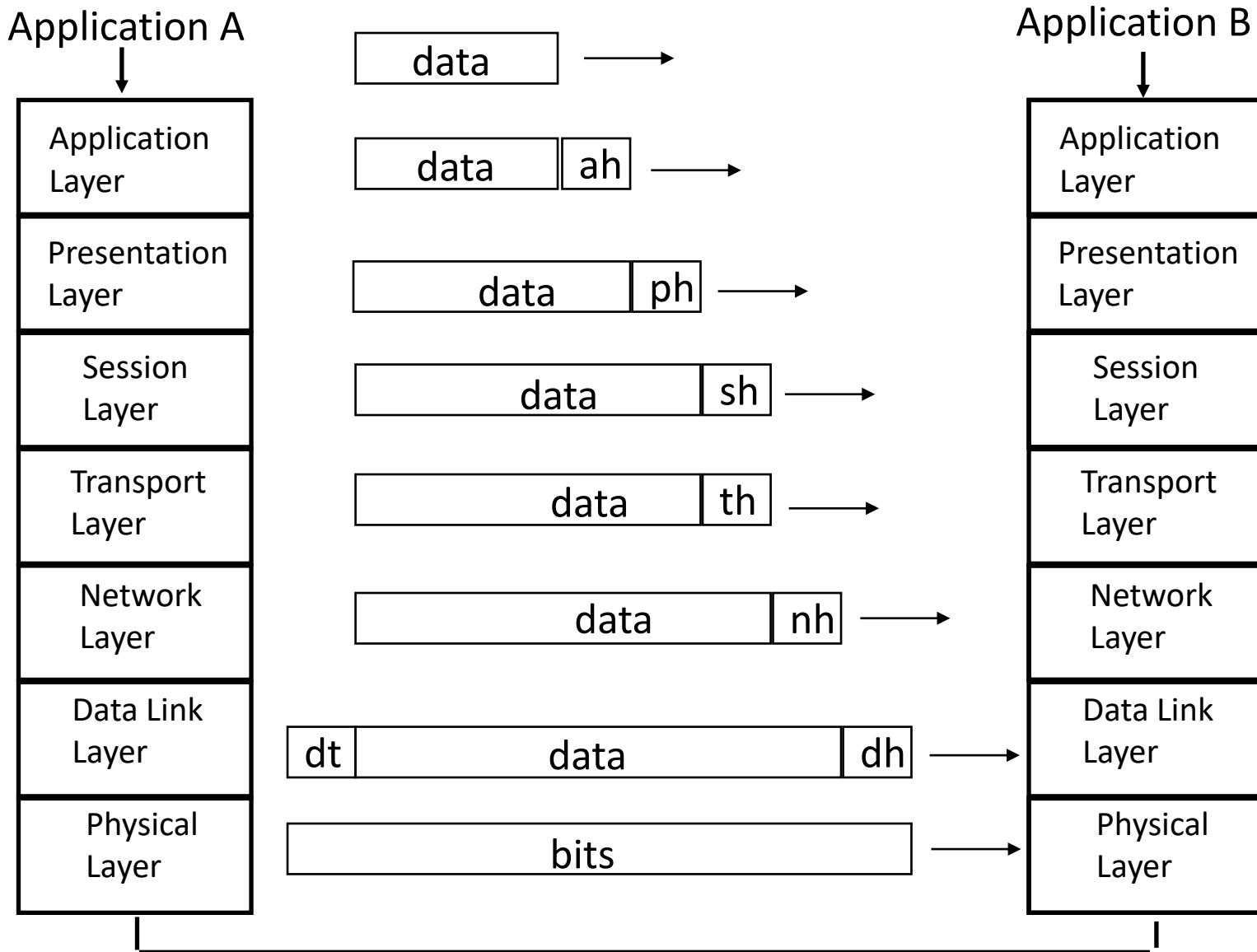
# Client/Server Model Examples

- HTTP (Hypertext Transfer Protocol)
- SMTP (Simple Mail Transfer Protocol)
- SSH (Secure Shell)
- DNS (Domain Name System)
- NFS/AFS (Network/Andrew File System)

# Network Protocols (“Protocol Stack”)



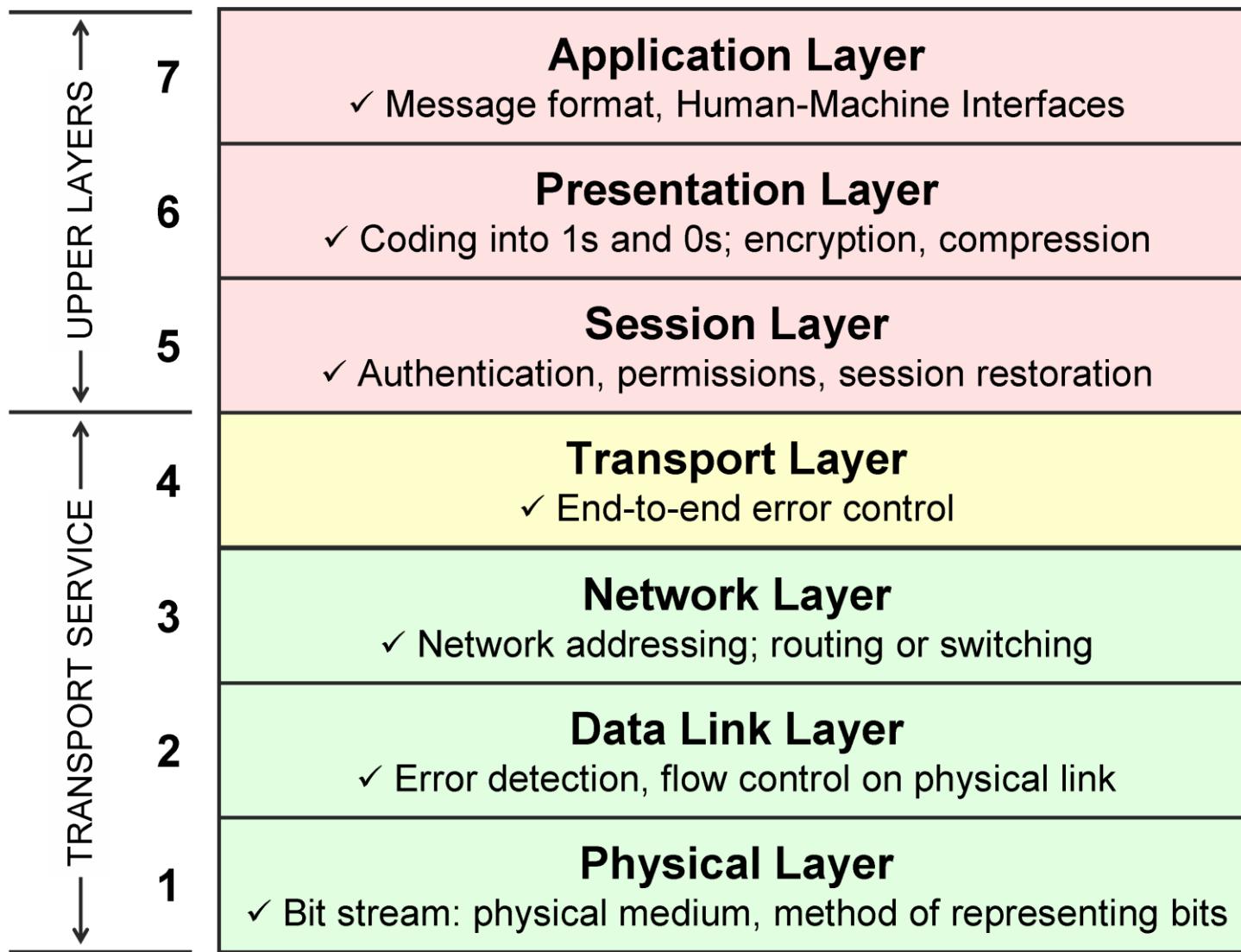
# Network Protocols (Headers/Trailers)



# Why a Layered Design?

- An explicit structure for dealing with a complex system
- Simplifies the design process
- Modularity of layers eases maintenance and updating of system components
- Accommodates incremental changes

# Open System Interconnection (OSI)



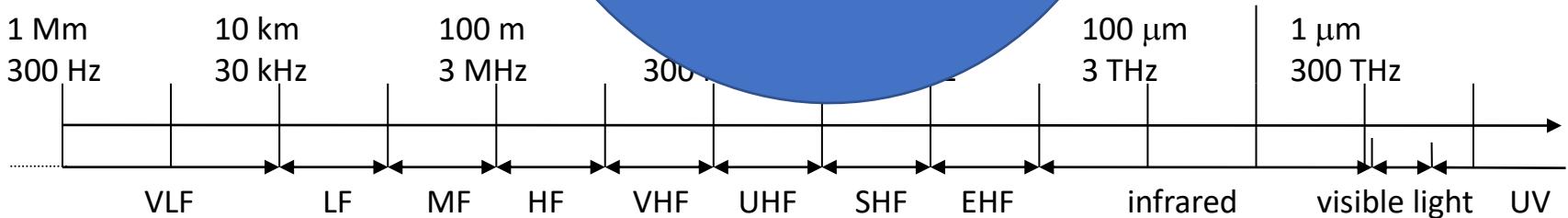
# Physical Layer (Layer 1)

- **Physical/electrical characteristics**
- Cable type, length, connectors, voltage levels, signal durations, ...
- Binary data (bits) as electrical or optical signals
- Frequencies (wireless)

# Wireless Characteristics

- VLF = Very Low Frequency
- LF = Low Frequency
- MF = Medium Frequency
- HF = High Frequency
- VHF = Very High Frequency
- Frequency and wave length
  - $\lambda = c/f$
  - wave length  $\lambda$ , speed of light

What is  
Frequency?



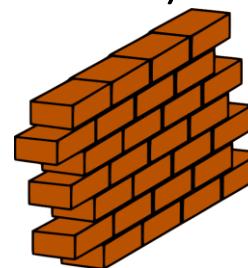
# Frequencies for Mobile Communication

- Low Frequencies:

- low data rates
- travel long distances
- follow Earth's surface
- penetrate objects and water (submarine communication)

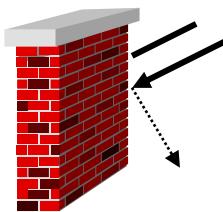
- High Frequencies:

- high data rates
- short distances
- straight lines
- cannot penetrate objects ("Line of Sight" or LOS)

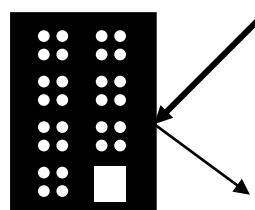


# Other Propagation Effects

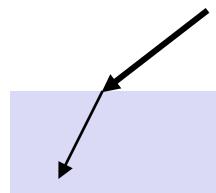
- **Shadowing**
- **Reflection** at large obstacles
- **Refraction** depending on the density of a medium
- **Scattering** at small obstacles
- **Diffraction** at edges



shadowing



reflection



refraction



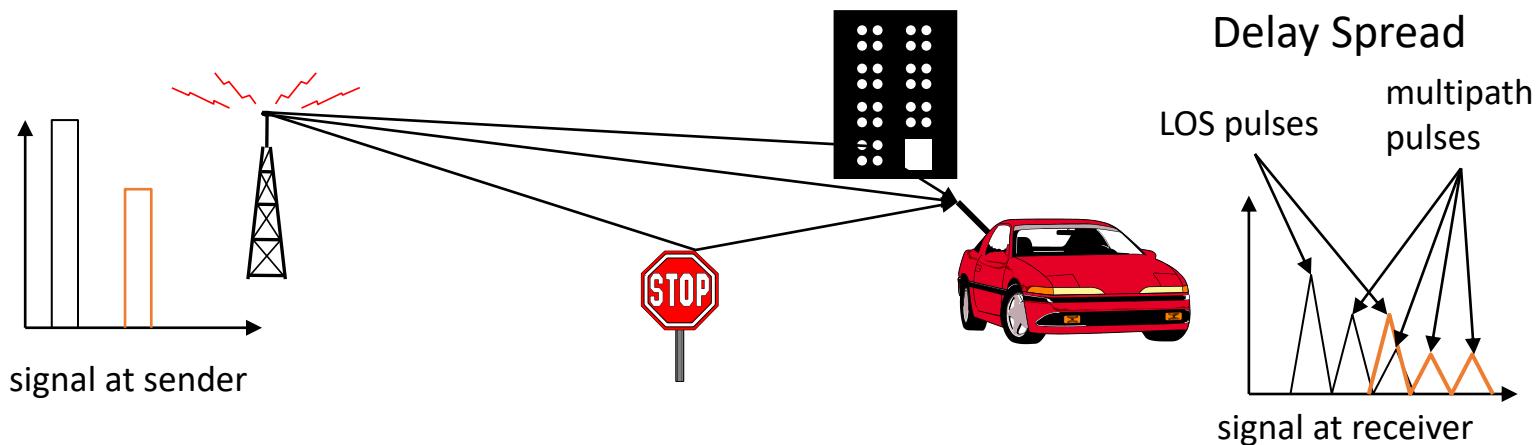
scattering



diffraction

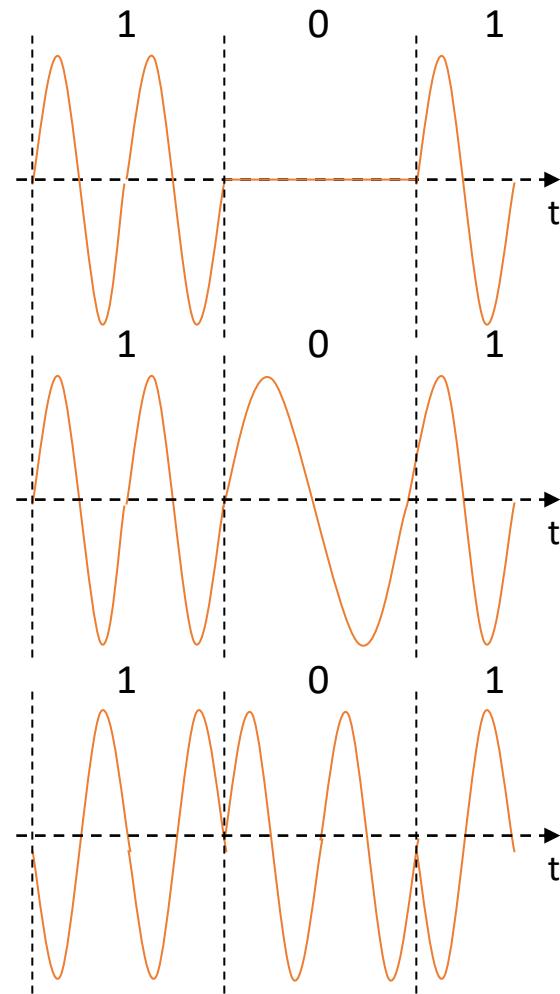
# Multipath Propagation

- Signal can take **many different paths** between sender and receiver due to reflection, scattering, diffraction



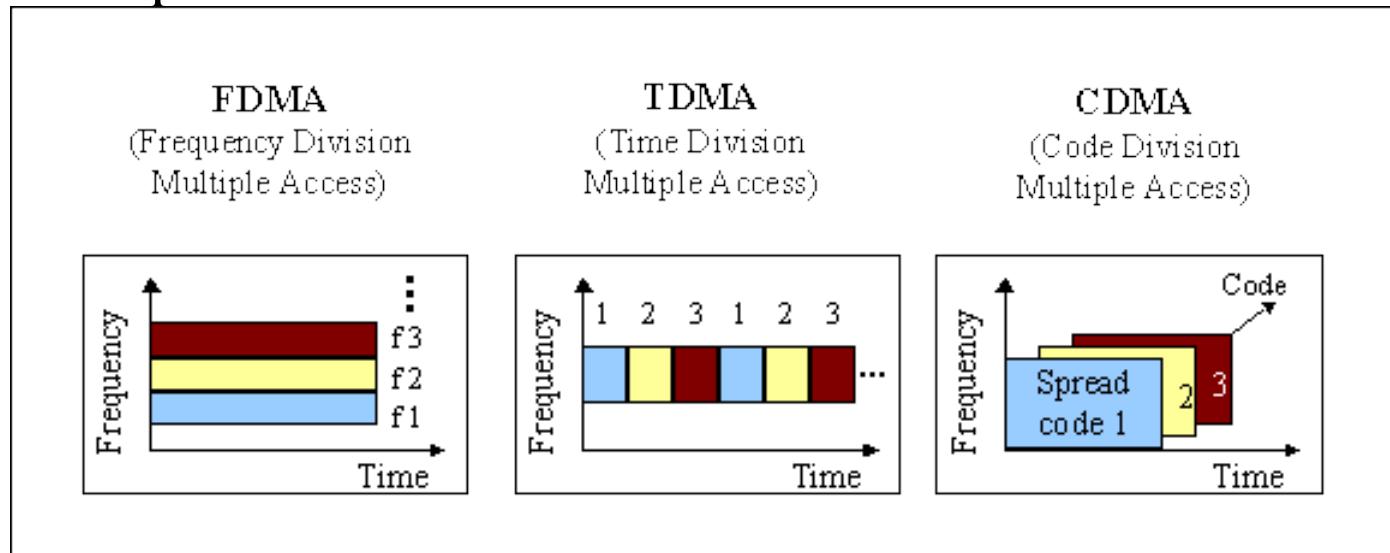
# Digital Modulation

- Amplitude Shift Keying (ASK)
- Frequency Shift Keying (FSK)
- Phase Shift Keying (PSK)



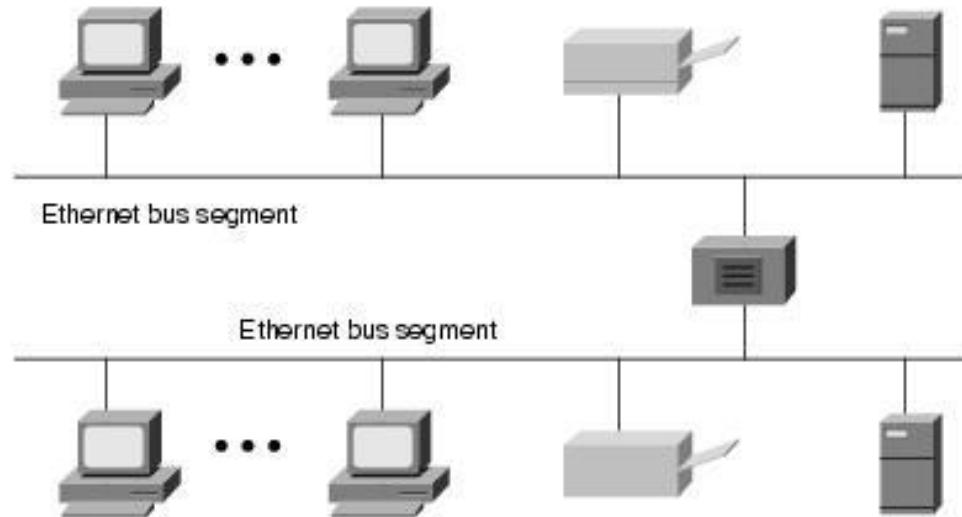
# Data Link Layer (Layer 2)

- **Defines when/how medium will be accessed for transmission**
- Units typically called “frames”; error detection/correction; divided into sublayers, including: **MAC = Medium Access Control** (MAC address 6f:00:2b:23:1f:32)
- Cell phone example:



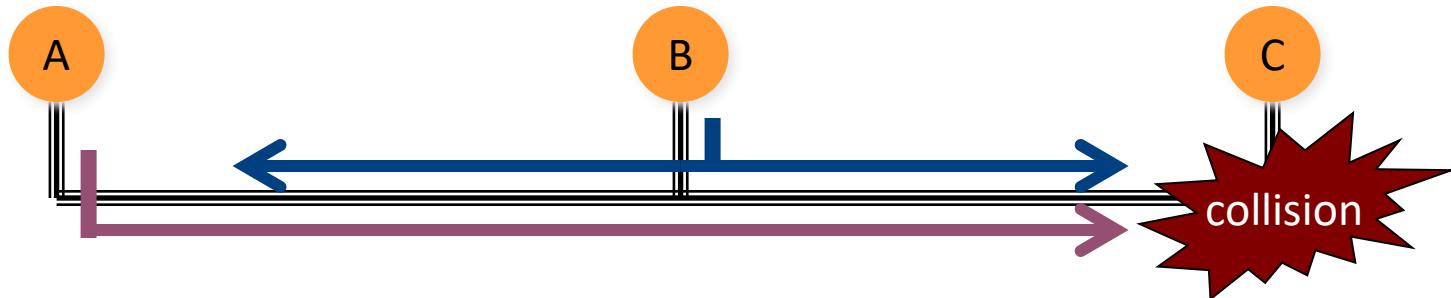
# Example: Ethernet (802.3)

- Most popular LAN technology, uses bus architecture
- Easy to install, inexpensive
- Data is broken into **packets**

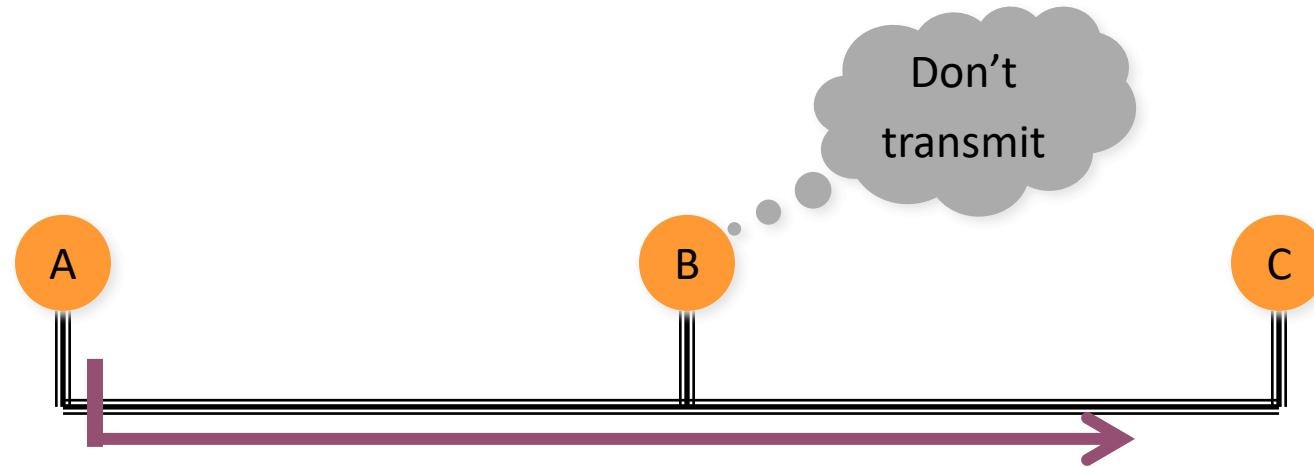


# Example: Ethernet

- Medium Access Control (MAC) protocol
- **CSMA/CD Protocol**
  - Carrier Sense
  - Multiple Access
  - Collision Detection



# Example: Ethernet

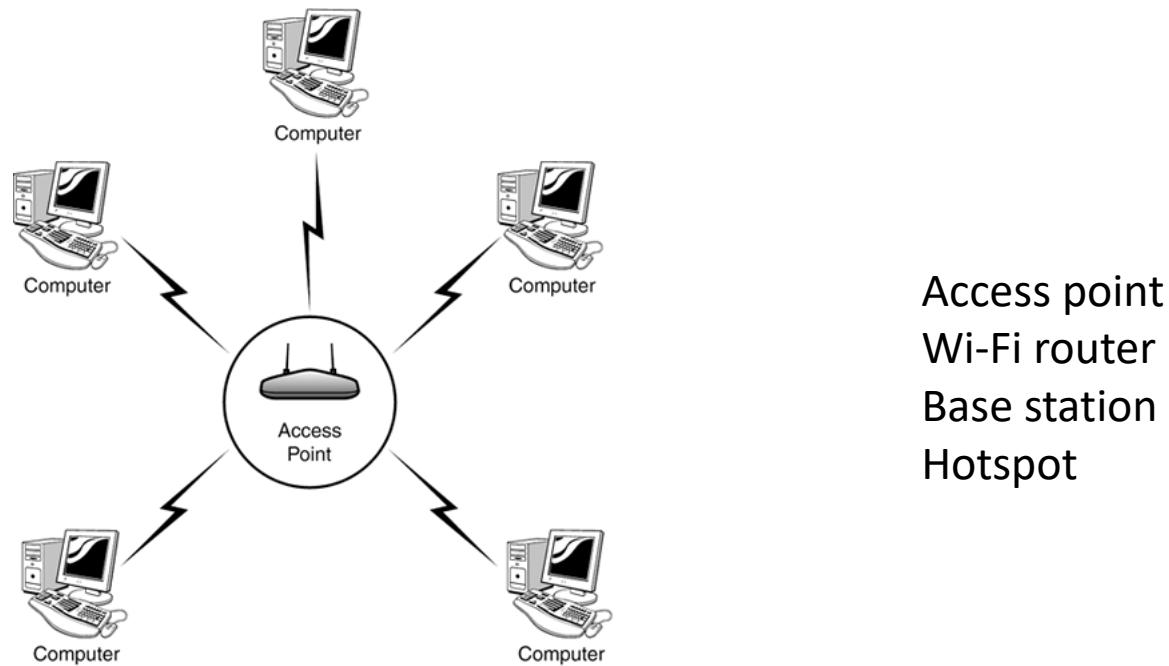


Can collisions still occur?

- “Sense” (listen) carrier (“is anyone else talking right now?”)
- If “busy”: wait; if “idle”: transmit
- CD: Keep listening while transmitting
  - If collision detected: retry at a later time

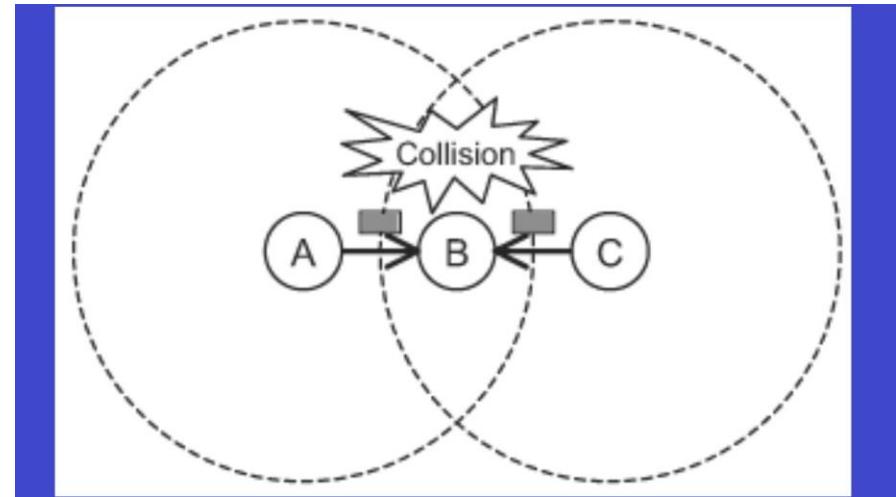
# Example: Wi-Fi (802.11)

- Most popular wireless LAN architecture



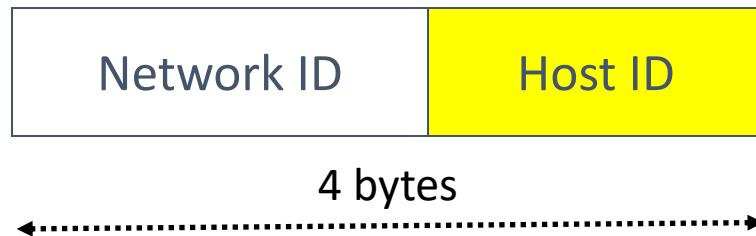
# Example: Wi-Fi (802.11)

- **CSMA/CA Protocol**
  - Carrier Sense
  - Multiple Access
  - Collision Avoidance
    - Channel reservations:
      - Transmitter sends request-to-send (RTS)
      - Receiver sends clear-to-send (CTS)
    - Advantages:
      - Nodes hearing RTS and/or CTS keep quiet
      - If collision, only small RTS or CTS packets are lost.



# Network Layer (Layer 3)

- **Dominant protocol: IP = Internet Protocol**
- Addressing and routing (sender & receiver IP address)
- Uses 32-bit **hierarchical address space** with location information embedded in the structure



- IPv4 address is usually expressed in dotted-decimal notation, e.g.:

**128.100.11.56**

# IPv4

**Class A**  
Subnet Mask

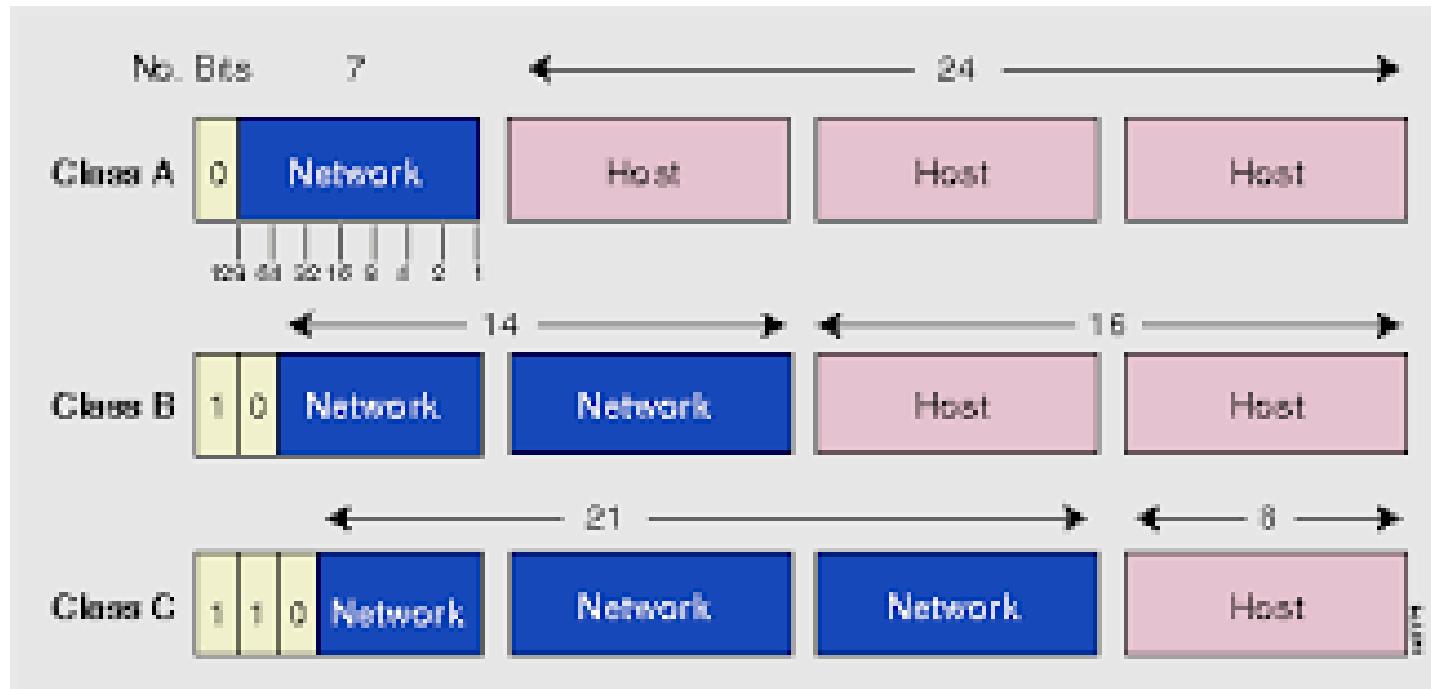
Netwok	Host	Host	Host
255	0	0	0

**Class B**  
Subnet Mask

Netwok	Network	Host	Host
255	255	0	0

**Class C**  
Subnet Mask

Netwok	Network	Network	Host
255	255	255	0

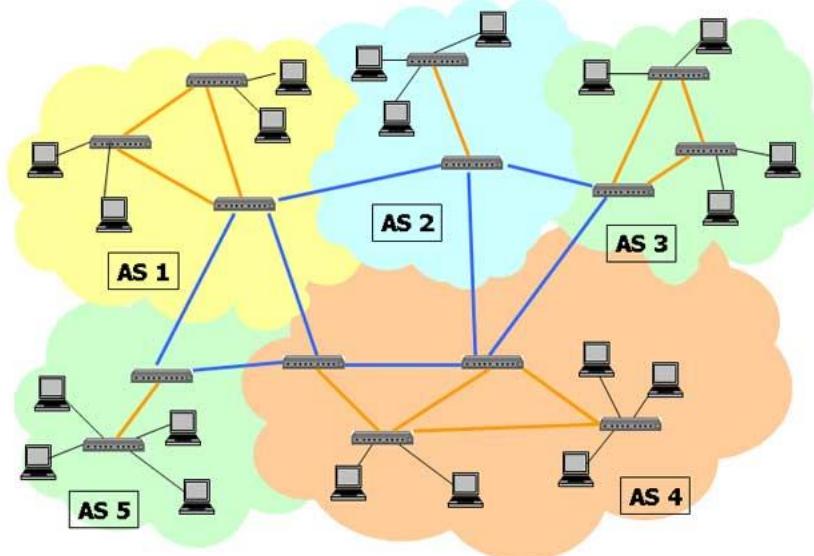


# IPv6

- IPv6 addresses are 128 bits long
- 16 bytes of IPv6 address are represented as a group of hexadecimal digits, separated by colons, e.g.:  
**2000:fdb8:0000:0000:0001:00ab:853c:39a1**
- Shorthand – leave out groups of zeros and leading zeros:  
**2000:fdb8::1:ab:853c:39a1**
- IPv4 Address space: 4,294,967,296 Addresses
- IPv6 Address space:  $(3.4 * 10^{38})$   
340,282,366,920,938,463,463,374,607,431,768,211,456

# Routers

- Form backbone of the Internet
- Use IP layer to identify source and destination of packets
- Look up **routing tables** that determines “**next hop**”



Destination	Next Hop
147.39.21.X	131.19.18.121
89.44.X.X	131.19.22.119
203.21.X.X	137.18.47.48

# Transport Layer (Layer 4)

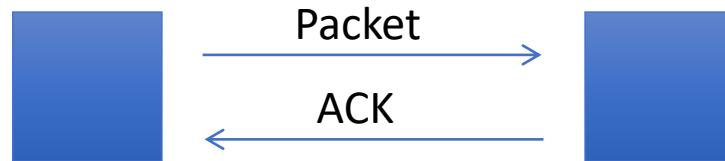
- **UDP (User Datagram Protocol)**



- Adds more addressing: “**ports**”
  - IP address tell you which computer
  - Ports tell you which application on that computer
  - Example: a web server “listens” to requests on port 80
  - Web browser: <http://www.google.com:80> = <http://216.58.216.100:80>
    - “:80”: optional
- **Unreliable!**
  - Packets can get lost; packets can arrive out of order

# Transport Layer

- **TCP** (Transmission Control Protocol)
- **Reliable** protocol!
- Adds ports (just like UDP), but also provides:
  - In-order delivery of packets (using sequence numbers)
  - Reliable delivery: using acknowledgment (ACK) packets



- **Flow control & congestion control:**
  - Allows receiver to slow down sender
  - Allows “network” to slow down sender

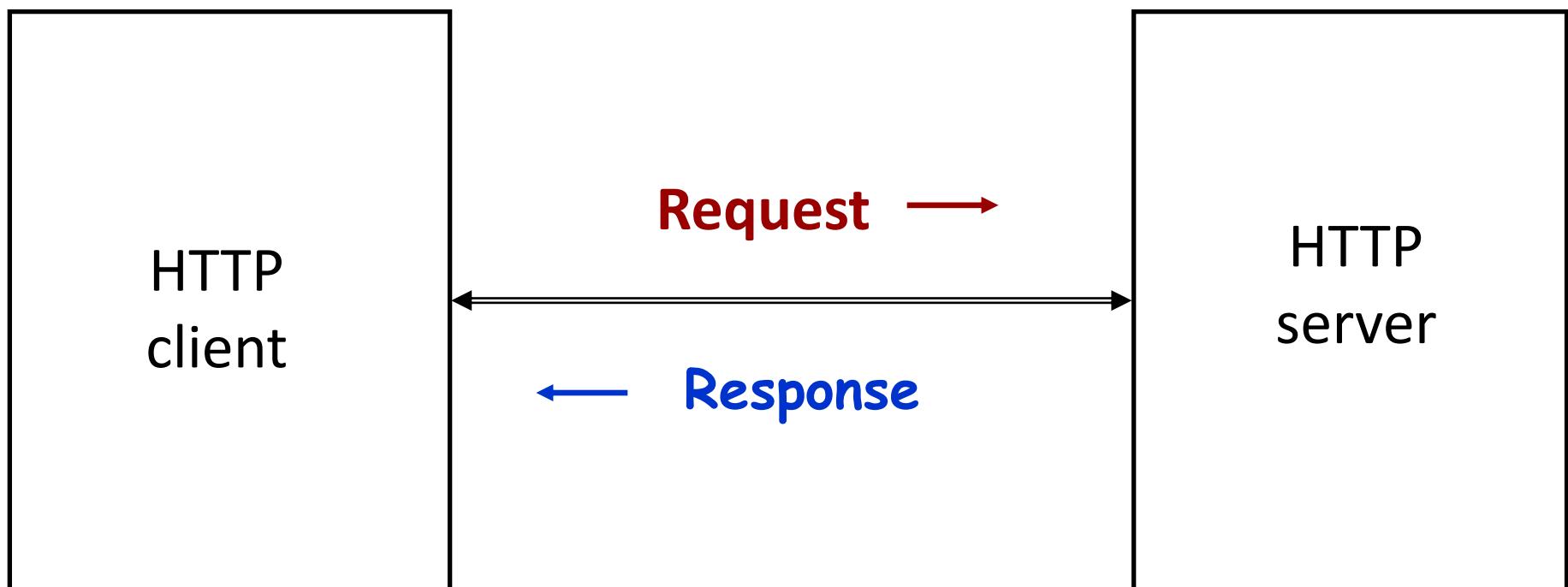
# UDP vs TCP

- TCP:
  - typical choice of most applications
  - do not want to lose data, out-of-order arrival, etc.
  - email, web traffic, financial transactions, etc.
- UDP:
  - can be “faster”
    - no flow/congestion control “slowing down” traffic
    - no retransmissions
    - good for “real-time” traffic
  - out-of-order arrival: can also “reorder” at application level
  - loss of data: can be acceptable
    - missing frames in video/audio stream

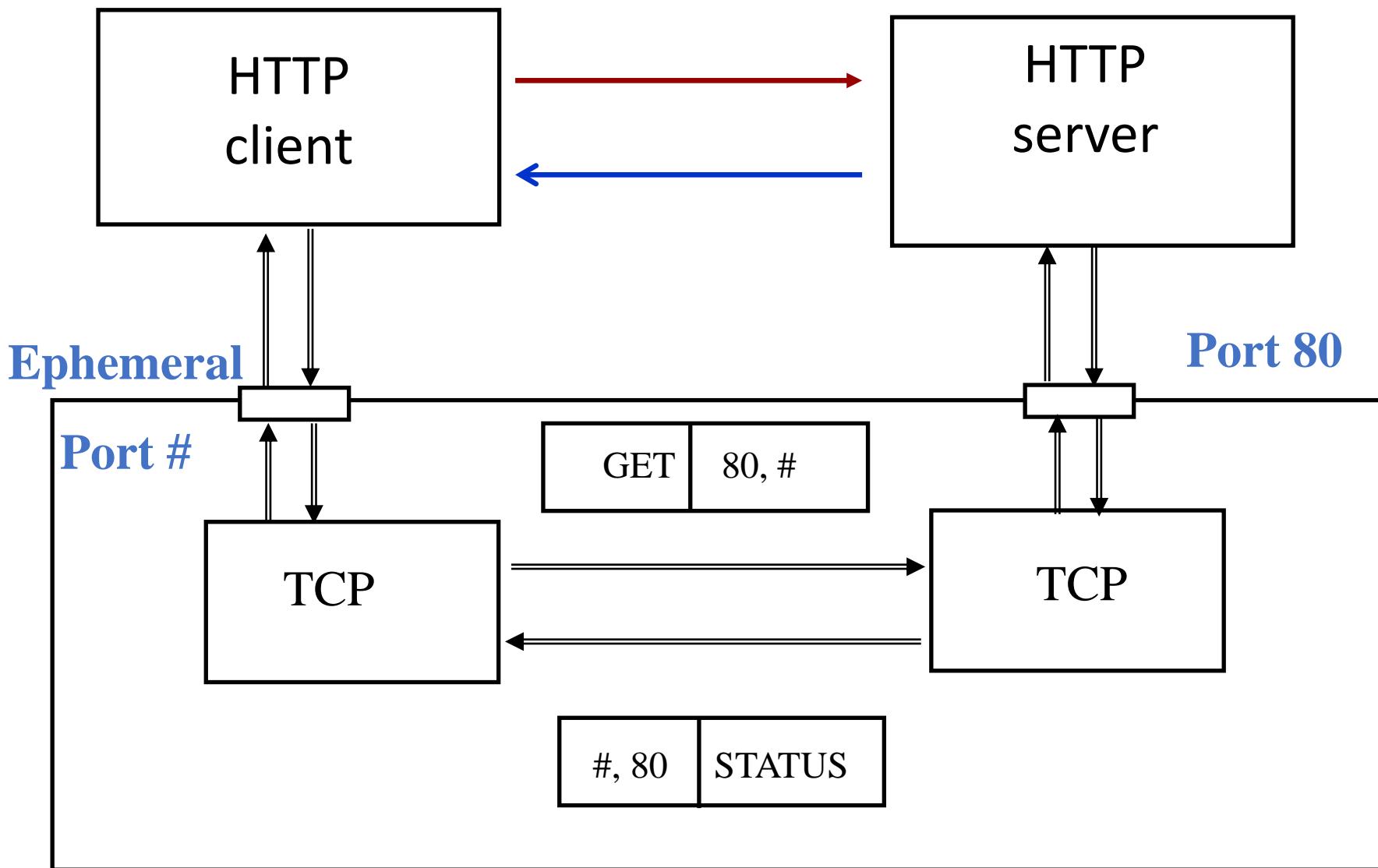
# Upper Layers (Layers 5-7)

- Session Layer
  - Management of “sessions”
- Presentation Layer
  - Data translation, formatting, encryption, compression
- Application Layer
  - Interface between user applications and lower network services

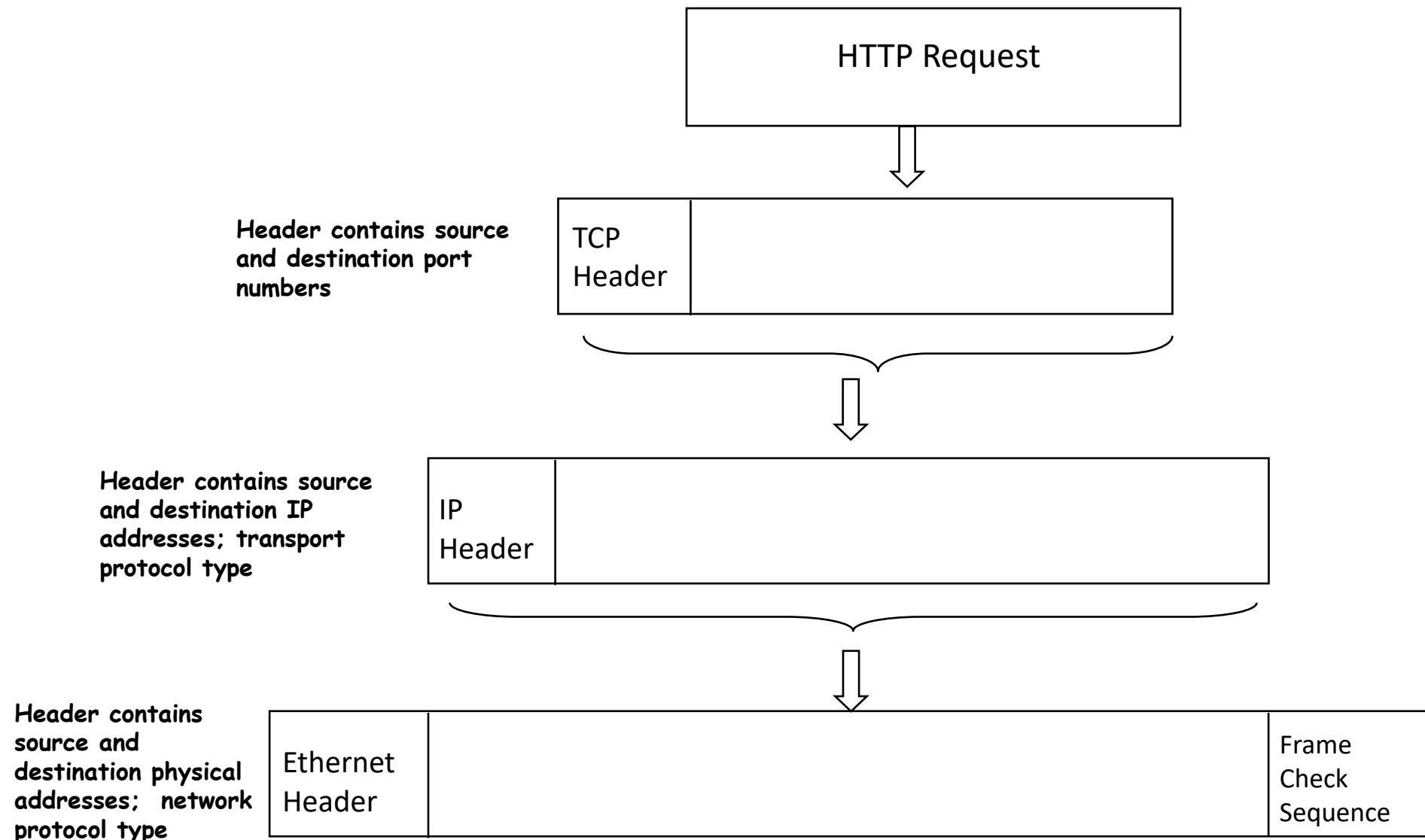
## Example: Web Servers



## Example: Web Servers



# Example: Web Servers



# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# IoT Architectural View

---

COCSC20

# Basic Premises

**Devices**

send and receive data interacting with the

**Network**

where the data is transmitted, normalized, and filtered using

**Edge Computing**

before landing in

**Data storage / Databases**

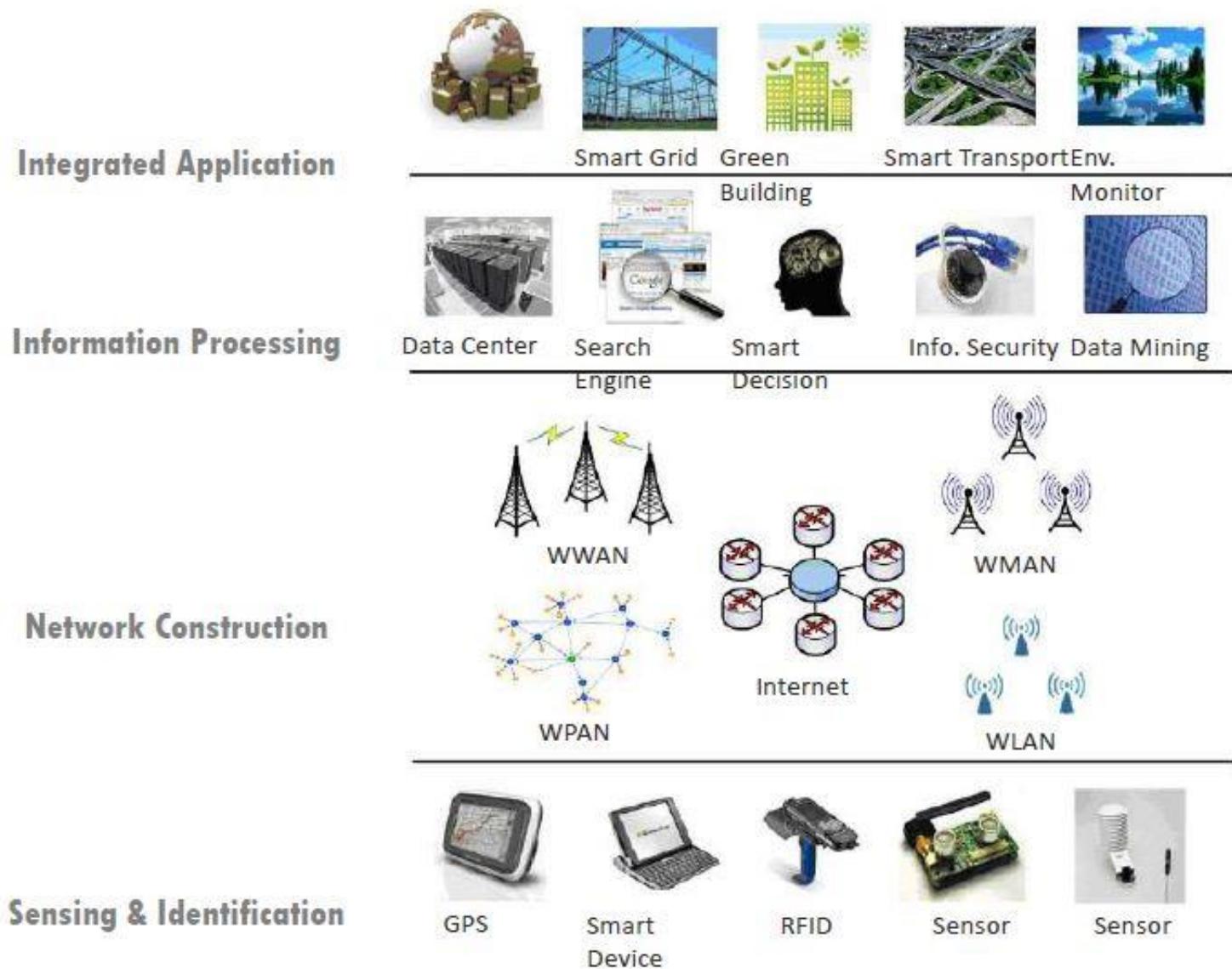
accessible by

**Applications**

which process it and provide it to people who will

**Act and Collaborate**

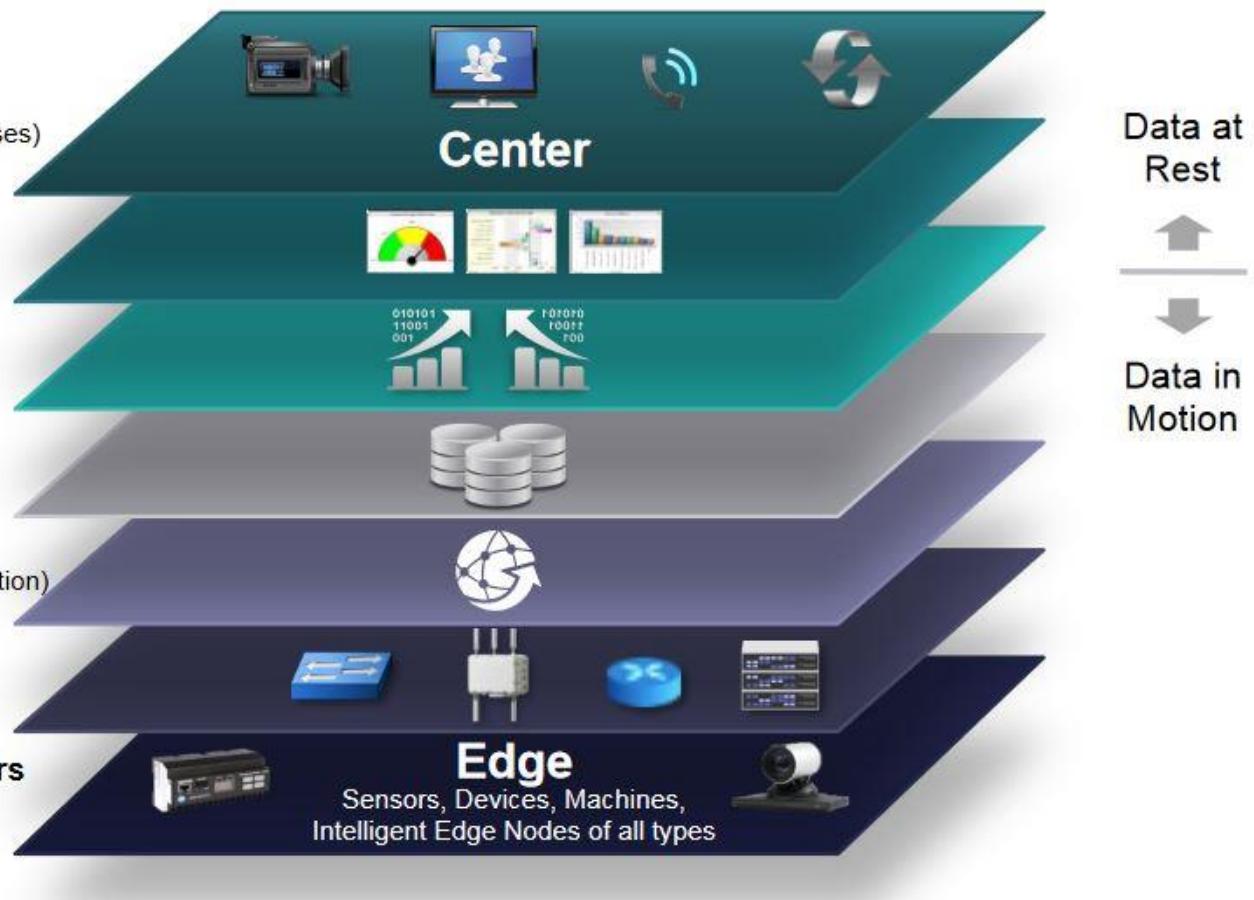
# IoT 4 Layers model



# Reference Model

## Levels

- 7 **Collaboration & Processes**  
(Involving People & Business Processes)
- 6 **Application**  
(Reporting, Analytics, Control)
- 5 **Data Abstraction**  
(Aggregation & Access)
- 4 **Data Accumulation**  
(Storage)
- 3 **Edge (Fog) Computing**  
(Data Element Analysis & Transformation)
- 2 **Connectivity**  
(Communication & Processing Units)
- 1 **Physical Devices & Controllers**  
(The "Things" in IoT)



# 1

## Physical Devices & Device Controllers (The “Things” in IoT)

IoT “devices” are capable of:

- Analog to digital conversion, as required
- Generating data
- Being queried / controlled over-the-net



### Edge

Sensors, Devices, Machines,  
Intelligent Edge Nodes of all types

## 2

## Connectivity

(Communication & Processing Units)

Level 2 functionality focuses  
on East-West communications

Connectivity includes:

- Communicating with and between the Level 1 devices
- Reliable delivery across the network(s)
- Implementation of various protocols
- Switching and routing
- Translation between protocols
- Security at the network level
- (Self Learning) Networking Analytics



3

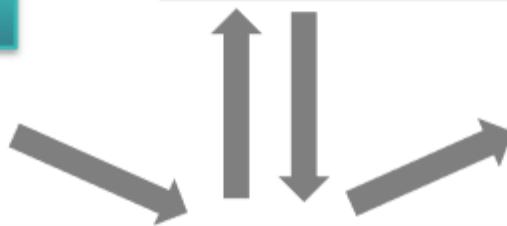
## Edge (Fog) Computing (Data Element Analysis & Transformation)

Level 3 functionality focuses on North-South communications

Include;

- Data filtering, cleanup, aggregation
- Packet content inspection
- Combination of network and data level analytics
- Thresholding
- Event generation

Data packets



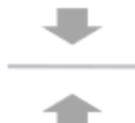
Information understandable to the higher levels

# 4

## Data Accumulation (Storage)

- Event filtering/sampling
- Event comparison
- Event joining for CEP
- Event based rule evaluation
- Event aggregation
- Northbound/southbound alerting
- Event persistence in storage

Query Based Data Consumption



Event Based Data Generation

Making network data usable by applications

1. Converts data-in-motion to data-at-rest
2. Converts format from network packets to database relational tables
3. Achieves transition from 'Event based' to 'Query based' computing
4. Dramatically reduces data through filtering and selective storing



or



# 5

## Data Abstraction (Aggregation & Access)

Abstracting the data interface for applications



## Information Integration

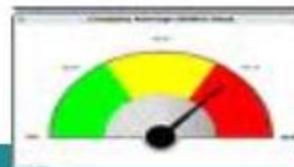
1. Creates schemas and views of data in the manner that applications want
2. Combines data from multiple sources, simplifying the application
3. Filtering, selecting, projecting, and reformatting the data to serve the client applications
4. Reconciles differences in data shape, format, semantics, access protocol, and security



# 6

## Application

(Reporting, Analytics, Control)



Control  
Applications



Vertical and  
Mobile  
Applications



Business  
Intelligence  
and Analytics

# 7

## Collaboration & Processes

(Involving people and business processes)



Center

# How Many Layers in OSI model?

- A. Four
- B. Five
- C. Six
- D. Seven
- E. None of the above.

# TCP/IP stands for?

---

Transmission Control Protocol/Internet Networking Protocol have

- A. Four
- B. Five
- C. Six
- D. Seven
- E. None of the above.

## OSI MODEL

Application Layer

Presentation Layer

Session Layer

Transport Layer

Network Layer

Data Link Layer

Physical Layer

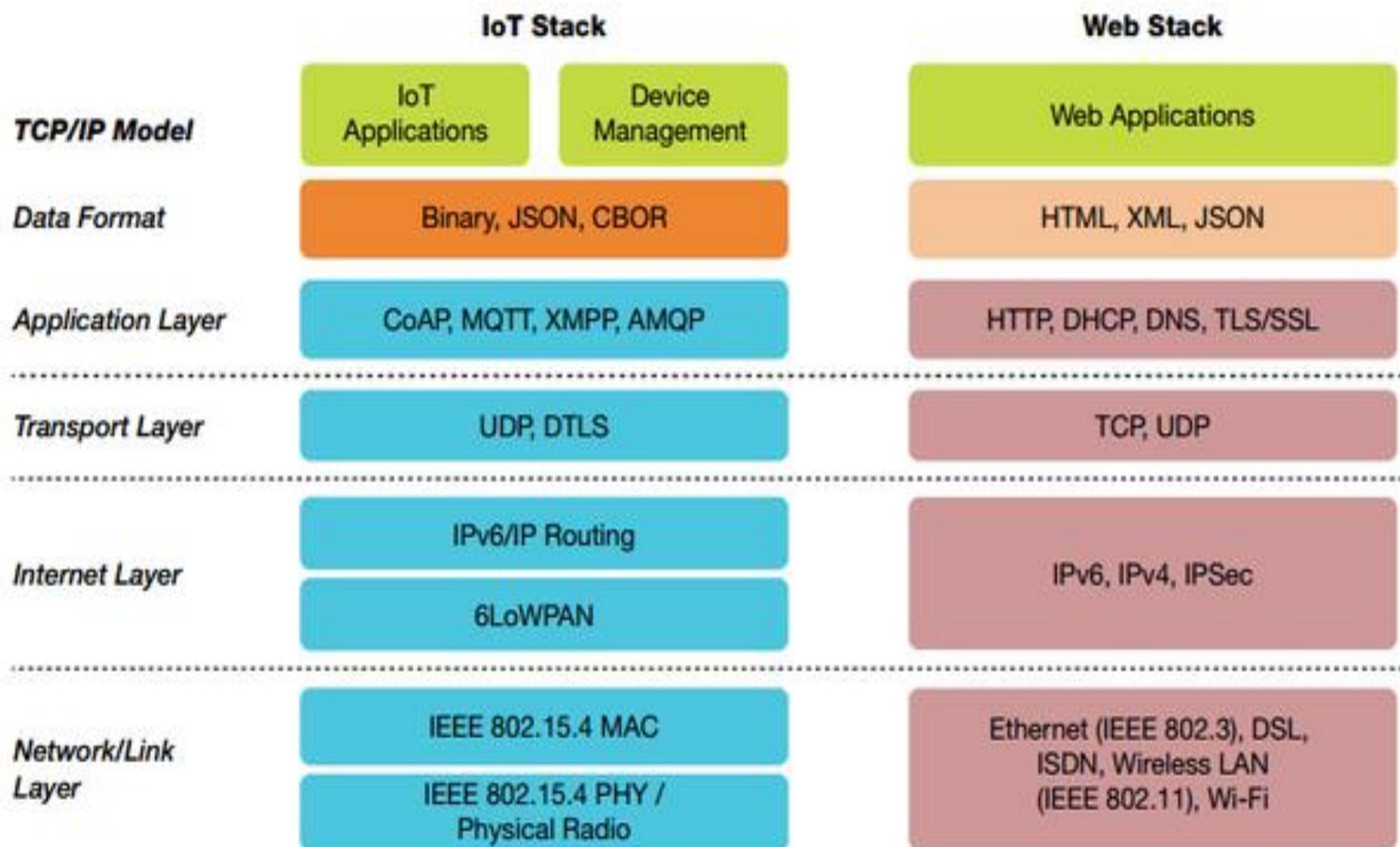
## TCP/IP MODEL

Application Layer

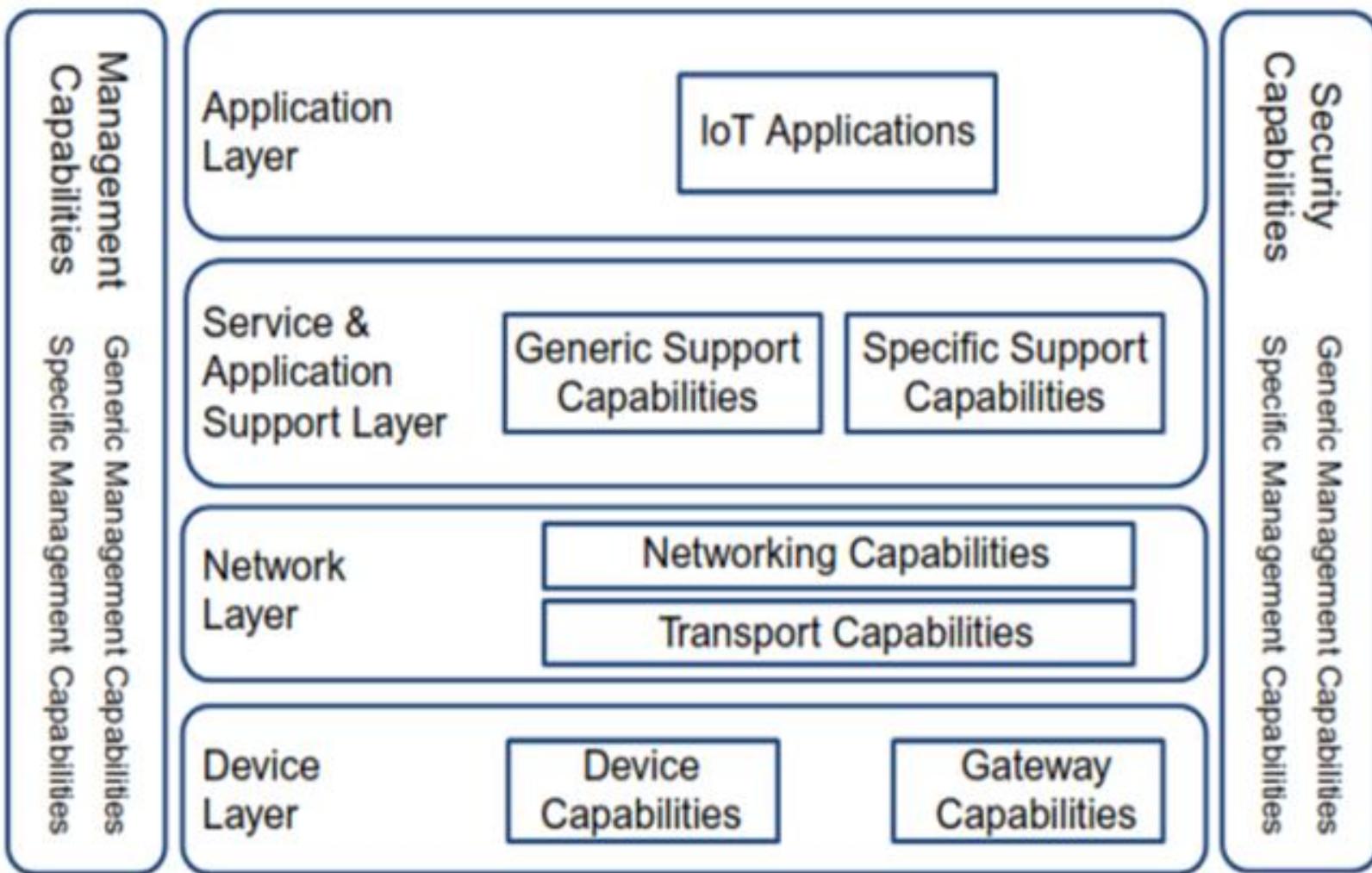
Transport Layer

Internet Layer

Network Access Layer



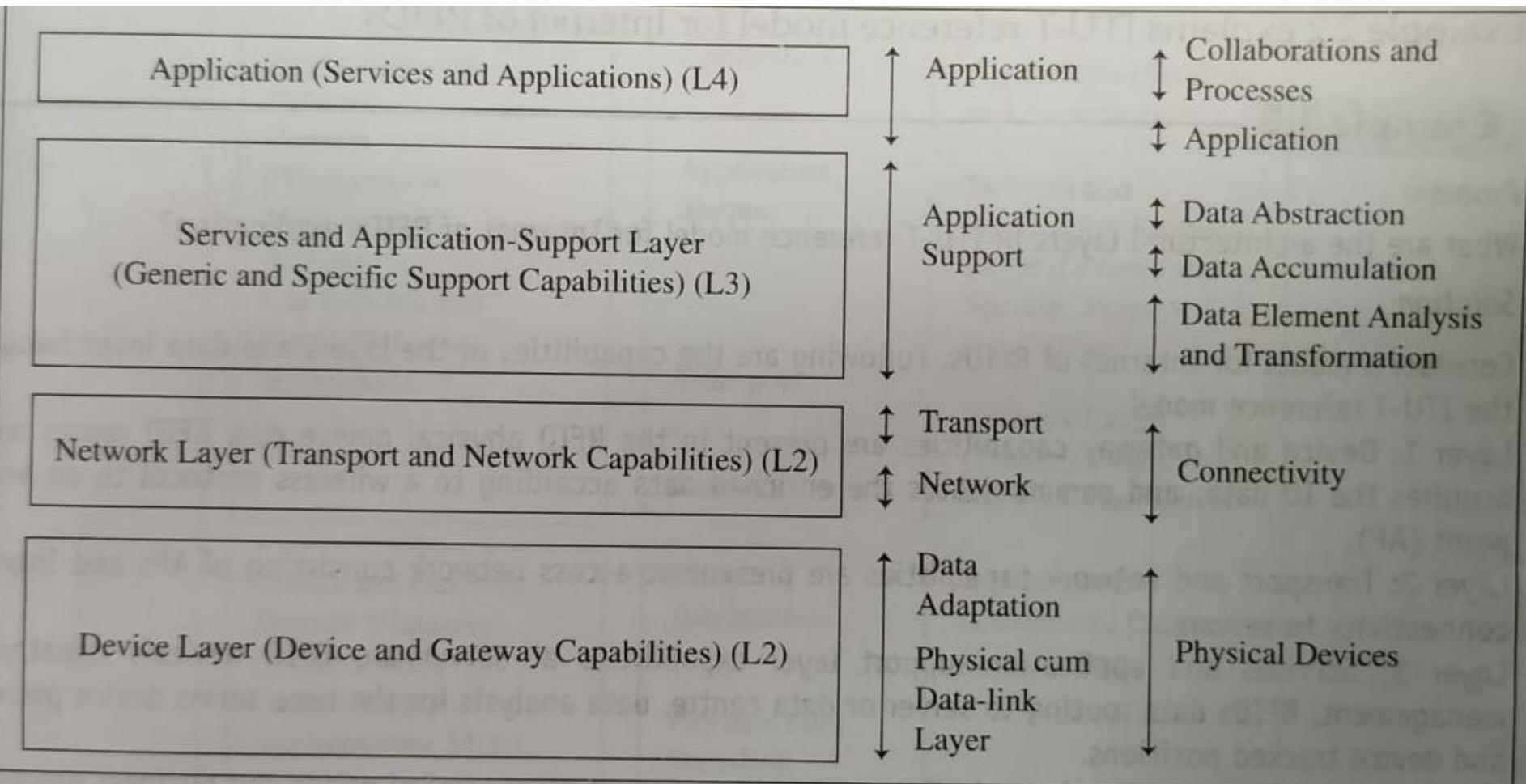
# ITU-T IoT Reference Model



ICMP stands for

- A. Internet Connect Message Protocol
- B. Internet Control Message Protocol
- C. International Connect Message Protocol
- D. International Control Message Protocol

# Comparison



# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# Mobile Adhoc Networks

COCSC20

# Wi-Fi

- Wi-Fi:
  - name is NOT an abbreviation
- **Wireless Local Area Network (WLAN)** technology.
- WLAN and Wi-Fi often used synonymous.
- Typically **in 2.4 and 5 GHz bands**.
- Based on **IEEE 802.11** family of standards.

# IEEE

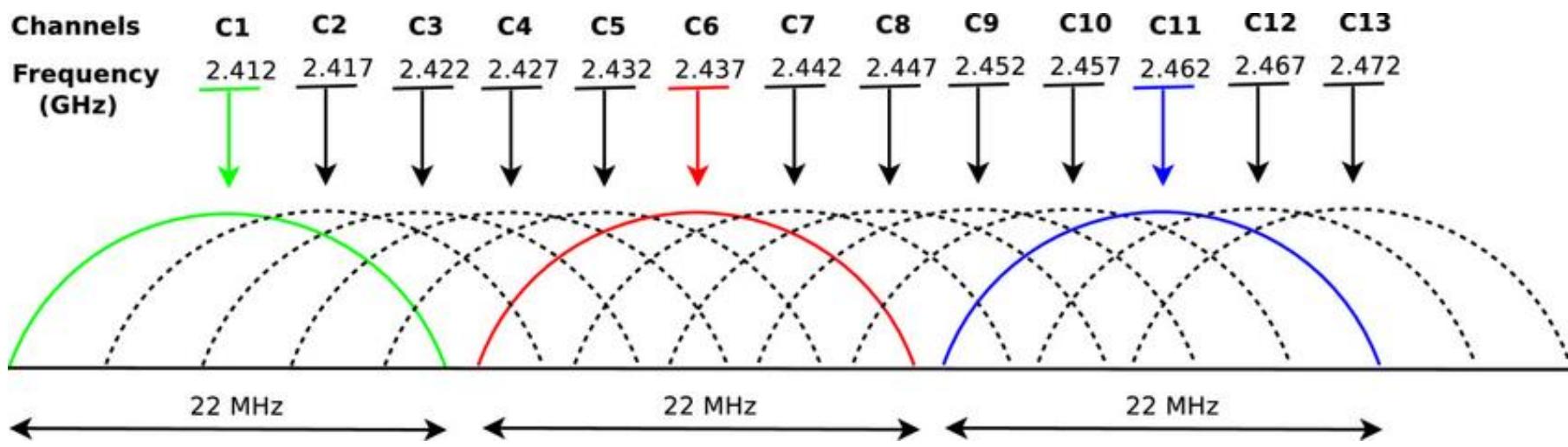
- IEEE (Institute of Electrical and Electronics Engineers) established the 802.11 Group in 1990. Specifications for standard ratified in 1997.
- Initial speeds were 1 and 2 Mbps.
- IEEE modified the standard in 1999 to include:
  - 802.11b
  - 802.11a
  - 802.11g
  - 802.11n
  - **802.11ac**

# WLAN (Wi-Fi)

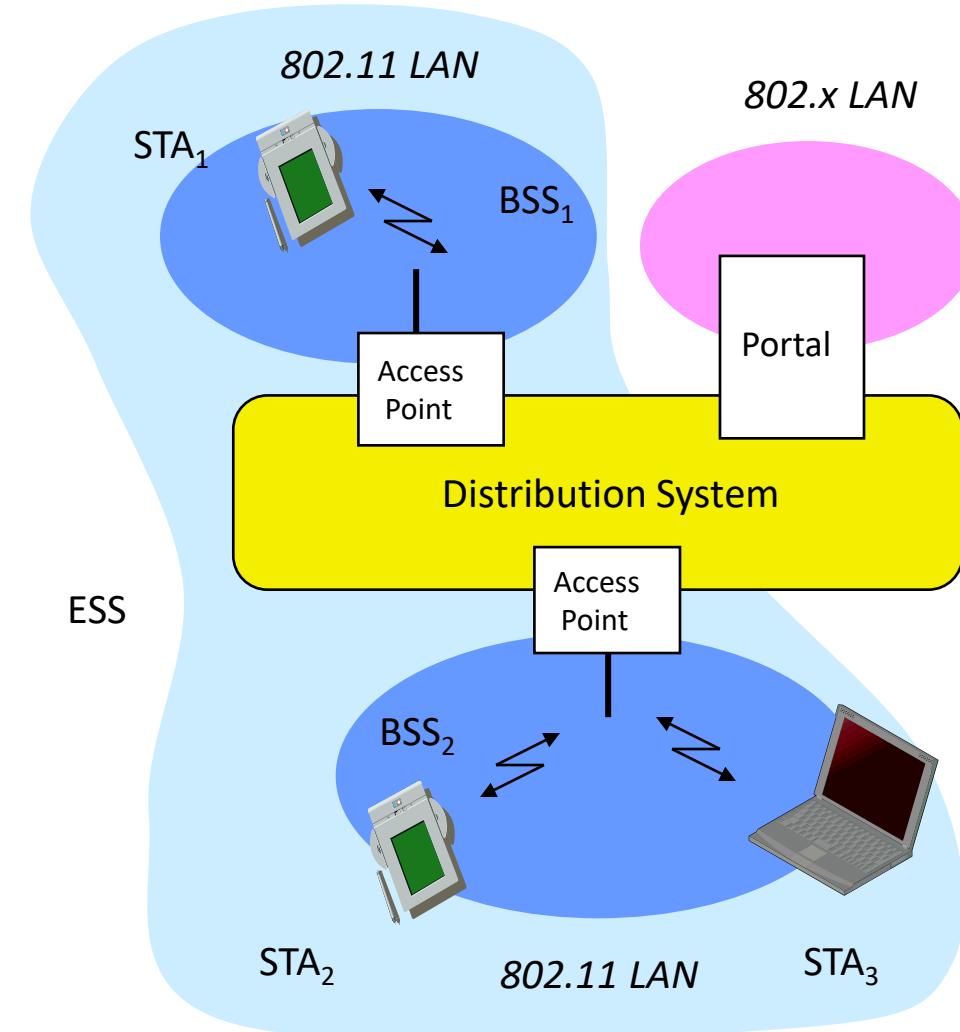
## 802.11 Wireless Standards

IEEE Standard	802.11a	802.11b	802.11g	802.11n	802.11ac
Year Adopted	1999	1999	2003	2009	2014
Frequency	5 GHz	2.4 GHz	2.4 GHz	2.4/5 GHz	5 GHz
Max. Data Rate	54 Mbps	11 Mbps	54 Mbps	600 Mbps	1 Gbps
Typical Range Indoors*	100 ft.	100 ft.	125 ft.	225 ft.	90 ft.
Typical Range Outdoors*	400 ft.	450 ft.	450 ft.	825 ft.	1,000 ft.

# Wi-Fi Channels

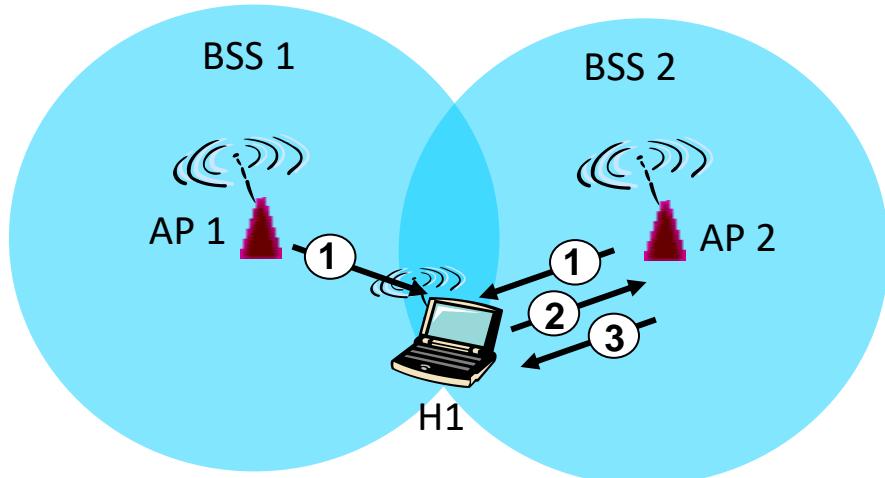


# 802.11 - Architecture of an Infrastructure Network



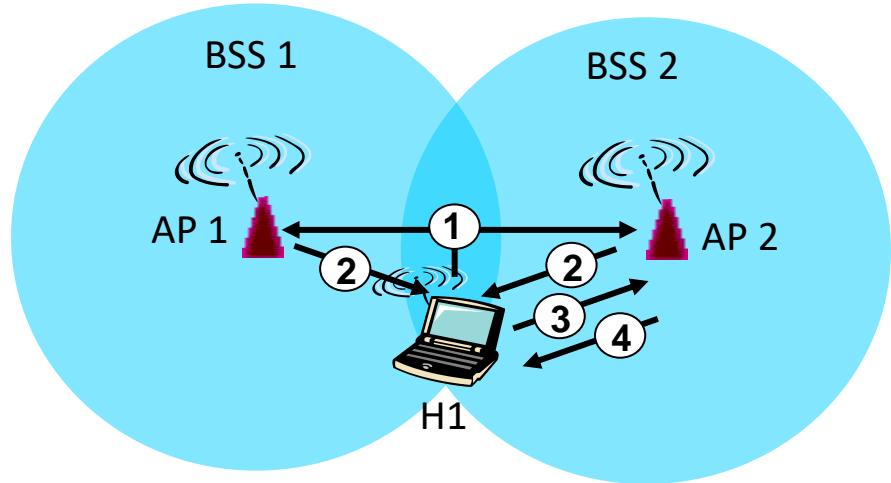
- **Station (STA)**
  - terminal with access mechanisms to the wireless medium and radio contact to the access point
- **Basic Service Set (BSS)**
  - group of stations using the same radio frequency
- **Access Point**
  - station integrated into the wireless LAN and the distribution system
- **Portal**
  - bridge to other (wired) networks
- **Distribution System**
  - interconnection network to form one logical network (ESS: Extended Service Set) based on several BSS

# Wi-Fi (802.11) Scanning



## Passive Scanning

- (1) Beacons sent from APs
- (2) Association Request sent from H1 to selected AP
- (3) Association Response sent from AP to H1



## Active Scanning

- (1) Probe Request (broadcast) sent from H1
- (2) Probe Response sent from APs
- (3) Association Request sent from H1 to selected AP
- (4) Association Response sent from AP to H1

# Wi-Fi Alliance Mission Statement

- Non-profit organization
- Certify the interoperability of products and services based on IEEE 802.11 technology
- Grow the global market for **Wi-Fi® CERTIFIED** products and services across all market segments, platforms, and applications
- Rigorous interoperability testing requirements

# Certificate & Logo

Wi-Fi® Interoperability Certificate

Certification ID: 24567832AP

**Wi-Fi CERTIFIED abgn**

This certificate represents the capabilities and features that have passed the interoperability testing governed by the Wi-Fi Alliance.

Detailed descriptions of these features can be found at [www.wi-fi.org/certificate](http://www.wi-fi.org/certificate)

**Certification Date:** February 14, 2004

**Category:** Access Point

**Company:** Name of Company

**Product:** Wireless LAN Access Point/Router Model#EX1010

**Model/SKU #:** EX1010

This product has passed Wi-Fi certification testing for the following standards:

IEEE Standard	Security	Quality of Service	Public Access
802.11a 802.11b 802.11g 802.11n	WPA - Personal WPA - Enterprise WPA2 - Personal (802.11i) WPA2 - Enterprise (802.11i)	WME (802.11e EDCA profile) WSM (802.11e HCCA profile)	
<b>Regulatory</b> 802.11d 802.11h	<b>Suplicant</b> EAP-TLS EAP-TTLS/MSCHAPv2 EAP-TTLS/PAP PEAPv0/EAP-MSCHAPv2 PEAPv1/EAP-GTC PEAPv1/EAP-MD5 EAP-SIM		
	<b>Authentication Server</b> EAP-TLS EAP-TTLS/MSCHAPv2 EAP-TTLS/PAP PEAPv0/EAP-MSCHAPv2 PEAPv1/EAP-GTC PEAPv1/EAP-MD5		

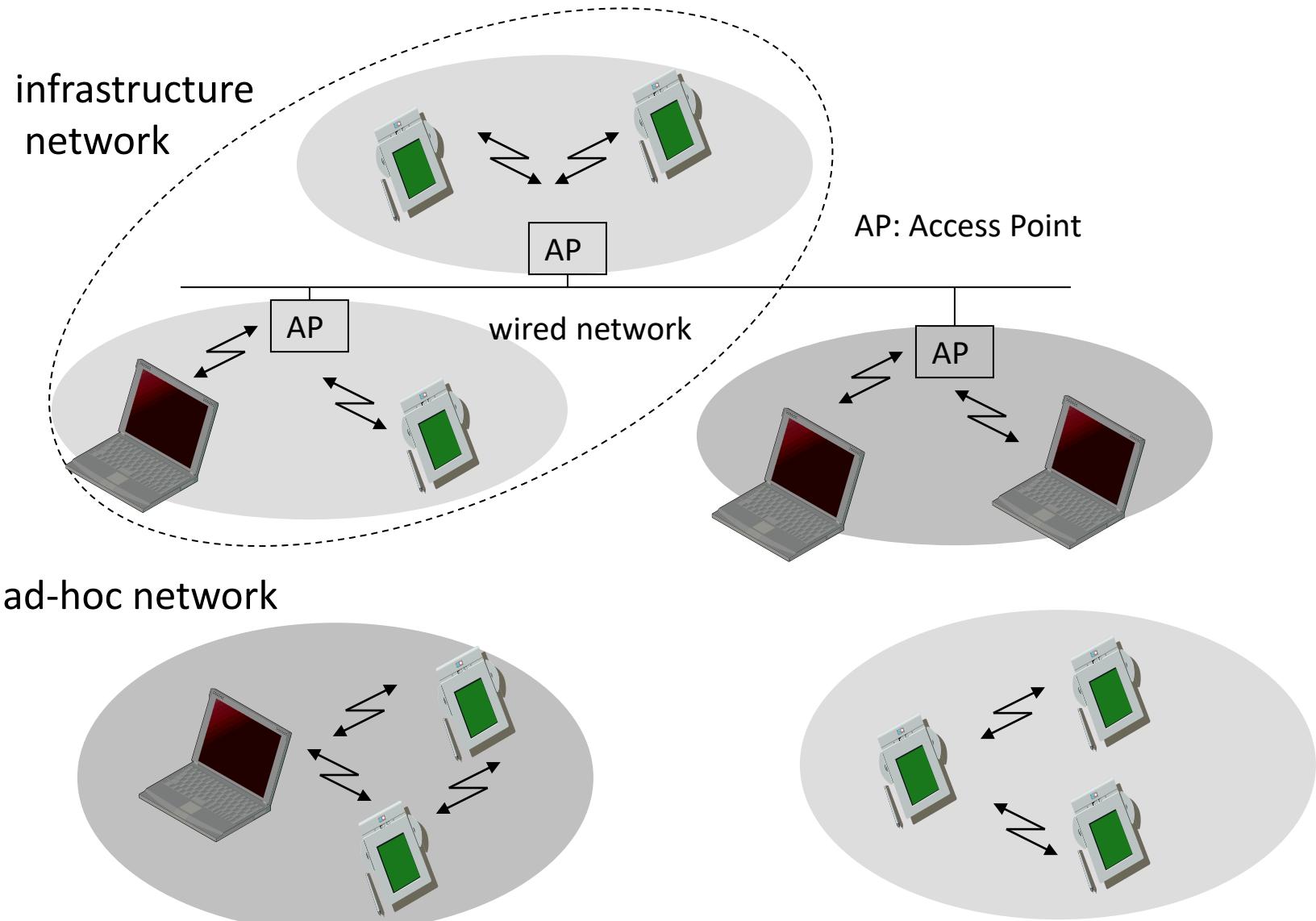
For more information: [www.wi-fi.org/certified\\_products](http://www.wi-fi.org/certified_products)

Certificate inside packaging (optional)



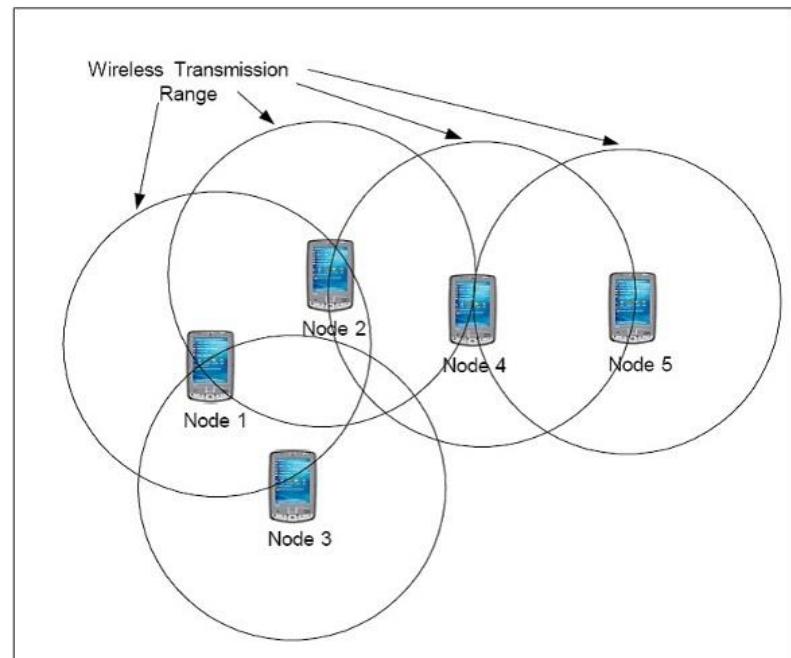
- Logo on product packaging (mandatory)
- Helps retailers and consumers

# Infrastructure vs. Ad-Hoc Networks



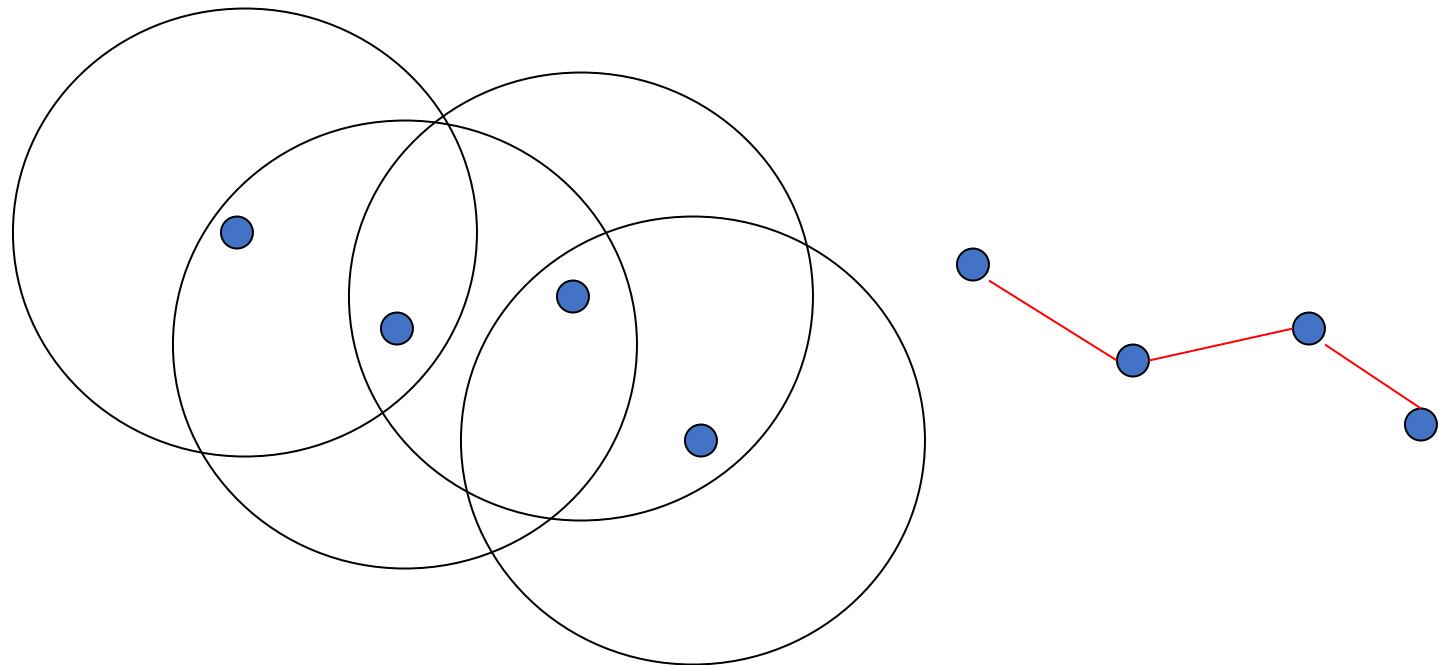
# Infrastructure-Less (Ad-Hoc)

- Ad-hoc means '*for this purpose*'
- No need for infrastructure (like routers, cell towers, etc.)
- MANET: **Mobile Ad-Hoc Network**



# Routing

- Packets may need to traverse multiple links to reach destination
- Mobility causes route changes

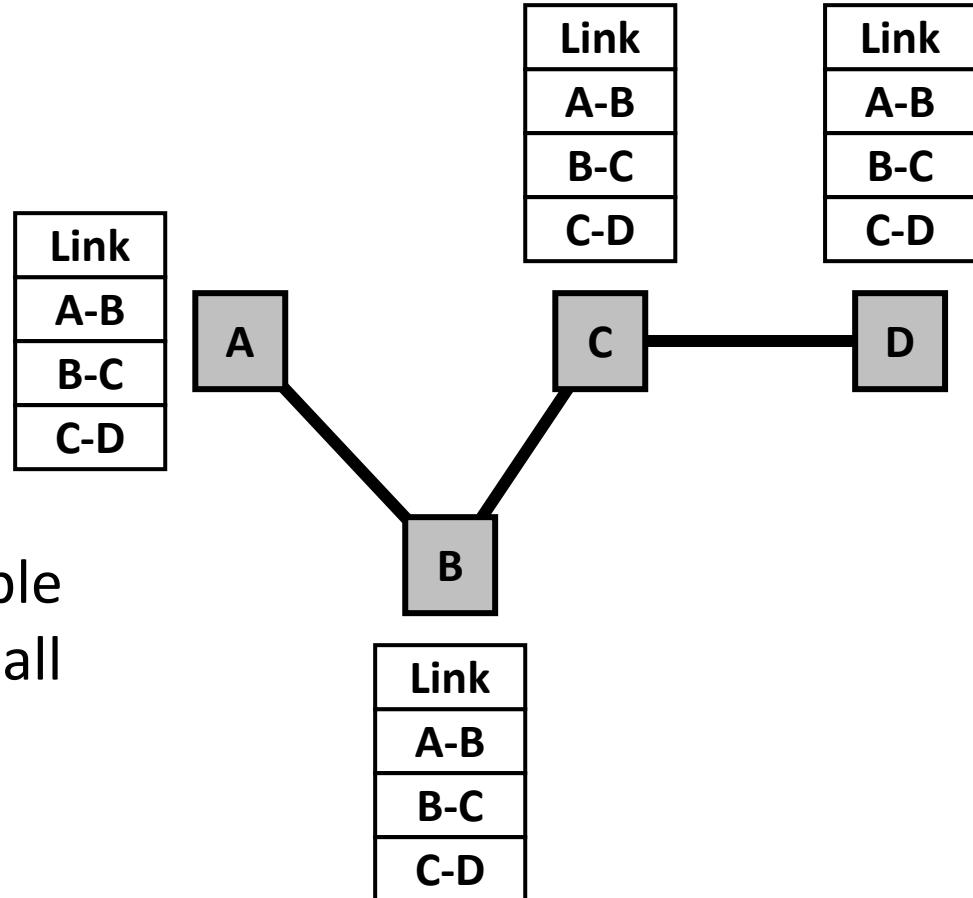


# Ad-Hoc Routing Protocol

- An ad-hoc routing protocol is a convention that controls how nodes decide which way to route packets between computing devices in a mobile ad-hoc network
- Foundation in most protocols: **neighbor discovery**
  - Nodes send periodic announcements as broadcast packets (beacon messages, alive messages, ...)
  - Can embed “neighbor table” into such messages; allows nodes to learn “2-hop neighborhood”
- Popular types of routing protocols:
  - **Proactive**
  - **Reactive**
  - **Geographic**

# Proactive: “Link-State” Algorithms

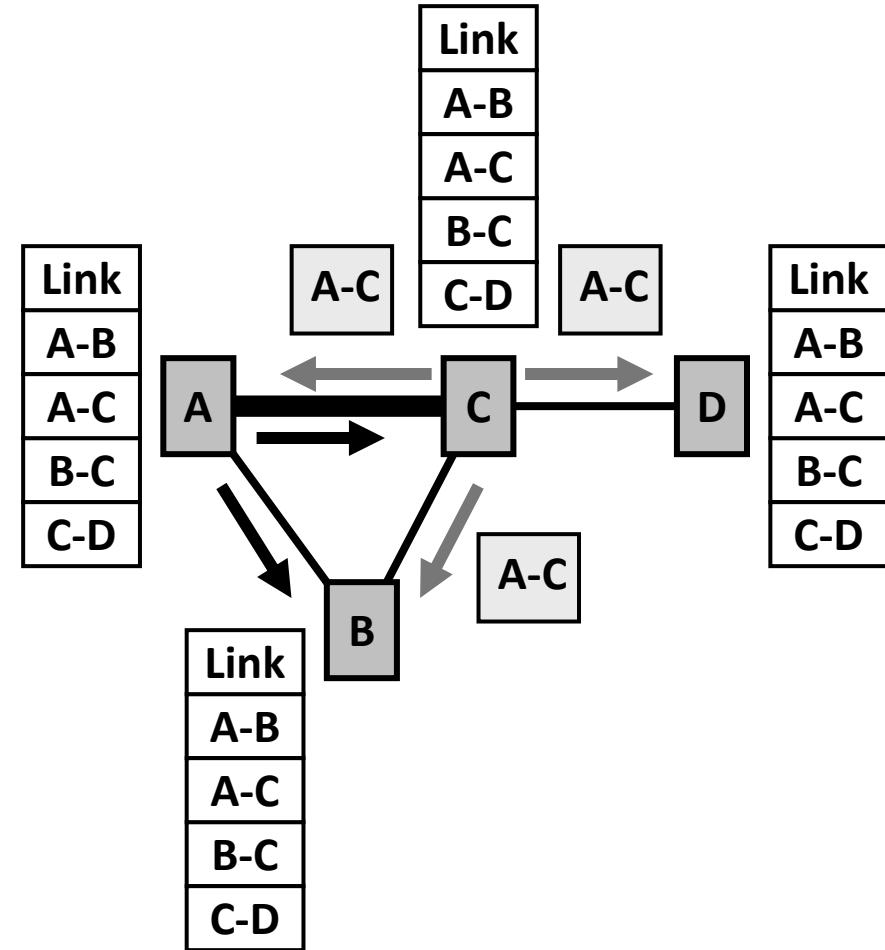
- Each node shares its link information so that all nodes can build a map of the full network topology



- Assuming the topology is stable for a sufficiently long period, all nodes will have the same topology information

# Proactive: “Link-State” Algorithms

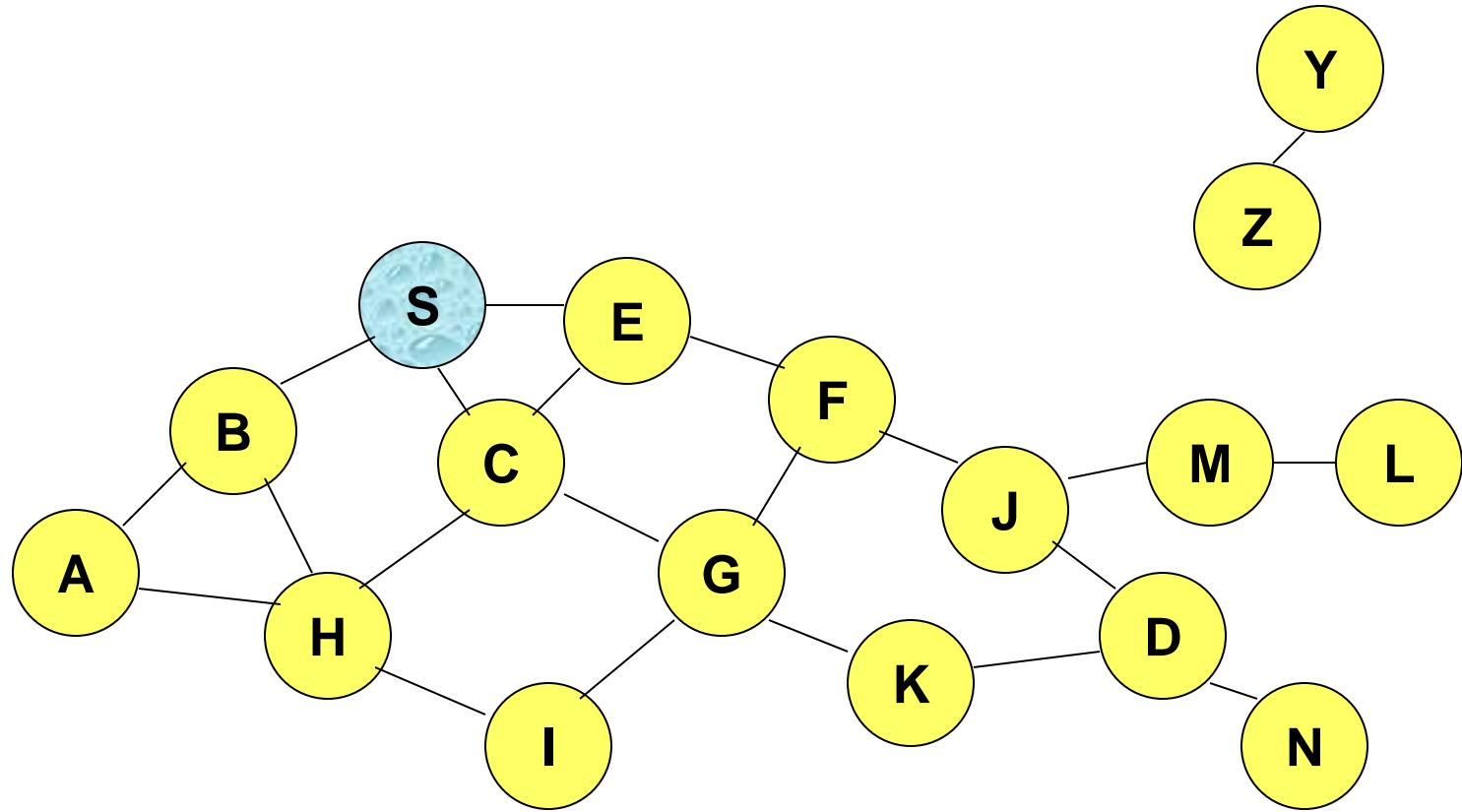
- Link information is updated when a link changes state (goes up or down)
  - by sending small “hello” packets to neighbors
- Nodes A and C propagate the existence of link A-C to their neighbors and, eventually, to the entire network



## Reactive: DSR

- **Dynamic Source Routing**
- Search for route when needed only
  - Search using **Route Request (RREQ)** broadcasts
  - Response using **Route Reply (RREP)** message
- Every message along route contains entire path to help intermediate nodes to decide what to do with message

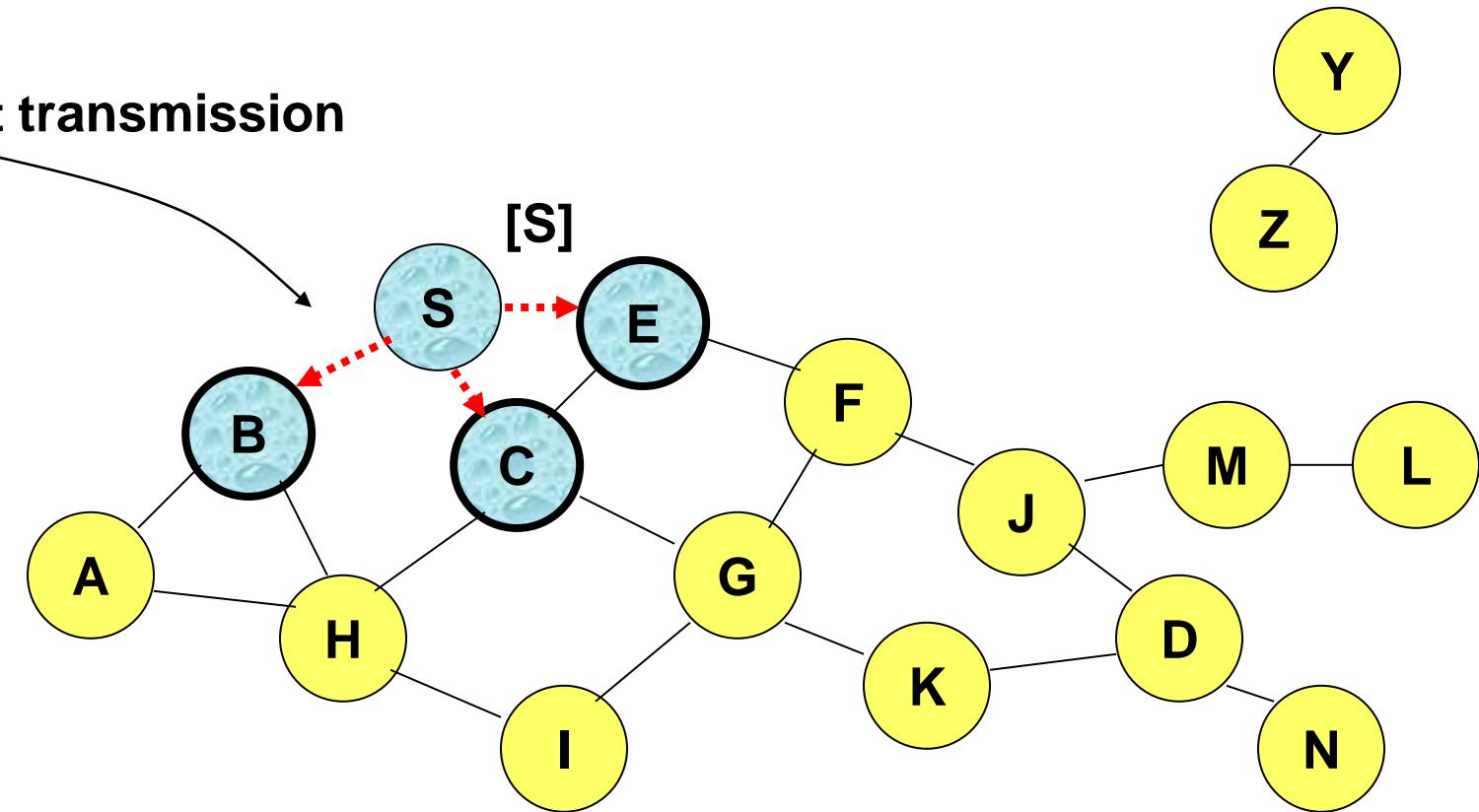
# Route Discovery in DSR



Represents a node that has received RREQ for D from S

# Route Discovery in DSR

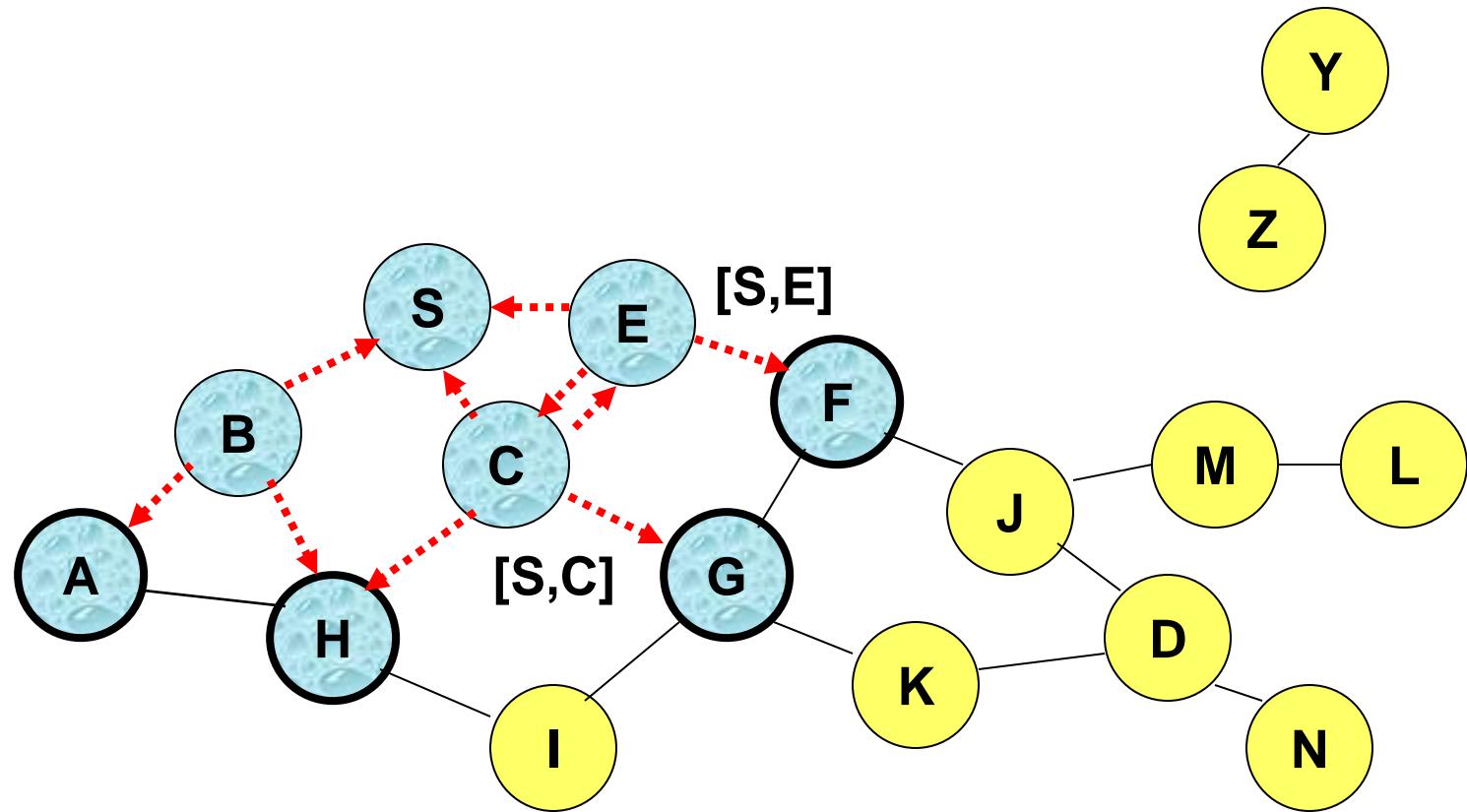
Broadcast transmission



-----> Represents transmission of RREQ

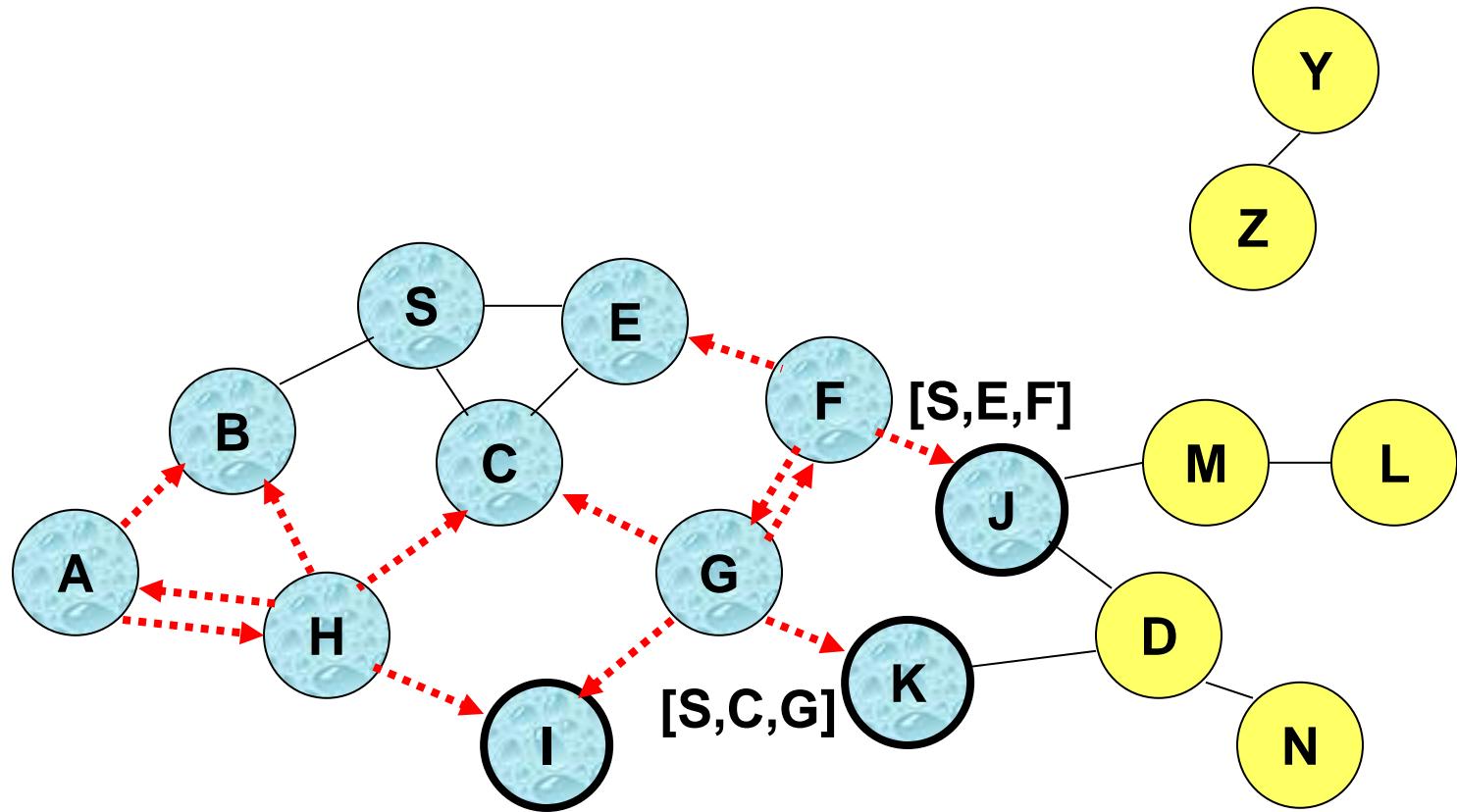
[X,Y] Represents list of identifiers appended to RREQ

# Route Discovery in DSR



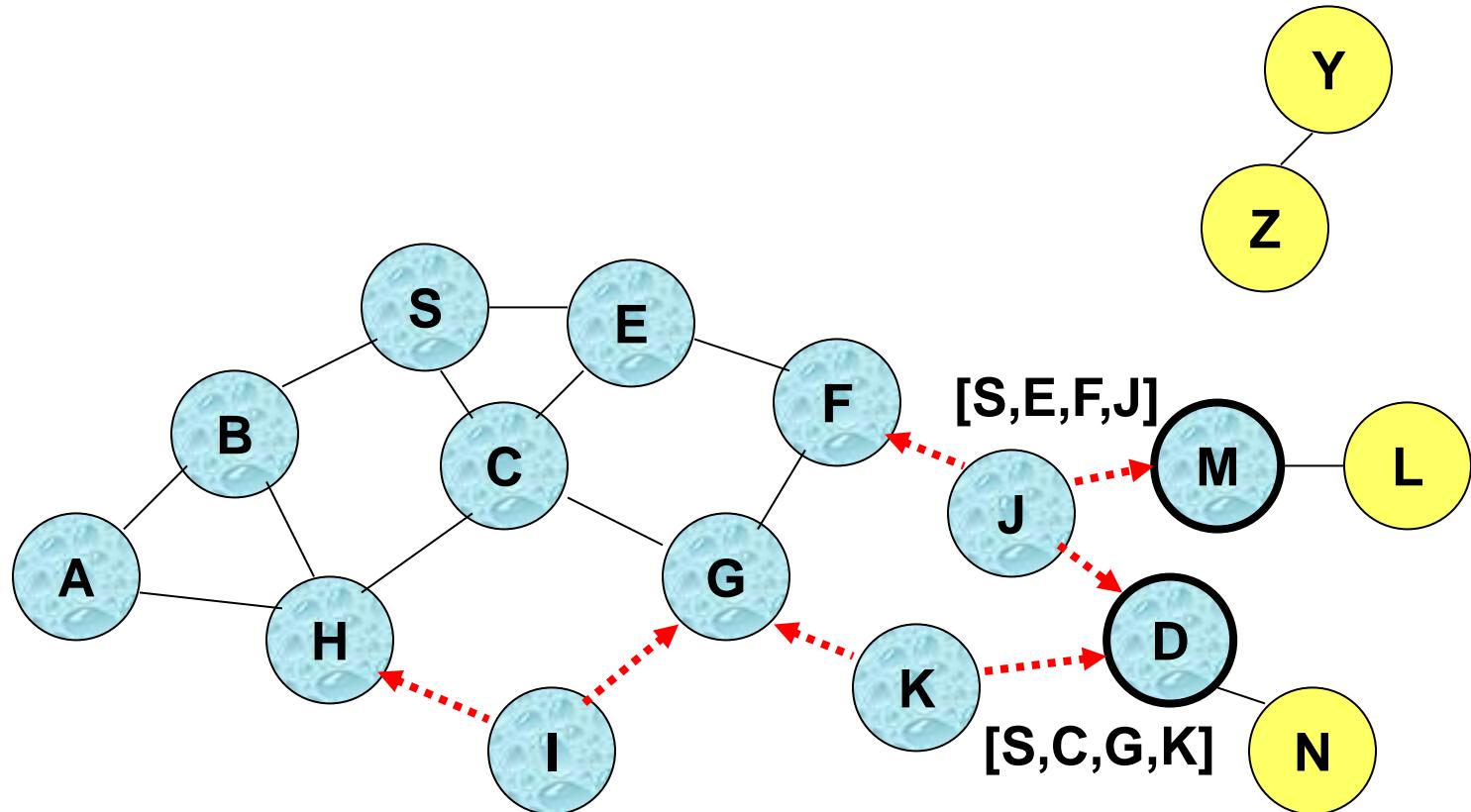
- Node H receives packet RREQ from two neighbors:  
**potential for collision**

# Route Discovery in DSR



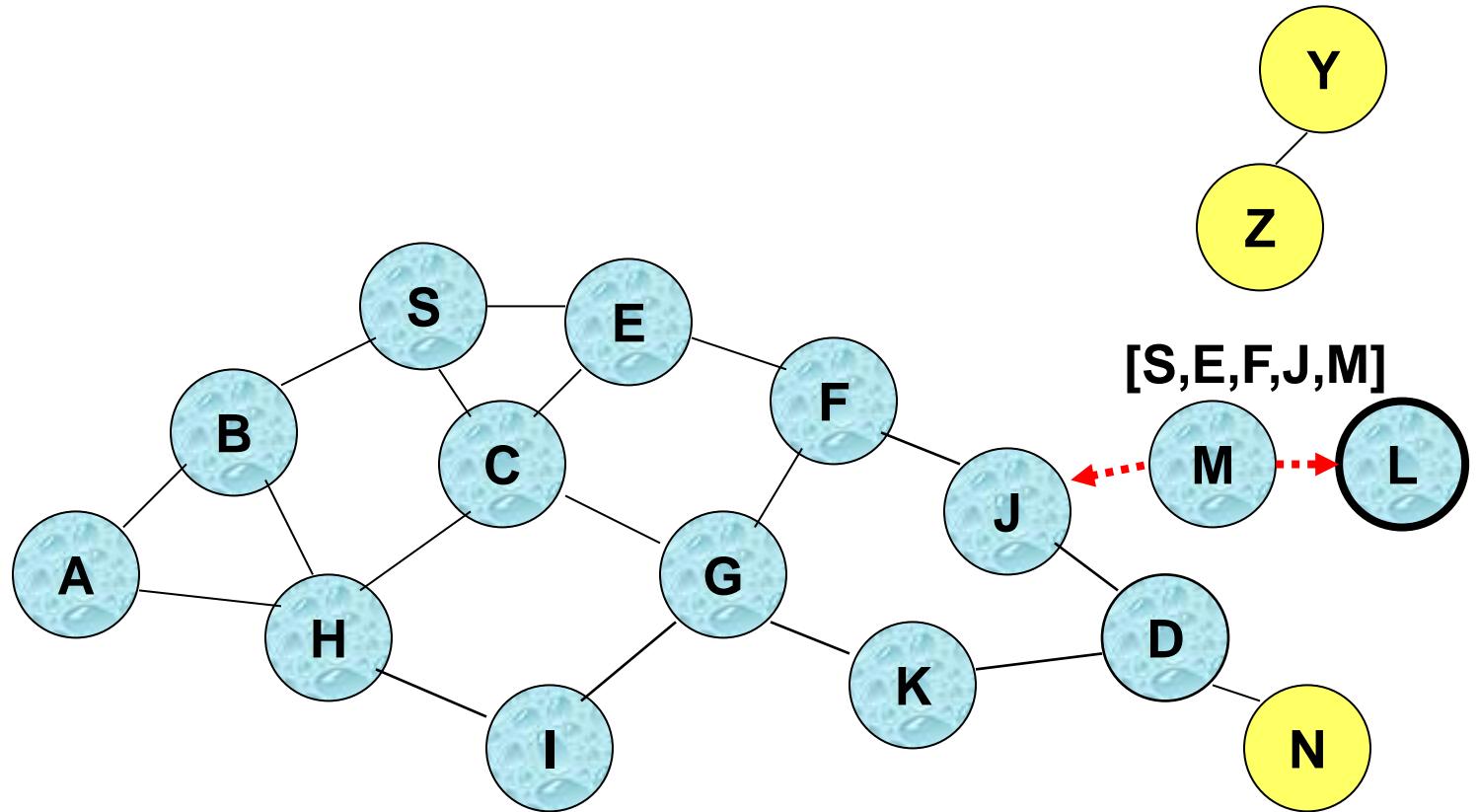
- Node C receives RREQ from G and H, but does not forward it again, because node C has **already forwarded RREQ once**

# Route Discovery in DSR



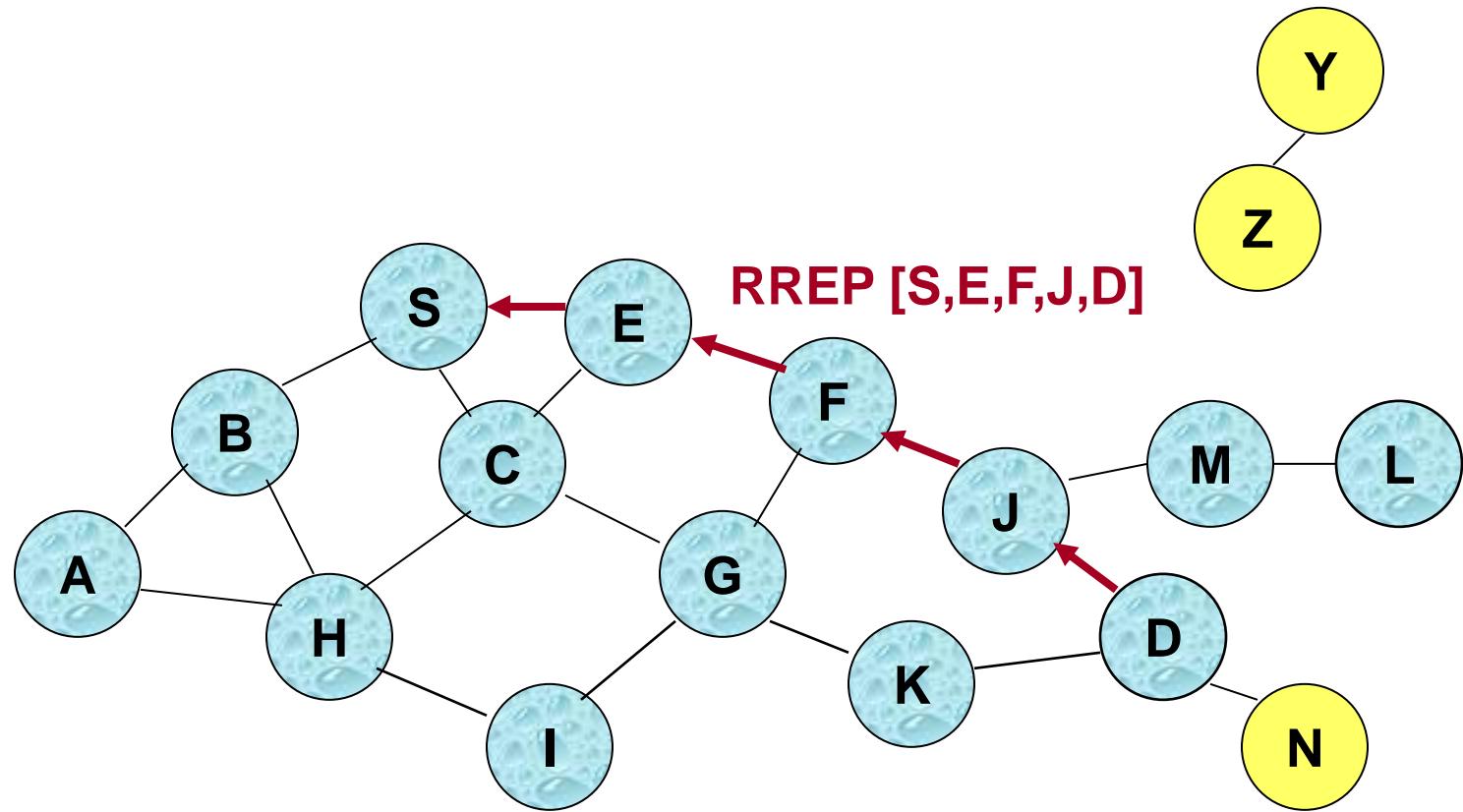
- Nodes J and K both broadcast RREQ to node D
- Since nodes J and K are **hidden** from each other, their **transmissions may collide**

# Route Discovery in DSR

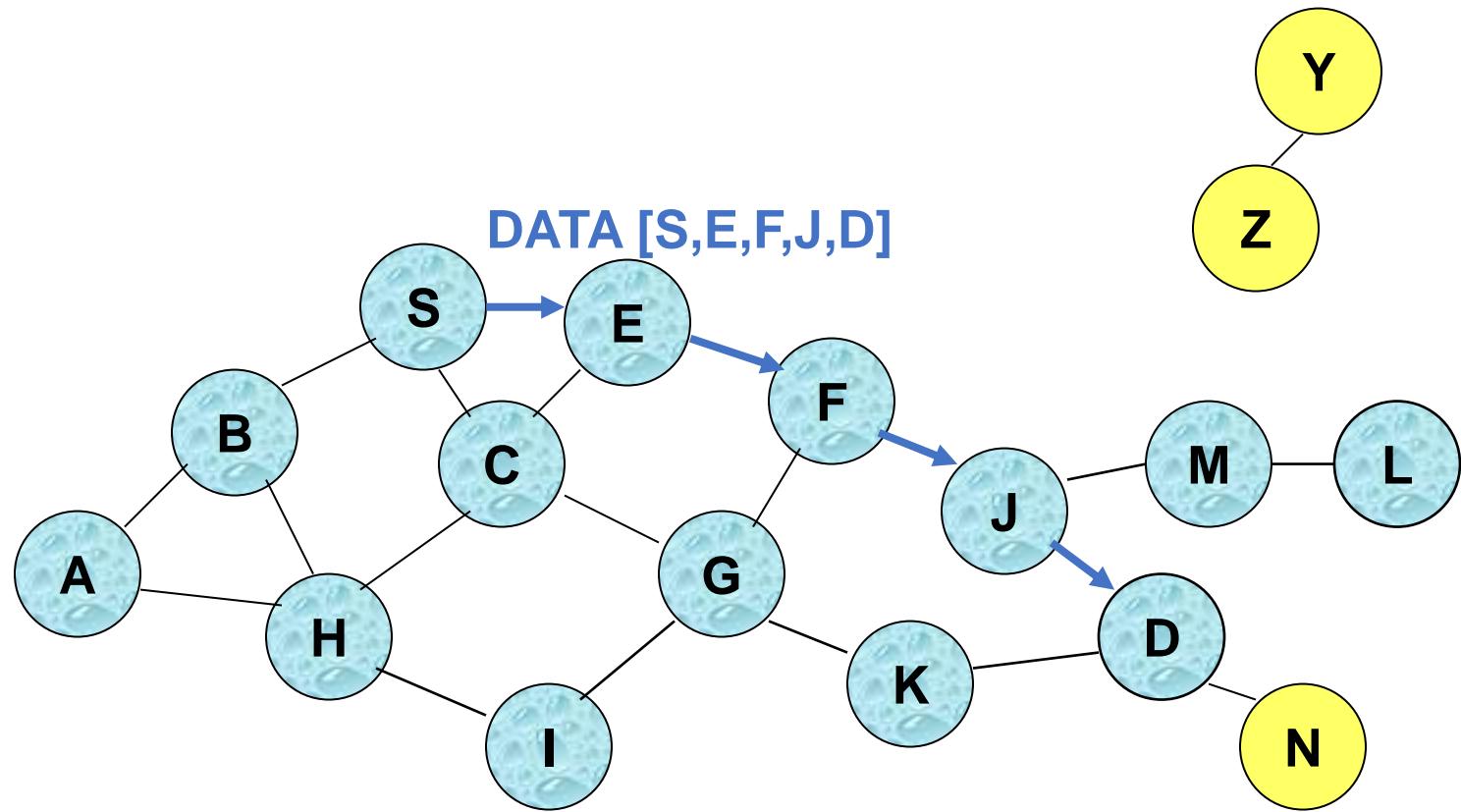


- Node D **does not forward RREQ**, because node D is the **intended target** of the route discovery

# Route Reply in DSR



# Data Delivery in DSR



**Packet header size grows with route length**

# Proactive vs Reactive

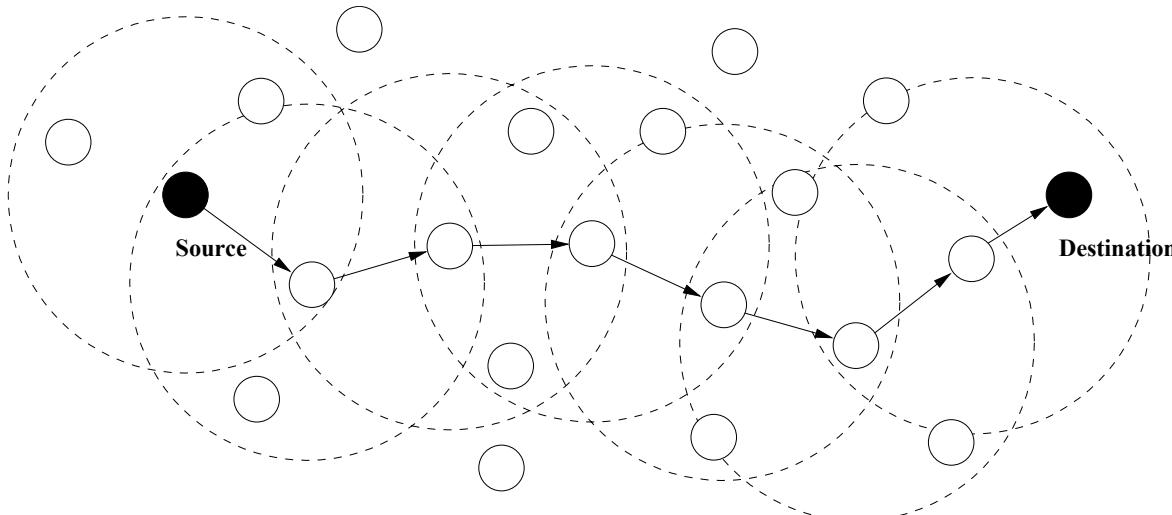
- Reactive:
  - Only establish/maintain routes between nodes needed them (in contrast: tables store ALL routes)
  - Store entire route in each message; message size grows with route length
  - Route requests cause “flooding”
- Proactive:
  - Route information always available; no need to search for route (but route information can be outdated)
  - Continuous exchange of route change updates

# Geographic Routing

- Nodes use location information to make routing decisions
  - sender must know the locations of itself, the destination, and its neighbors
  - location information can be queried or obtained from a **location broker**
  - location information can come from GPS (Global Positioning System) or some other form of positioning technology.

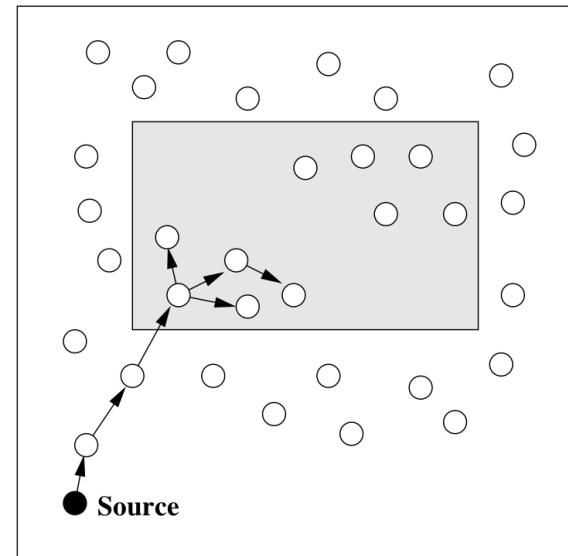
# Unicast Location-Based Routing

- One single destination
- Each forwarding node makes localized decision based on the location of the destination and the node's neighbors (**greedy forwarding**)
- Challenge: packet may arrive at a node without neighbors that could bring packet closer to the destination (**voids or holes**)



# Geocasting

- Packet is sent to all or some nodes within specific geographic region
- Example: query sent to all sensors within geographic area of interest
- Routing challenge:
  - propagate a packet near the target region (similar to unicast routing)
  - distribute packet within the target region (similar to flooding)



# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# **Internet of Things**

## **COCSC20**

**Lecture 1: Introduction to Course Structure**

**Semester 6**

# Course Website

- Google Classroom:

[https://classroom.google.com/c/NTgwOTkyNjUzNTA4?cjc=w  
epfkgw](https://classroom.google.com/c/NTgwOTkyNjUzNTA4?cjc=wepfkgw)

- Class Joining Code: wepfkgw
- We will continuously putting reading material, lectures and labs.

# Prerequisite

- Interested in the learning of Course.
- Interested in Programming
  - Python and C programming preferred
- Basic Hardware Knowledge.
- Fascinated to solve real life problem using IoT skills.
- Curious to build new/reliable solution.

# Evaluation Components

<b>Components of Course Evaluation</b>	<b>Percentage</b>
Mid Term Examination	25
Continuous Evaluation	25
End Term Examination	50
Project	
Research Paper	

# Course Outcomes

- To understand the concepts of IoT and related protocol.
- To learn to develop the sensor networks for collecting the data.
- To create IoT solutions using sensors, actuators, and Devices.
- To understand the upcoming advancement in the domain of IoT.
- To deploy an IoT solution for the nearby society for improving their experiences.

# Major Topics

- IoT: Applications, Requirements, Architectural View, Examples, case studies.
- Sensors and things
- Hardware Components: Arduino, NodeMCU, Raspberry Pi, Jetson Nano.
- IoT Network and Protocols: 6LoWPAN, HTTP, MQTT, CoAP.
- IoT Security, SDN, NFV
- Fog and Edge Computing, NodeRED

# Course Structures

- Lectures participation.
- Multiple Quiz during the lectures (Minimum 2)
- Mid Term Exam and End Term Exam
- Reading for Web source and Journals (Self-Learning)
- Hands-on Lab (Continuous Evaluation)
- Semester Project (Three evaluations: Beginning, Mid, Final)
- Expert Lectures
- Flipped Learning (Two Lectures)
- For doubt (drop an email or message)

# Semester Project

Will provide the set of questions for the project idea in a week.

- Team Name
- Team Members (3)
- Project Idea
- Project Novelty
- Society Impact
- Hardware Requirements
- Possible end-Customers

# Desired Outcome

- A IoT based Project.
- A video for explaining the project (5 Minutes)
- A novel research paper/report/patent from the project work.

# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[gaurav.singal@nsut.ac.in](mailto:gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

WhatsApp Group link: We will create it.

# Internet-of-Things (IoT)

Introduction



What is the Internet-of-Things?

# How Does My Fridge Do That?

- You are leaving the home (sense user)
- There's no milk in fridge (sense object)
- Use this information to make a decision (process)
- Inform user of decision (communicate)

You are leaving the home (sense user)

There's no milk in fridge (sense object)

- What type of sensor?
- Is milk needed?
- No milk or “little” milk? (prediction)

Use this information to make a decision (process)

Inform user of decision (notify)

# How Does My Fridge Do That?

You are leaving the home (sense user)

- What type of sensor?
- Distinguish between parent and child
- Identify reason for leaving home
- Identify other contexts (e.g., store hours)

There's no milk in fridge (sense object)

Use this information to make a decision (process)

Inform user of decision (notify)

# How Does My Fridge Do That?

You are leaving the home (sense user)

There's no milk in fridge (sense object)

**Use this information to make a decision (process)**

- Where is processor?
- What are the rules?
- Fixed rules versus dynamic rules (learning)

Inform user of decision (notify)

# How Does My Fridge Do That?

You are leaving the home (sense user)

There's no milk in fridge (sense object)

Use this information to make a decision (process)

### **Inform user of decision (notify)**

- How?
- When?
- Privacy?
- Subtleness?
- Information overflow?

# How Does My Fridge Do That?

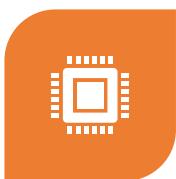
# Internet-of-Things (IoT)

Physical object (“thing”)  
+  
Controller (“brain”)  
+  
Sensors  
+  
Actuators  
+  
Networks (Internet)



# Related Areas/Terminology

---



**EMBEDDED SYSTEMS:**  
NOT NECESSARILY  
CONNECTED



**SENSOR NETWORKS:**  
COLLECTION OF  
SENSOR DEVICES  
CONNECTED THROUGH  
WIRELESS CHANNELS



**CYBER-PHYSICAL  
SYSTEMS:** FOCUS ON  
INTERACTION  
BETWEEN PHYSICAL  
AND CYBER SYSTEMS



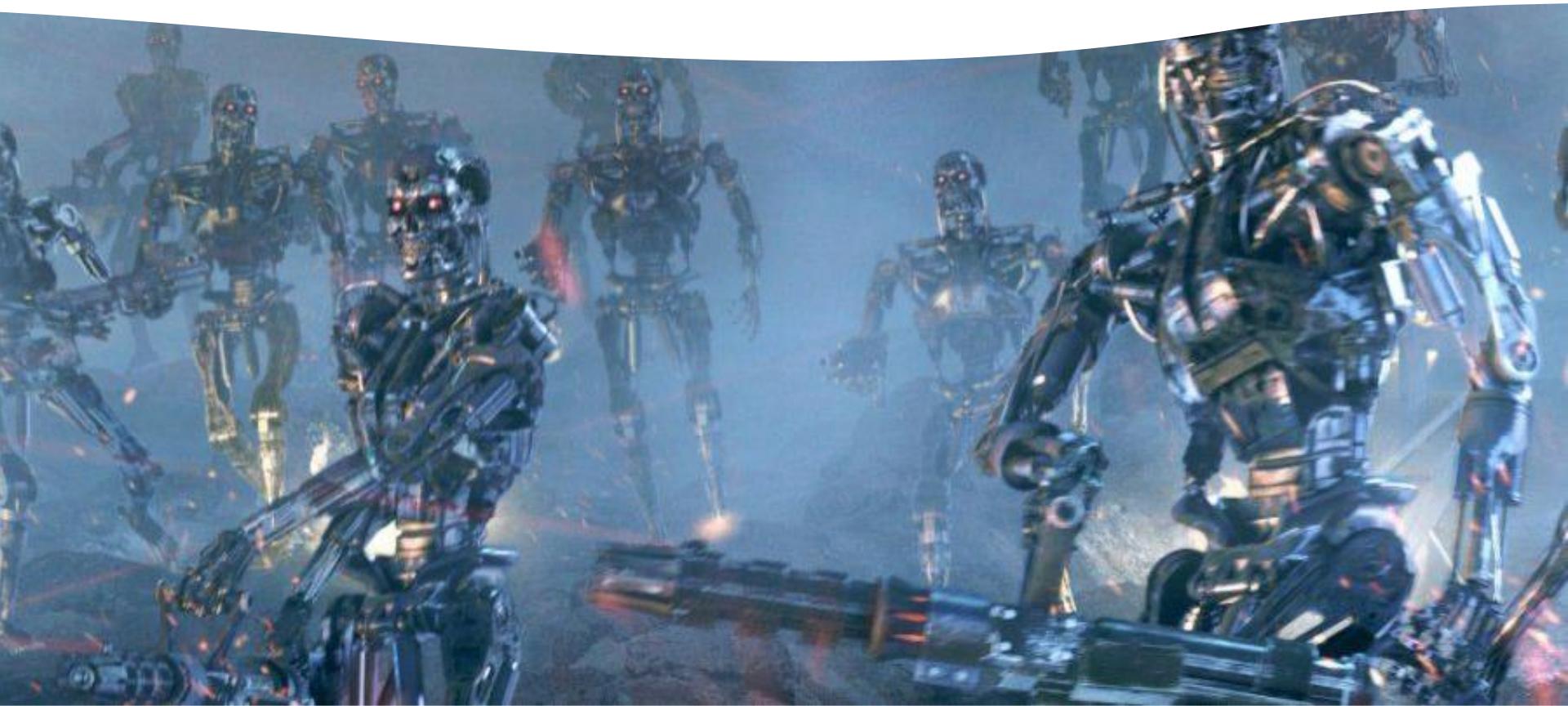
**REAL-TIME SYSTEMS:**  
FOCUS ON TIME  
CONSTRAINTS



**PERVASIVE/UBIQUITOUS COMPUTING:**  
FOCUS ON  
ANYTIME/ANYWHERE  
COMPUTING

## Related Areas

- Machine-to-machine (M2M) communications
- Internet of Everything (Cisco Systems)
- “Skynet” (Terminator movie)



# “Internet-of- Things”

---

Term coined by British entrepreneur Kevin Ashton, while working at MIT Auto-ID Labs

---

Referred to (and envisioning) a future global network of objects connected specifically by RFID (radio-frequency identification)

---

Complete automation of data collection

---

First article about IoT in 2004 from MIT; called it ‘Internet 0’.

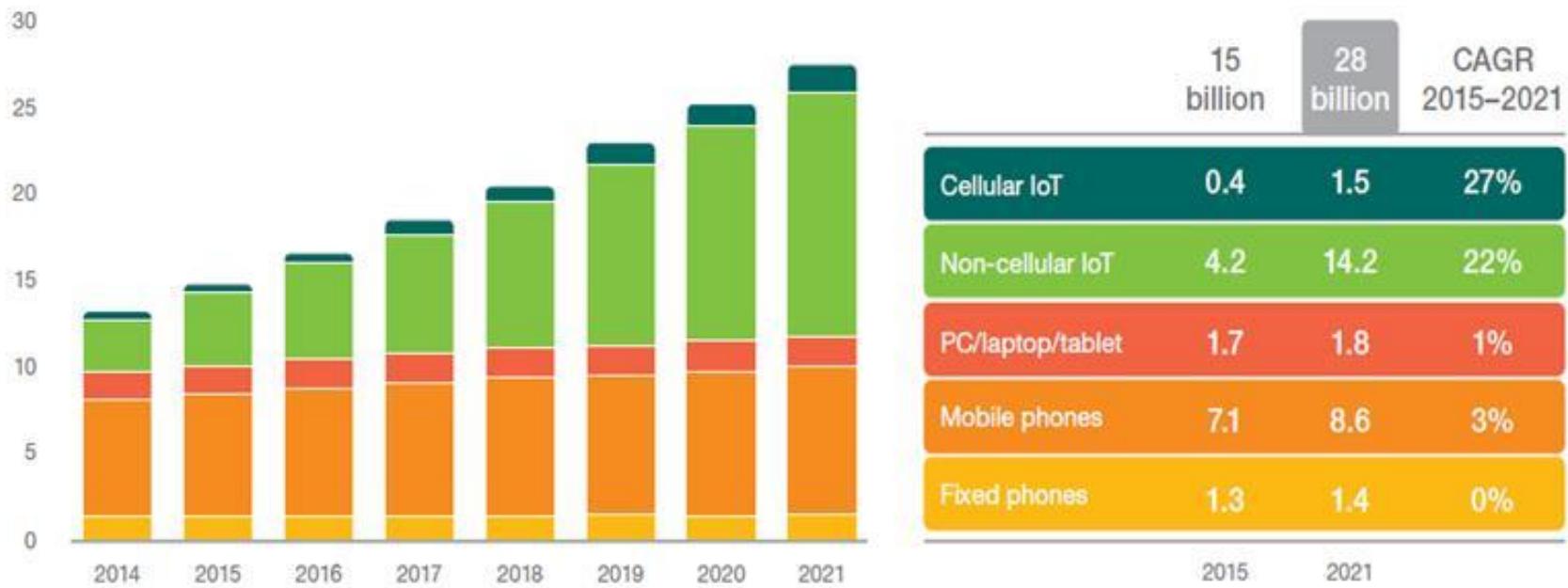
---

\*[https://en.wikipedia.org/wiki/Internet\\_0](https://en.wikipedia.org/wiki/Internet_0)

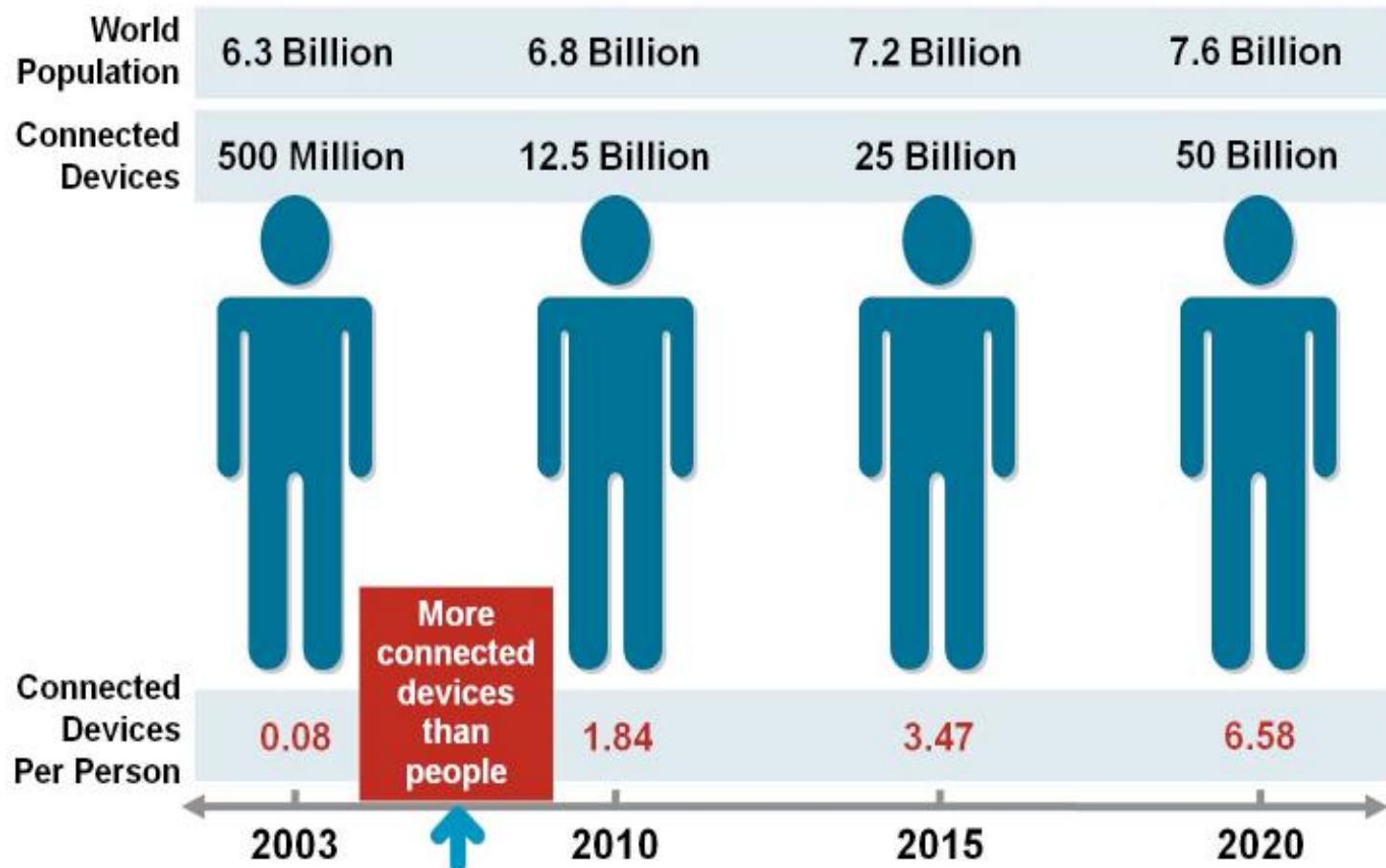
## Internet-of-Things Vision & Growth

# THE INTERNET OF THINGS

Connected devices (billions)



# Internet-of-Things Vision & Growth



Source: Cisco IBSG, April 2011

# What is IoT

- Internet of things (IoT) is the network of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these things to connect, collect and exchange data<sup>1</sup>.
- IoT refer to the connection of devices to the Internet.

<sup>1</sup>[https://en.wikipedia.org/wiki/Internet\\_of\\_things](https://en.wikipedia.org/wiki/Internet_of_things)

# Internet-of-Things (IoT)

Introduction

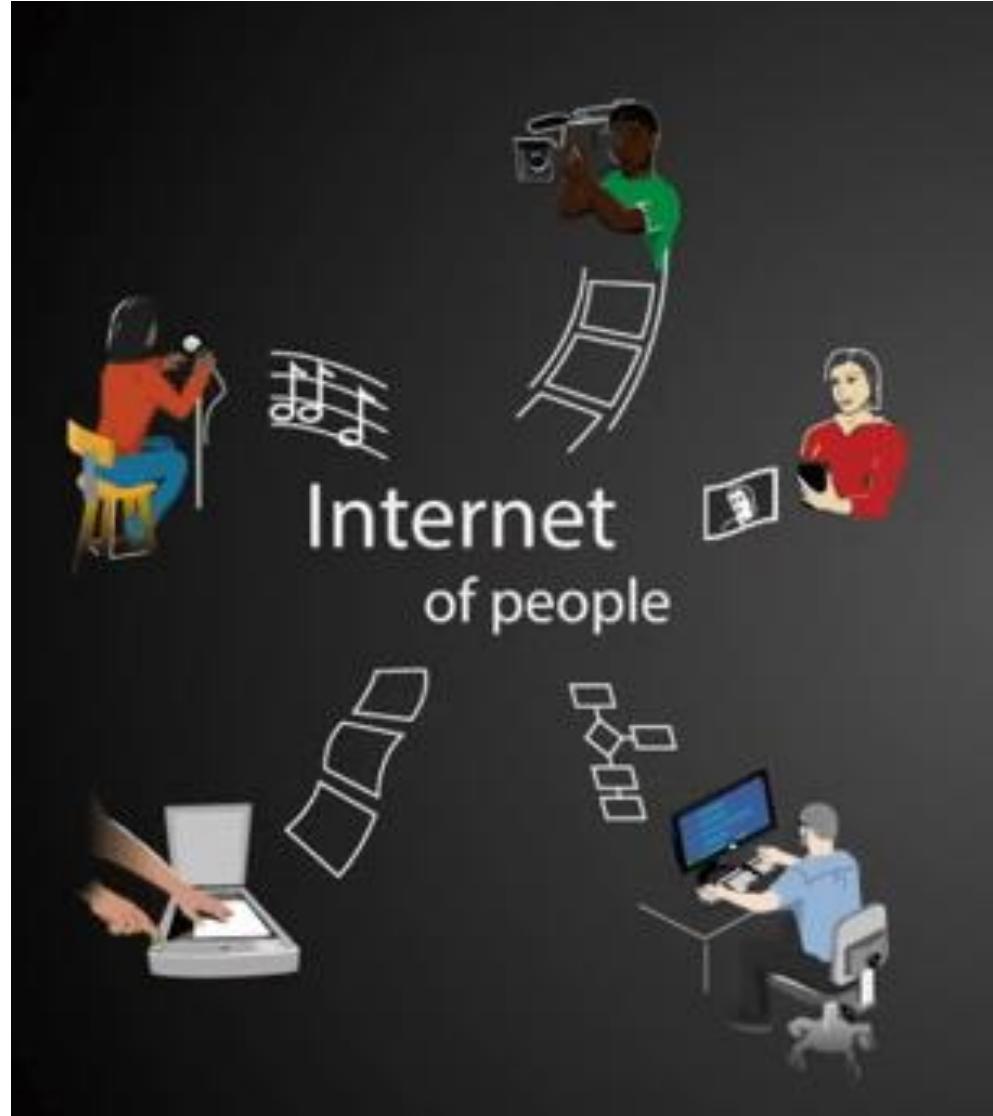
# What is IoT

- Internet of things (IoT) is the network of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these things to connect, collect and exchange data<sup>1</sup>.
- IoT refer to the connection of devices to the Internet.

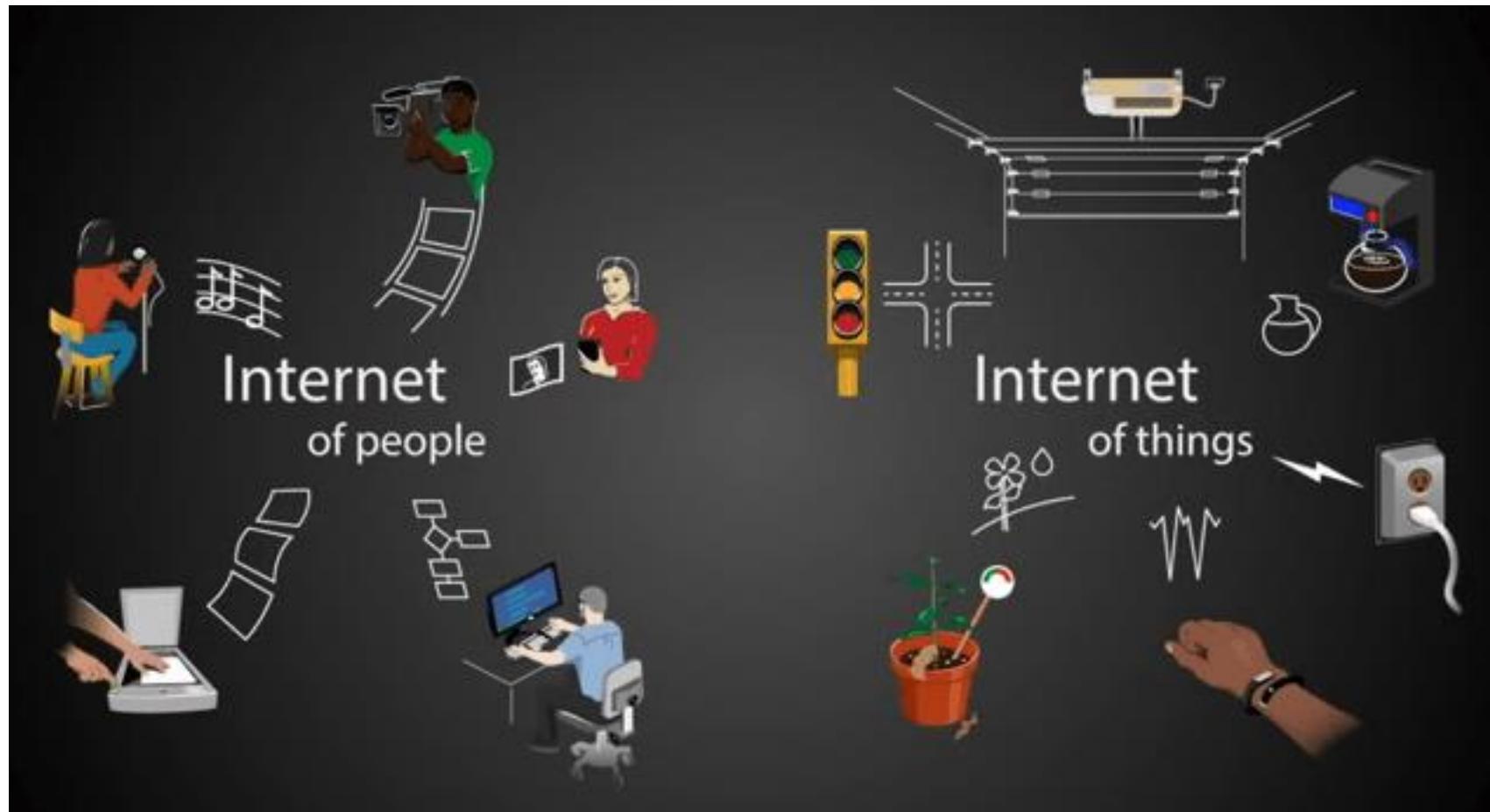
<sup>1</sup>[https://en.wikipedia.org/wiki/Internet\\_of\\_things](https://en.wikipedia.org/wiki/Internet_of_things)

## Internet of People (IOP)

- People are connected with the Internet.
- Internet is everywhere in the World.
- It is the primary connection between people.

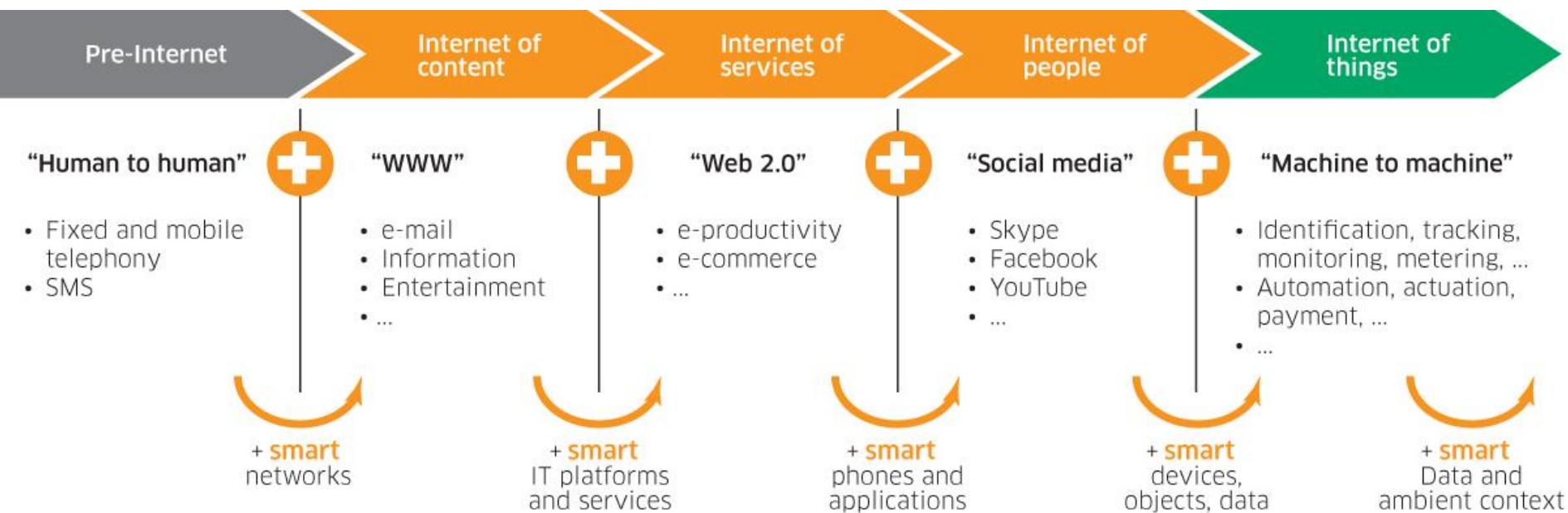


IOP

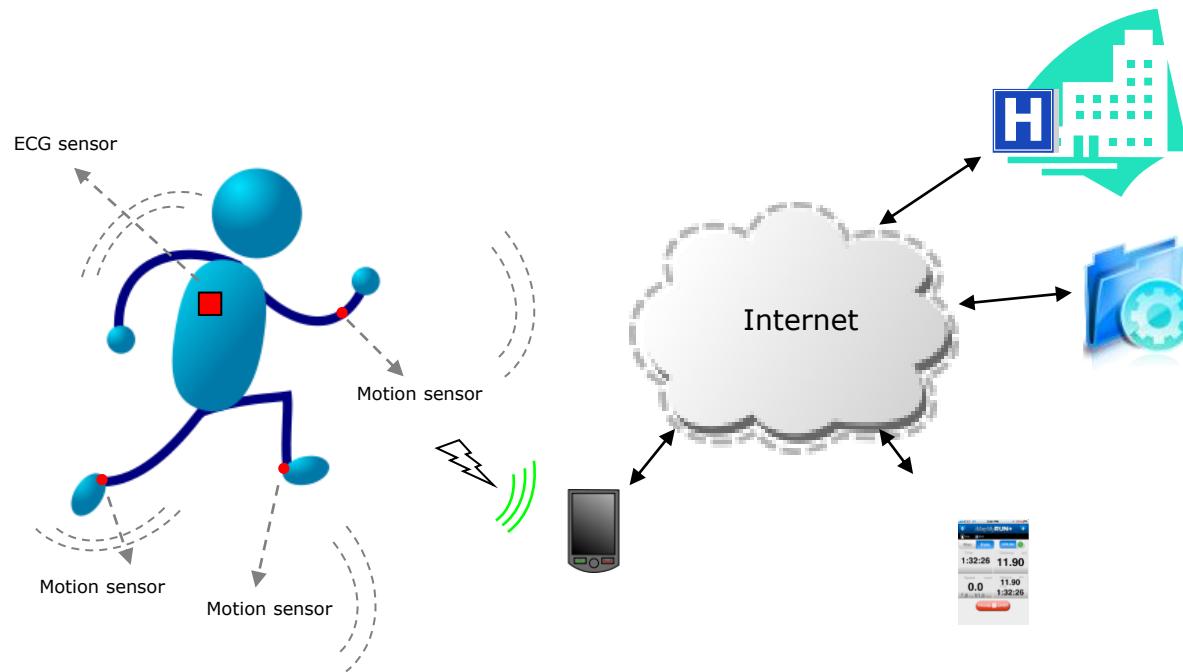


IOT

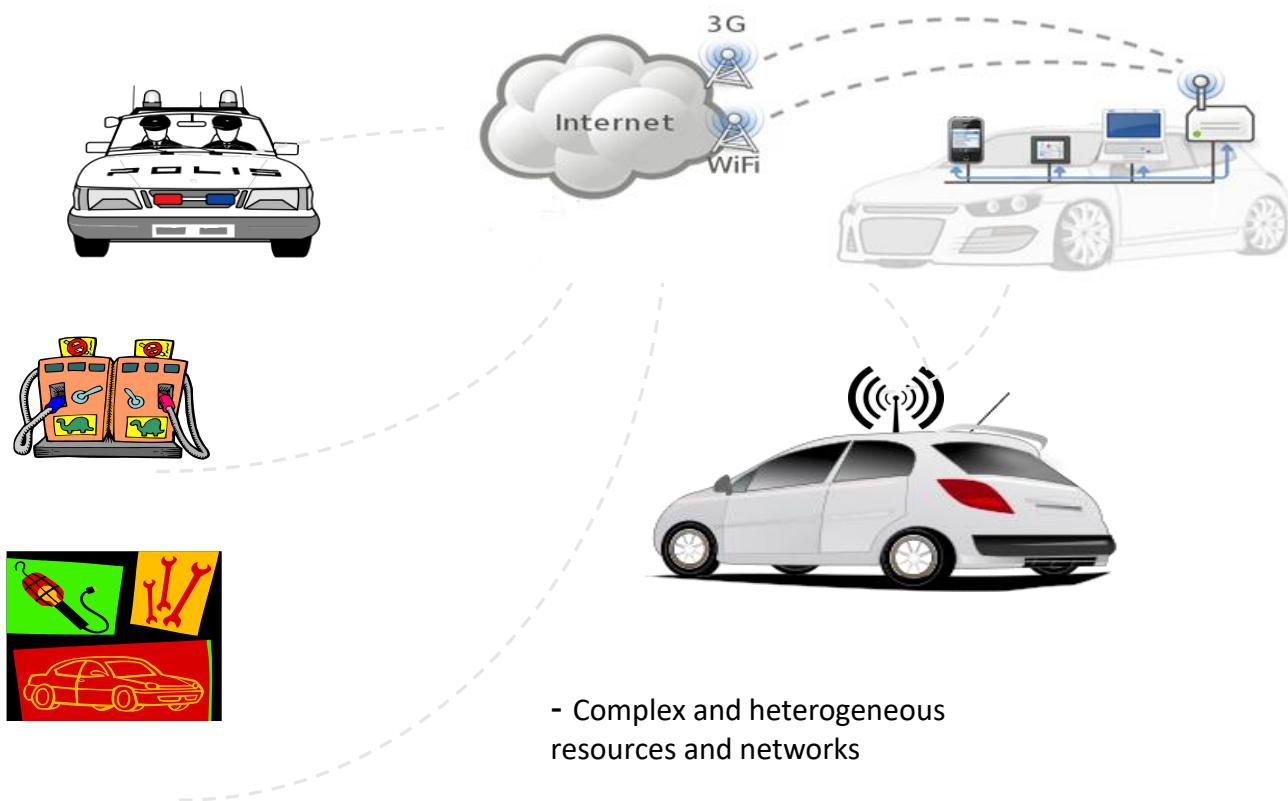
# Internet-of-Things Evolution



# People Connecting with Things



# Things Connecting with Things



# Where is IOT?

---

IOT is everywhere!

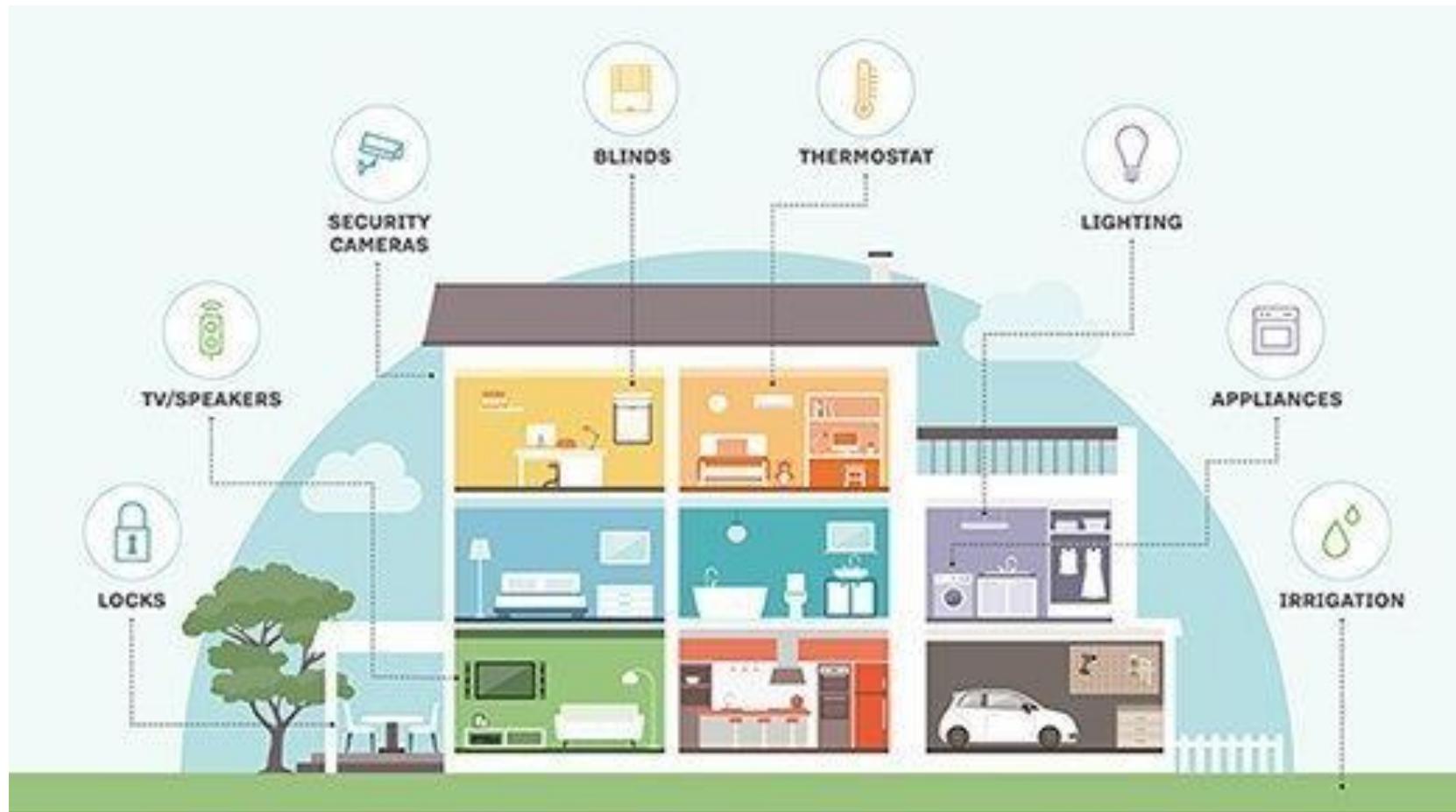




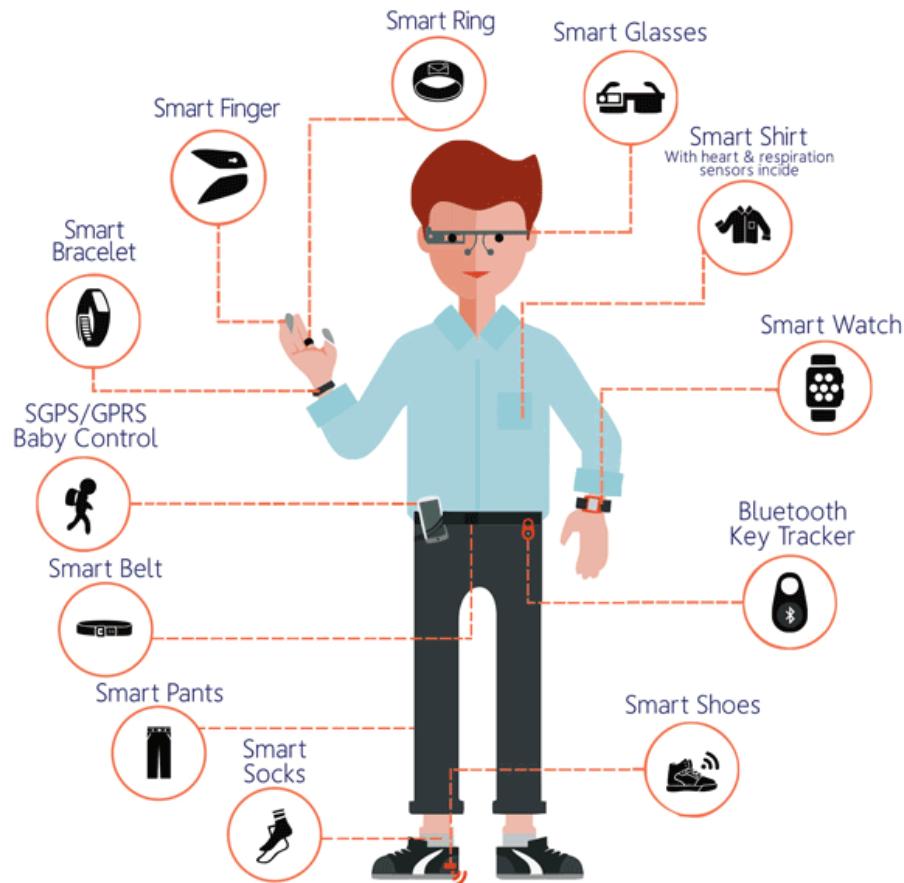
A large blue circle is positioned in the upper left area. To its right is a smaller yellow circle, and further to the right is a very small gray circle. A large dark gray shape, resembling a semi-circle or a large oval, is located in the upper right corner, partially overlapping the yellow circle.

Applications |

# Smart Homes

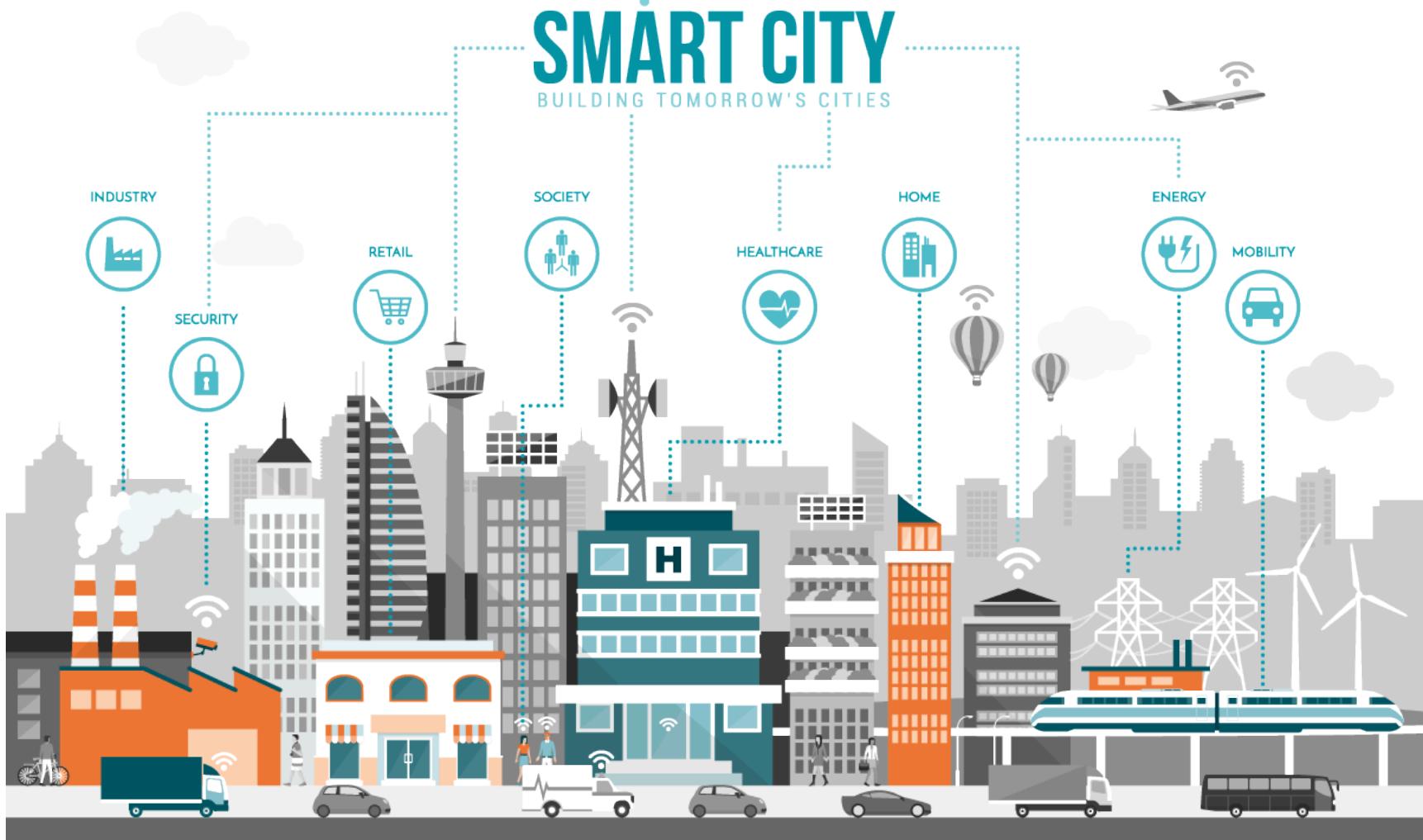


# Wearable



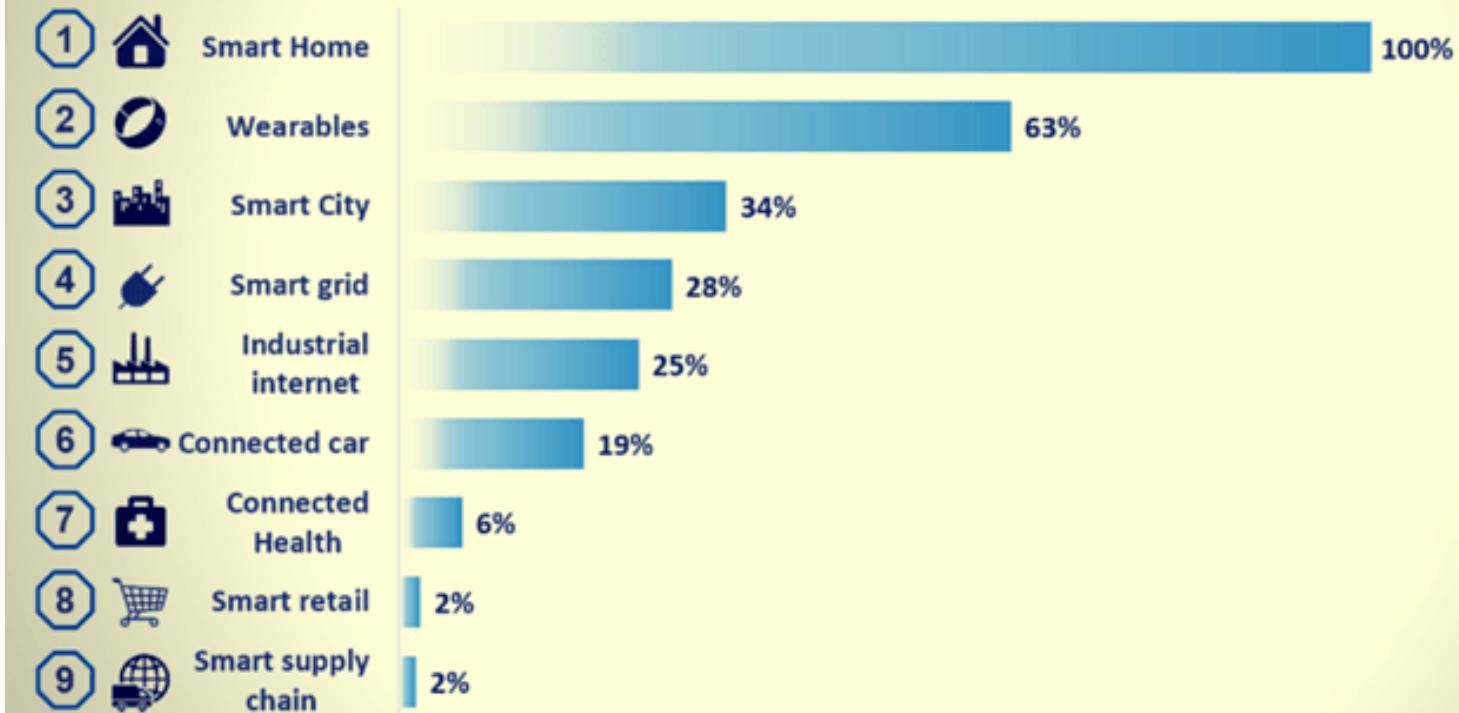
# SMART CITY

BUILDING TOMORROW'S CITIES



## The 10 most popular Internet of Things applications

A ranking based on web analytics



# Augment Existing Things

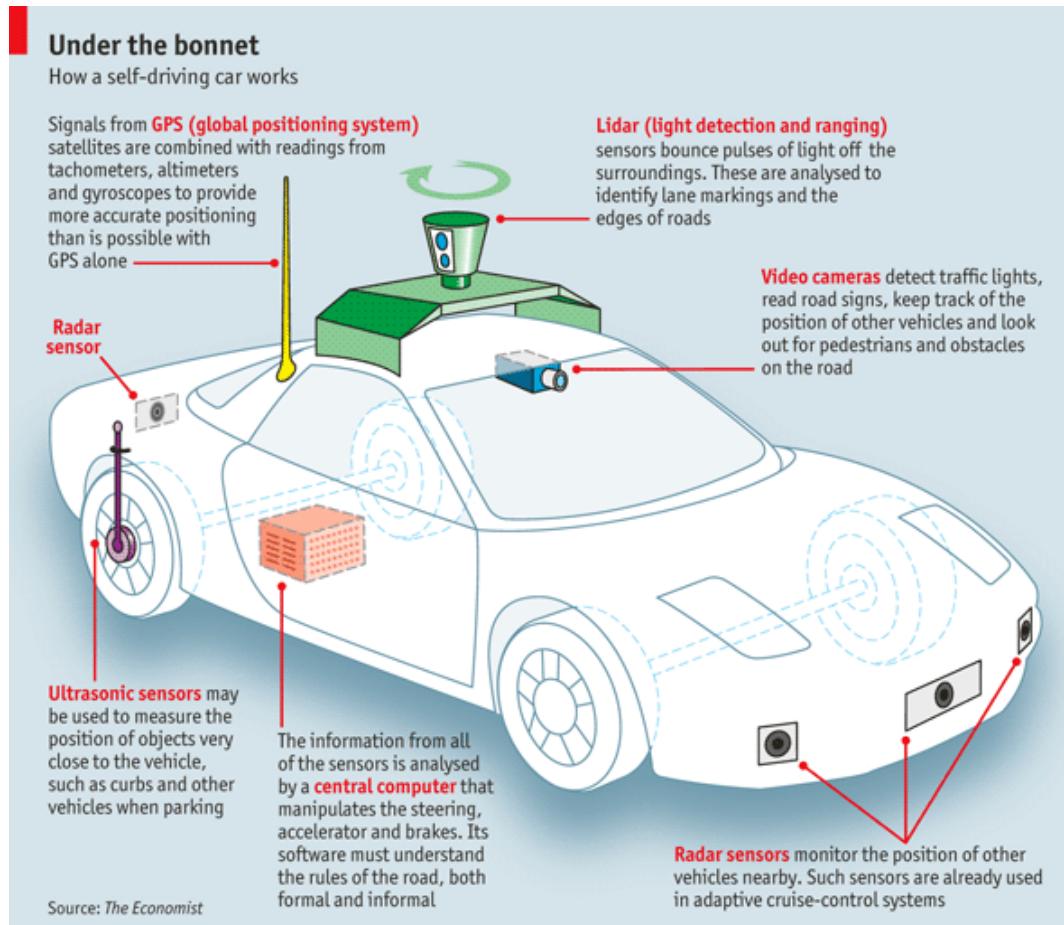




## Augmenting Life With New Things

- Smart City
- Smart Car
- Smart Me (healthcare, fitness, wellness)

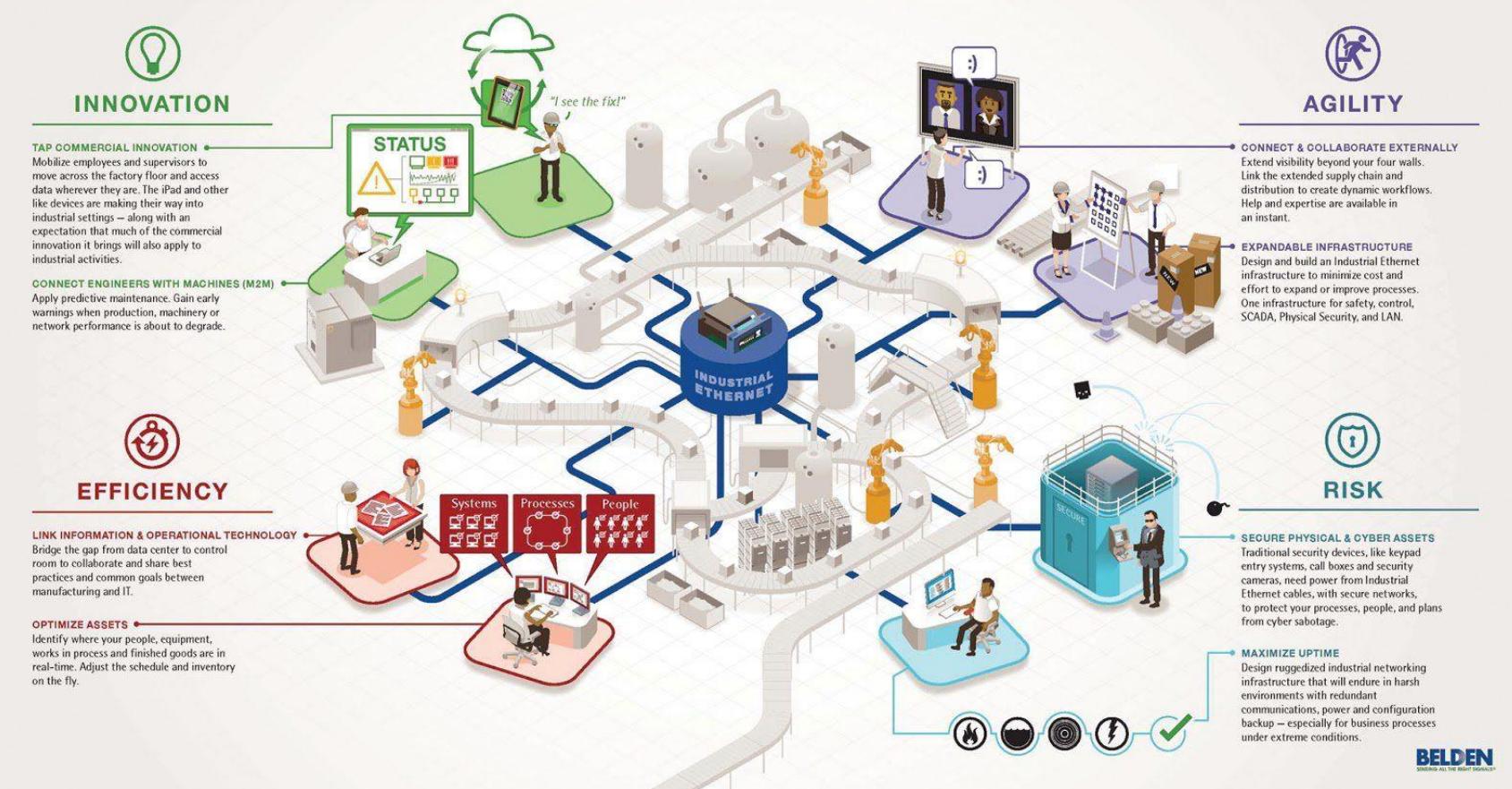
# Example: Connected Roadways



# Example: Connected Roadways

[State of Self-Driving Car](#)

# The Connected Factory in Action

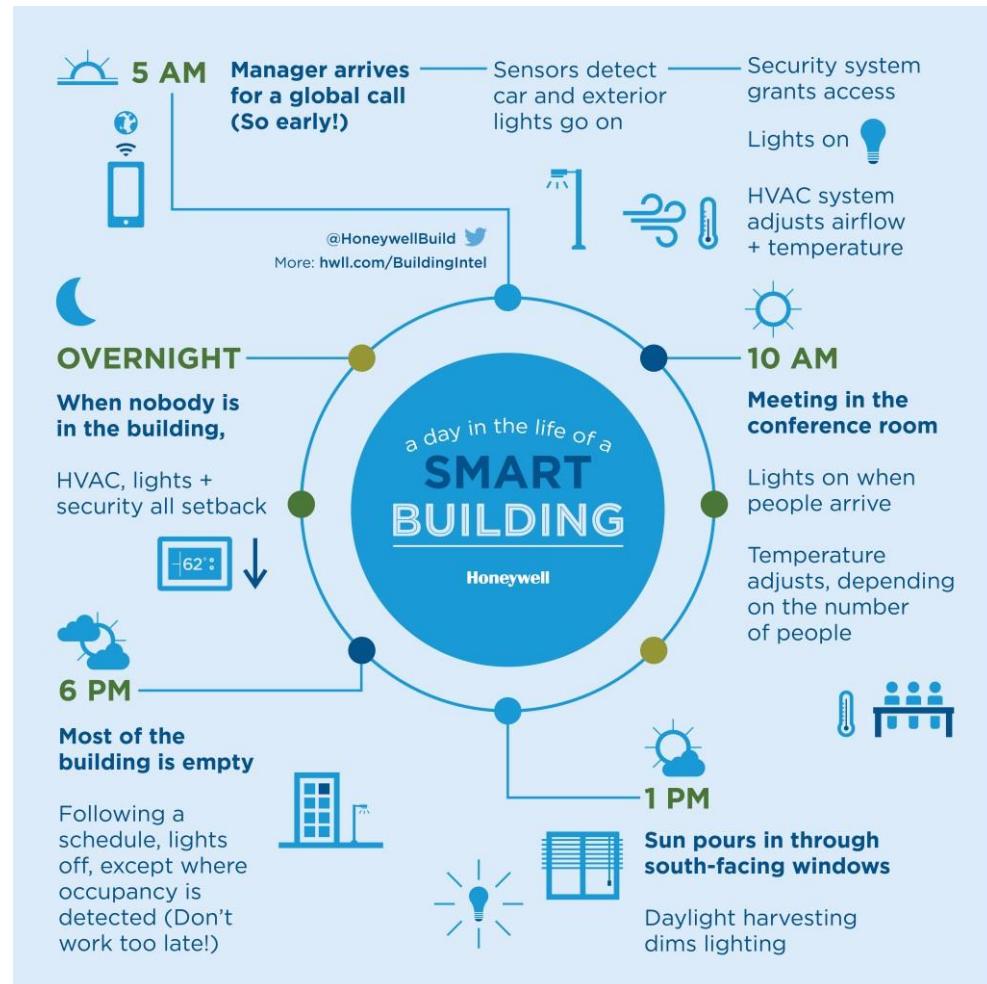


## Example: Connected Factory

- New product and service introductions faster
- Increasing production, quality, uptime
- Mitigating unplanned downtime
- Protecting from cyber threats
- Worker productivity and safety

# Example: Smart & Connected Buildings

- Energy management
- Lighting
- Safety
- HVAC
- Building automation
- Smart spaces

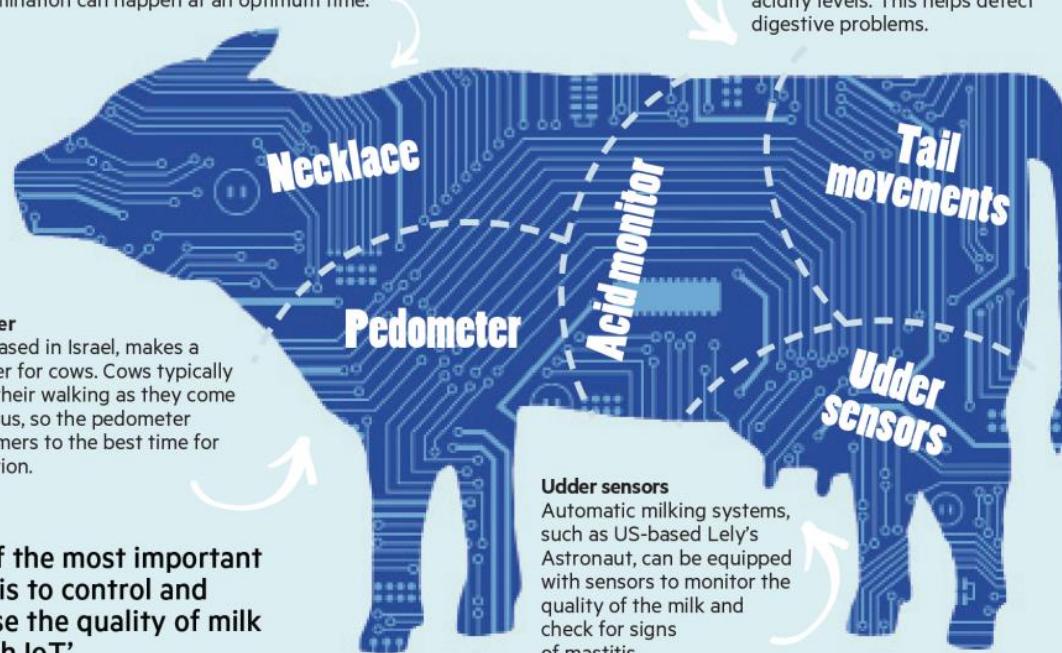


# Example: Smart Creatures

## The connected cow

### Necklace

Connecterra, a Dutch company, makes Fitbit-style necklaces that monitor a cow's movement and feeding habits. The sensor can be used to detect health problems and to tell when the cow is in heat, so that insemination can happen at an optimum time.



### Acid monitor

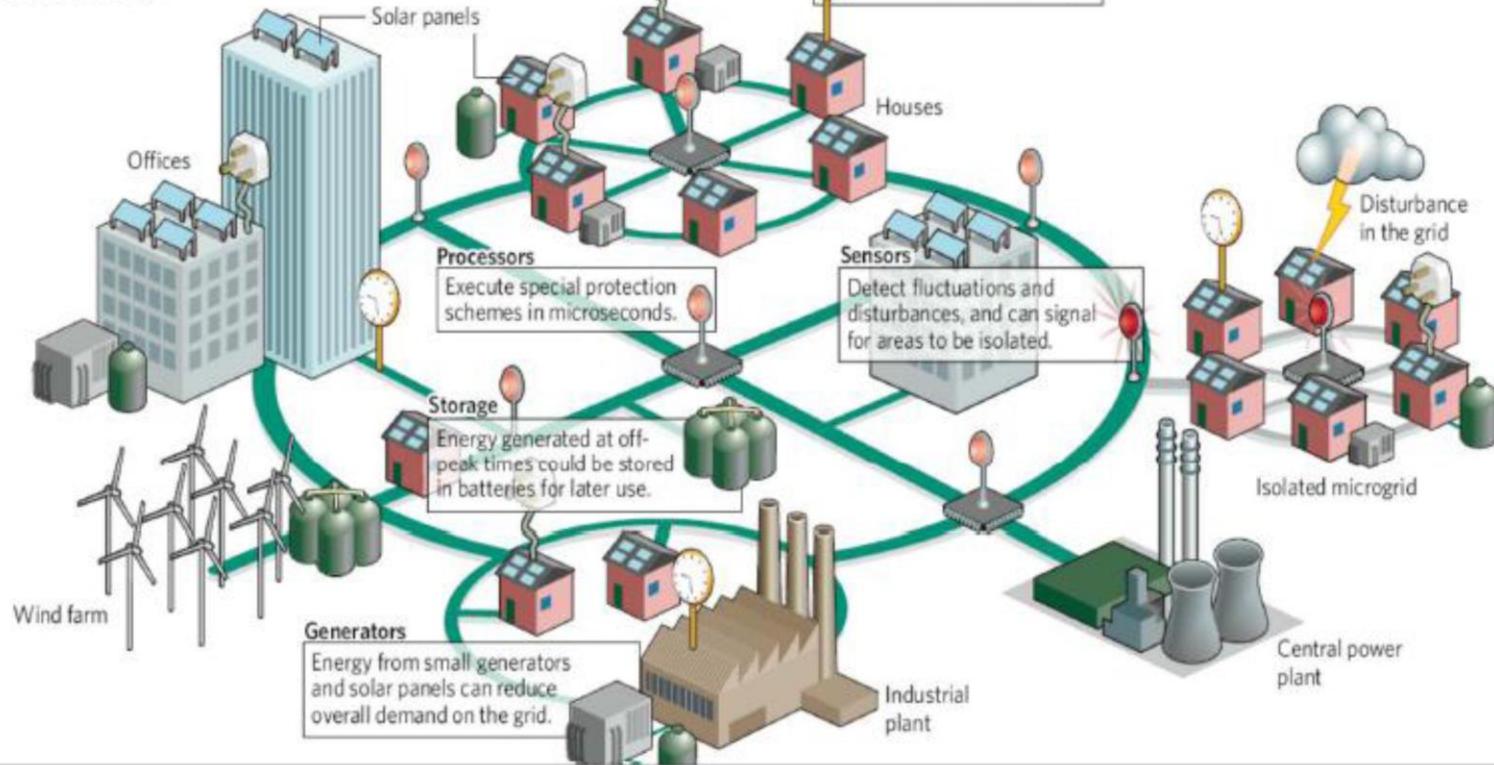
Well Cow, a British company, has developed a bolus that is inserted into the cow's rumen to monitor acidity levels. This helps detect digestive problems.

### Tail movements

Moocall, an Irish company, makes a birthing sensor that attaches to the tail. It measures tail movements triggered by labour contractions, and sends a farmer an SMS alert approximately one hour before a cow is due to calve.

## SMART GRID

A vision for the future — a network of integrated microgrids that can monitor and heal itself.



## Example: Smart Grid



## Enablers: Portability

Reducing the size of hardware to enable the creation of computers that could be physically moved around relatively easily



## Enablers: Miniaturization

Creating new and significantly smaller mobile form factors that allowed the use of personal mobile devices while on the move



50mm x 50mm

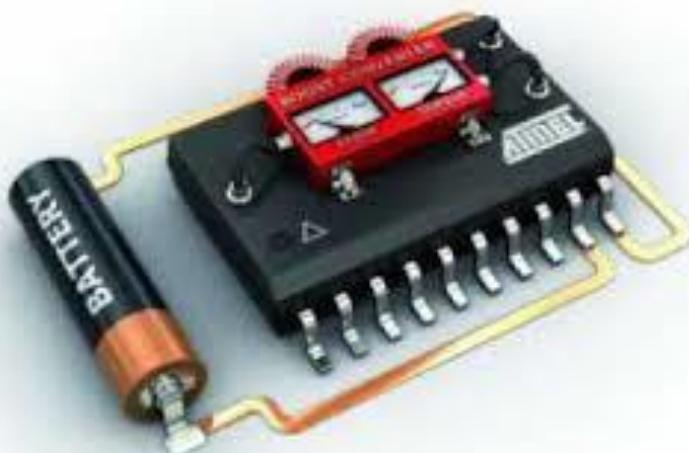


35mm x 35mm



15mm x 15mm

# Enablers: Low Power and Low Heat



- Low power architectures
- Low power radios
- Sleep modes
- Energy harvesting

# Enablers: Connectivity

- Developing devices and applications that allowed users to be online and communicate via wireless data networks while on the move



Bluetooth®

# Enablers: Convergence

---

Integrating emerging types of digital mobile devices, such as Personal Digital Assistants (PDAs), mobile phones, music players, cameras, games, etc., into hybrid devices.



# Enablers: Divergence

Opposite approach to interaction design by promoting information appliances with specialized functionality rather than generalized ones



# Enablers: Ecosystems



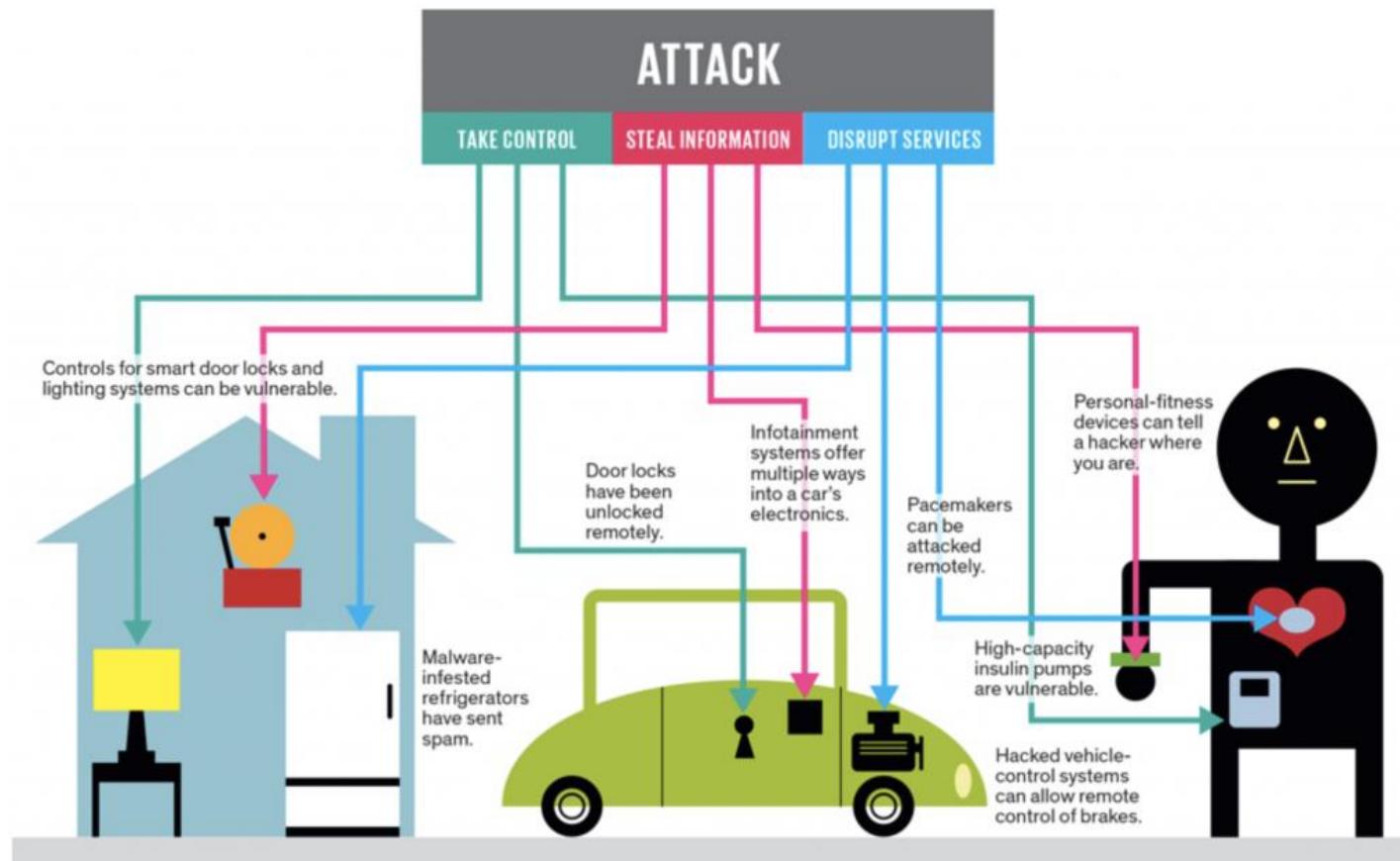
The emerging wave of *digital ecosystems* is about the larger wholes of pervasive and interrelated technologies that interactive mobile systems are increasingly becoming a part of.

# Example: Smartphone

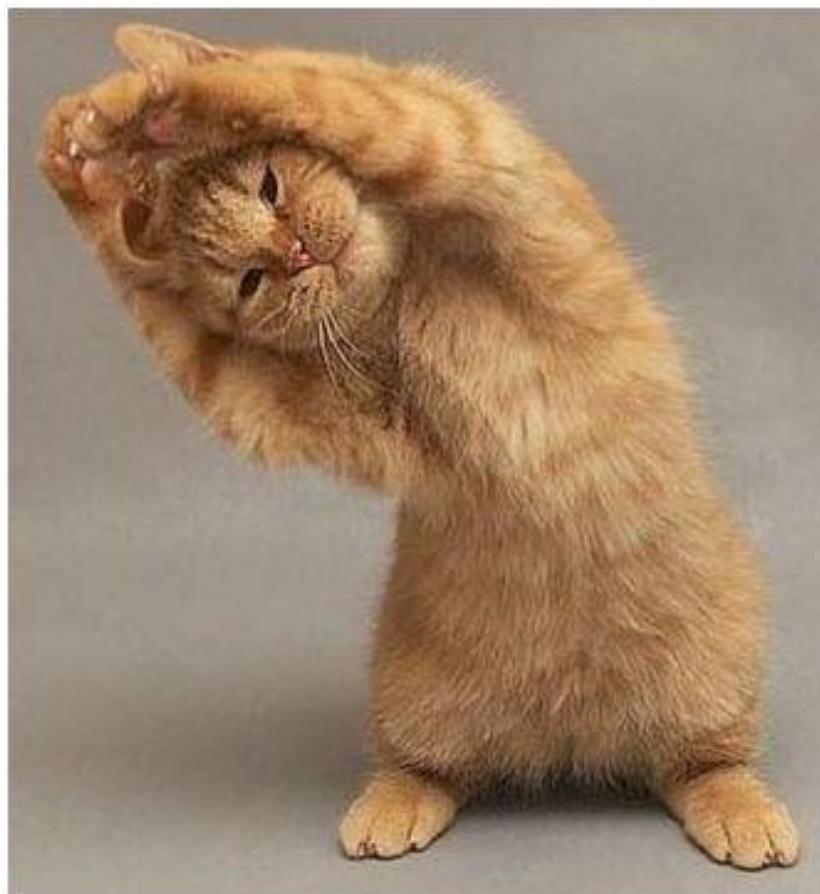
---

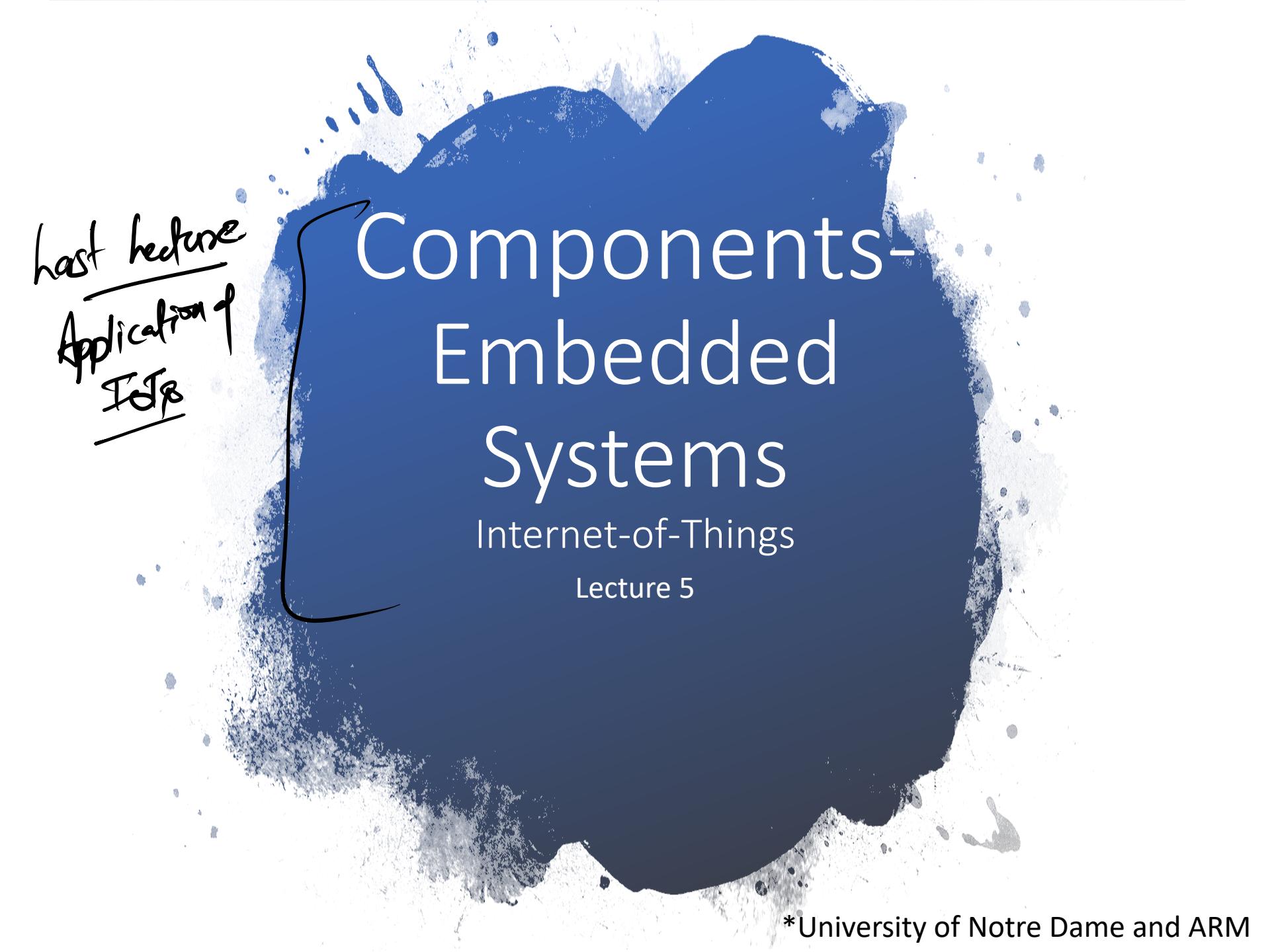
- Portability: carry it anywhere you want
- Miniaturization: make it possible to build device to fit in your pocket
- Connectivity: Wi-Fi, LTE/4G, cellular, Bluetooth
- Convergence: phone, camera, gaming device, movie streaming, music player, ...
- Digital Ecosystem: cloud, social networks, software development kits, app stores, big data, standardization ...

# IoT Issues & Challenges



# BREAK





*last lecture*

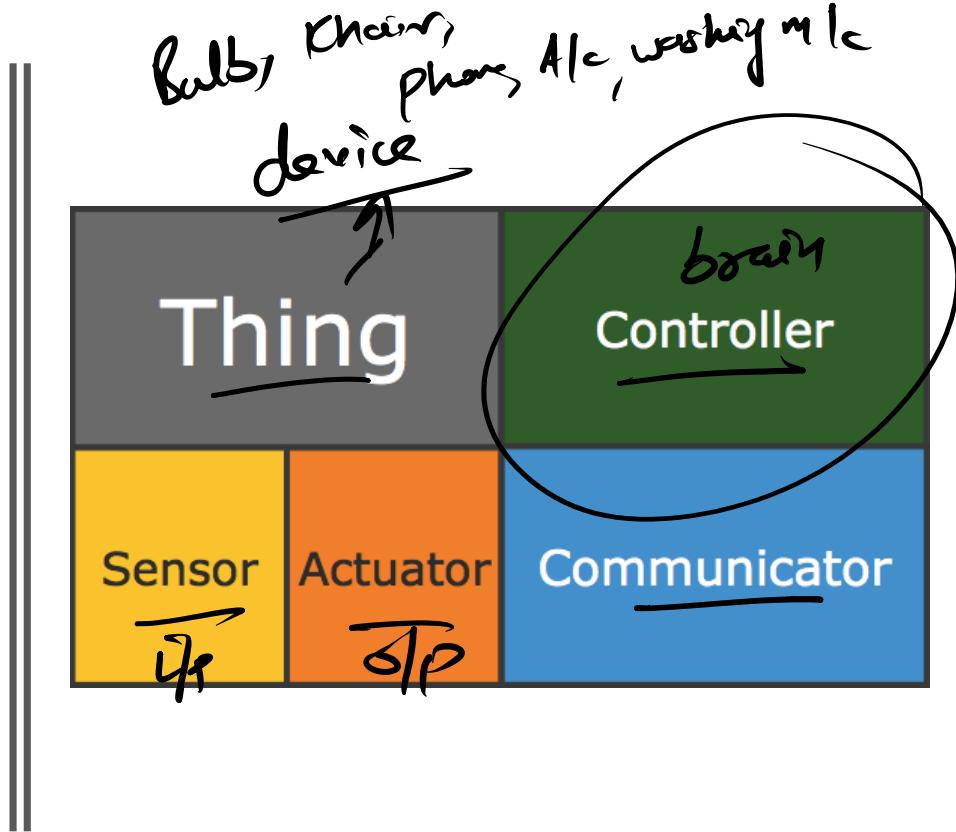
*Application of*

*IoT &*

# Components- Embedded Systems

Internet-of-Things

Lecture 5



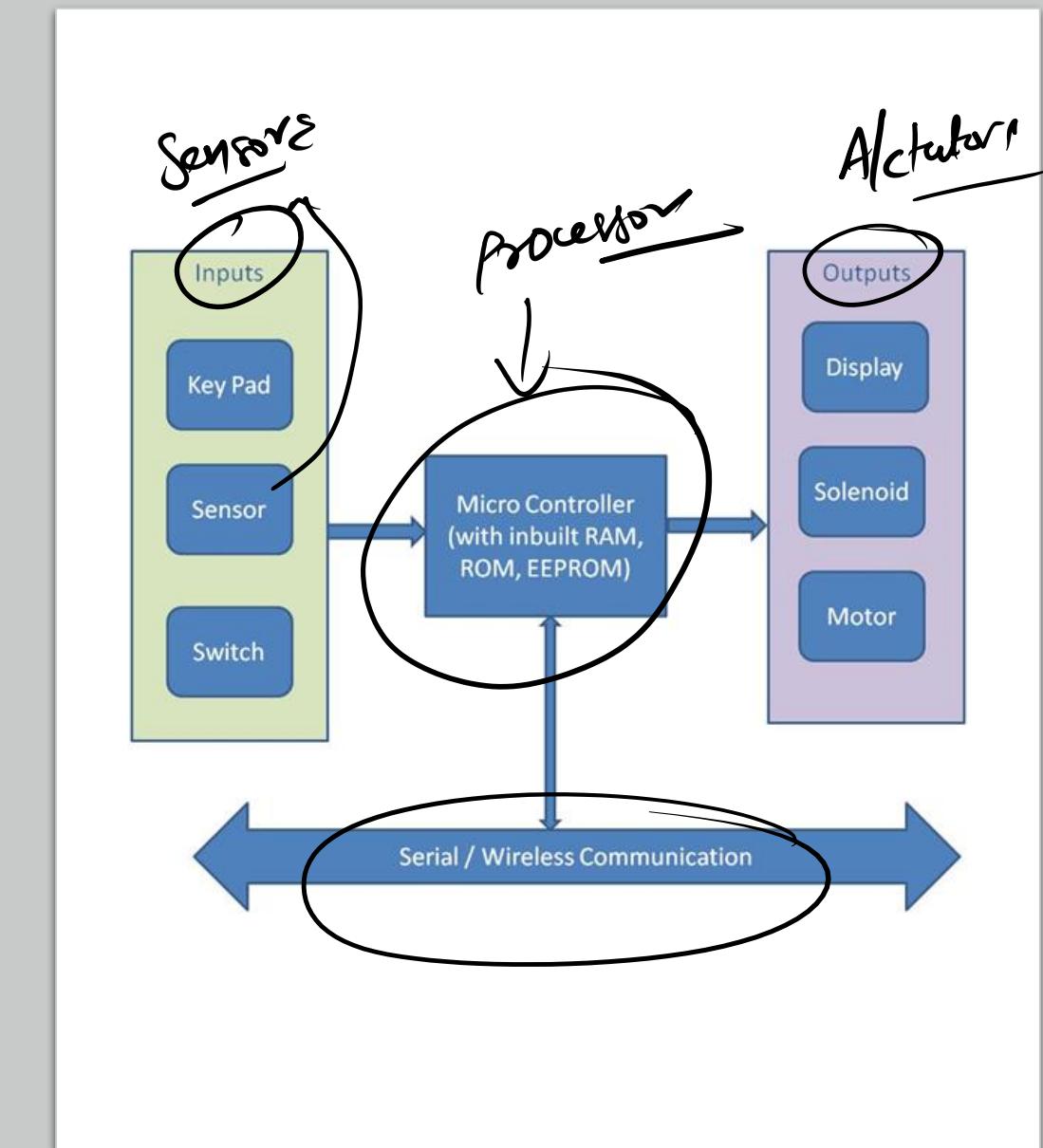
# Components of an IoT Device

# Embedded System/Computer

- “Any sort of device which includes a programmable computer but itself is not intended to be a general-purpose computer” - Wayne Wolf

*specl 2*  
to  
*some app'cns*

- General purpose
- Dedicated



# Embedded systems

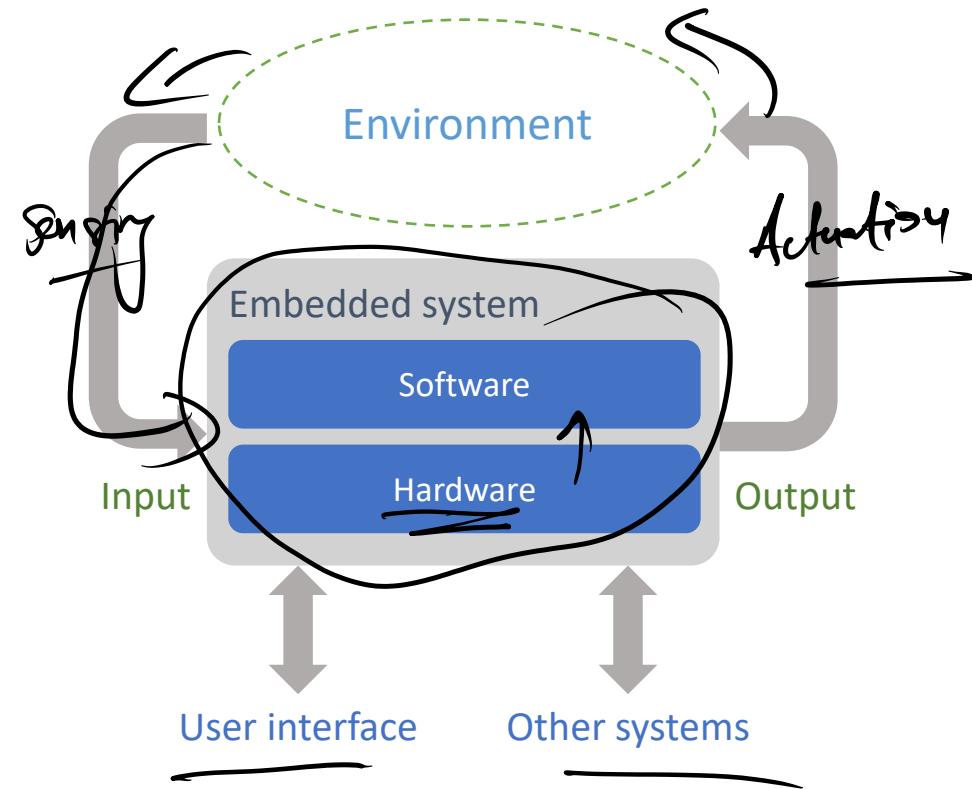
## Internet of Things (IoT)

- Application-specific computer system
- Built into a larger system
- Often with real-time computing constraints

## Adding embedded systems to larger systems

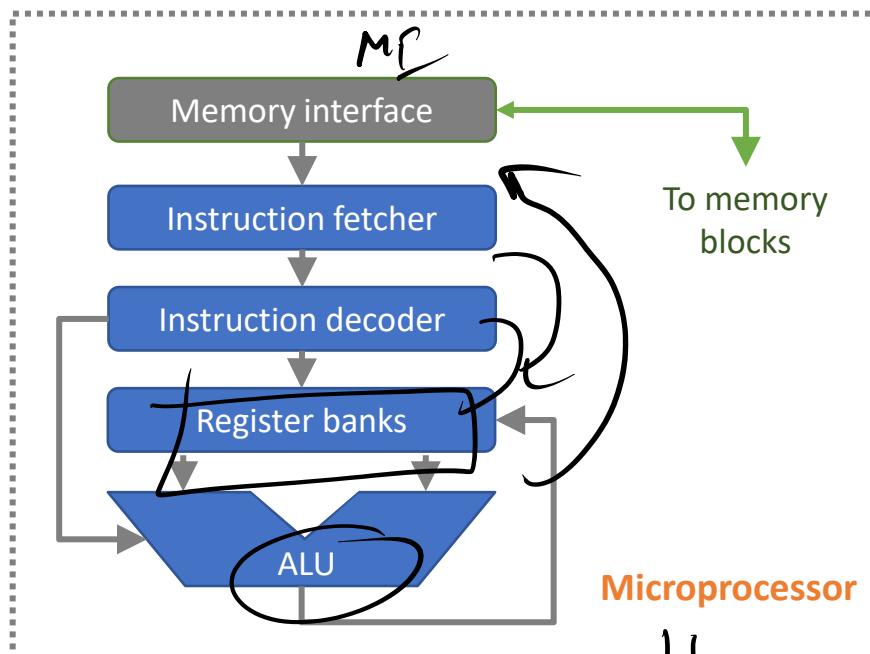
- Better performance
- More functions and features
- Lower cost, for e.g., through automation
- More dependability

**Examples:** smartphones, smart watches,  
printers, gaming consoles, wireless routers



# CPUs → MCUs → Embedded Systems

Microprocessor or Central Processing unit (CPU)

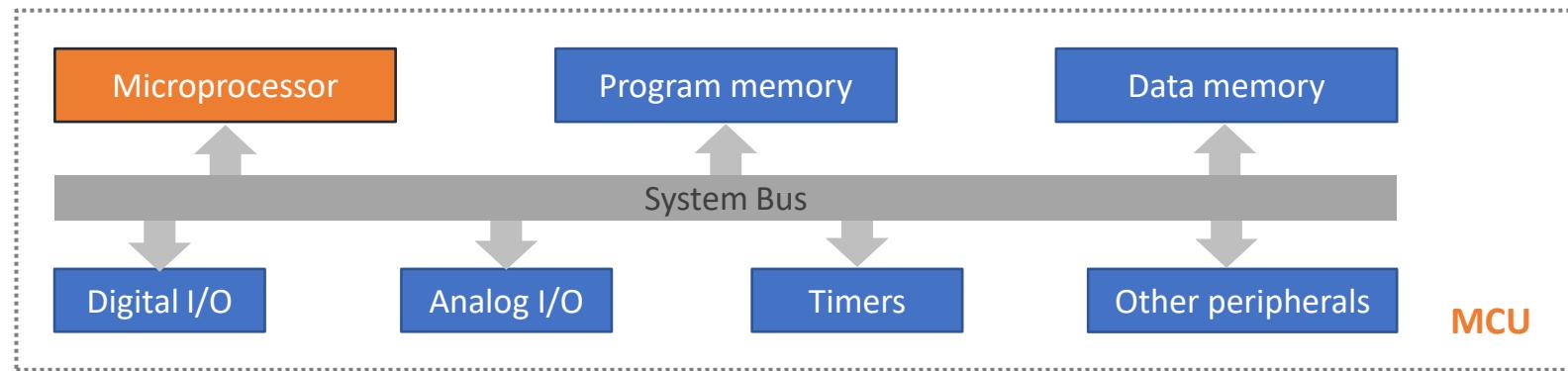


Defined typically as a single processor core that supports at least instruction fetching, decoding, and executing

  
Used for general purpose computing, but needs to be supported with memory and Input/Output (I/O) interfaces

# CPUs → MCUs → Embedded Systems

## Microcontroller Unit (MCU)



- Typically has a single processor core
- Has memory blocks, digital I/Os, analog I/Os, and other basic peripherals
- ⚙️ Used for basic control purposes, such as embedded applications

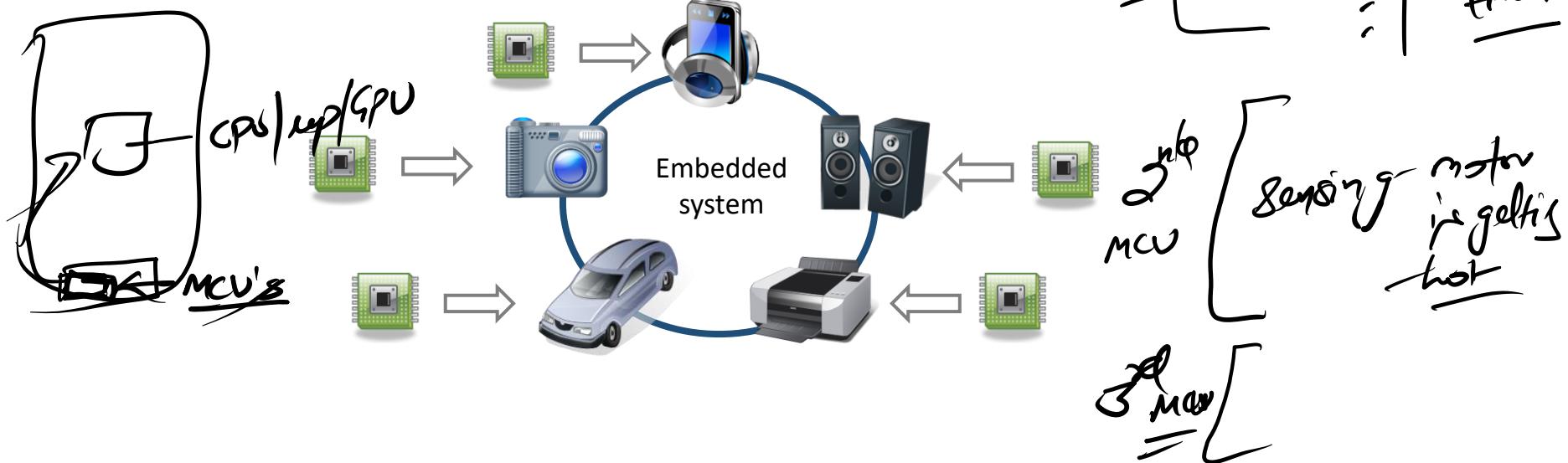
# CPUs → MCUs → Embedded Systems

## Embedded system

Typically implemented using MCUs

Often integrated into a larger mechanical or electrical system

Usually has real-time constraints



# Embedded system example: Bike computer

only  
one  
MCU

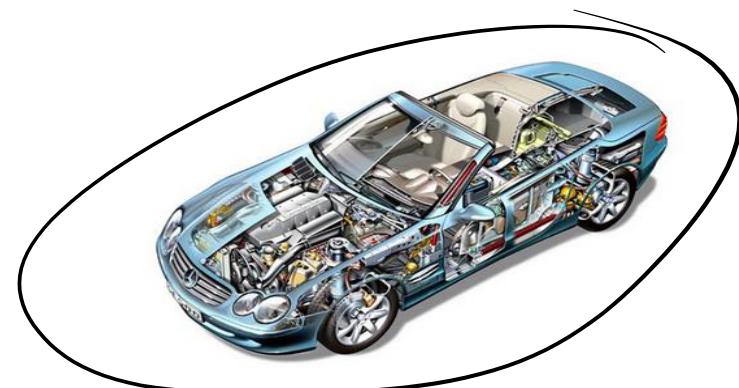
- Functions
  - Speed, cadence, distance, heart rate (HR) measurements
- Constraints
  - Size, weight, and cost; power and energy
- Inputs (Sensor)
  - Wheel rotation sensor and mode key
- Output
  - Liquid crystal display (LCD), BLE interface to smartphone
- Uses low performance microcontroller



# Embedded system example: Gasoline engine control unit

- Functions
  - Fuel injection ✓
  - Air intake setting ✓
  - Spark timing ✓
  - Exhaust gas circulation ✓
  - Electronic throttle control ↘
  - Knock control ✓
- Constraints
  - Reliability in a harsh environment
  - Cost
  - Weight

- Many inputs and outputs
  - Discrete sensors and actuators (MCU)
  - Network interface to rest of the car (MCU)
- Uses high performance microcontroller
  - E.g., 32-bit, 3MB flash memory, 150–300MHz

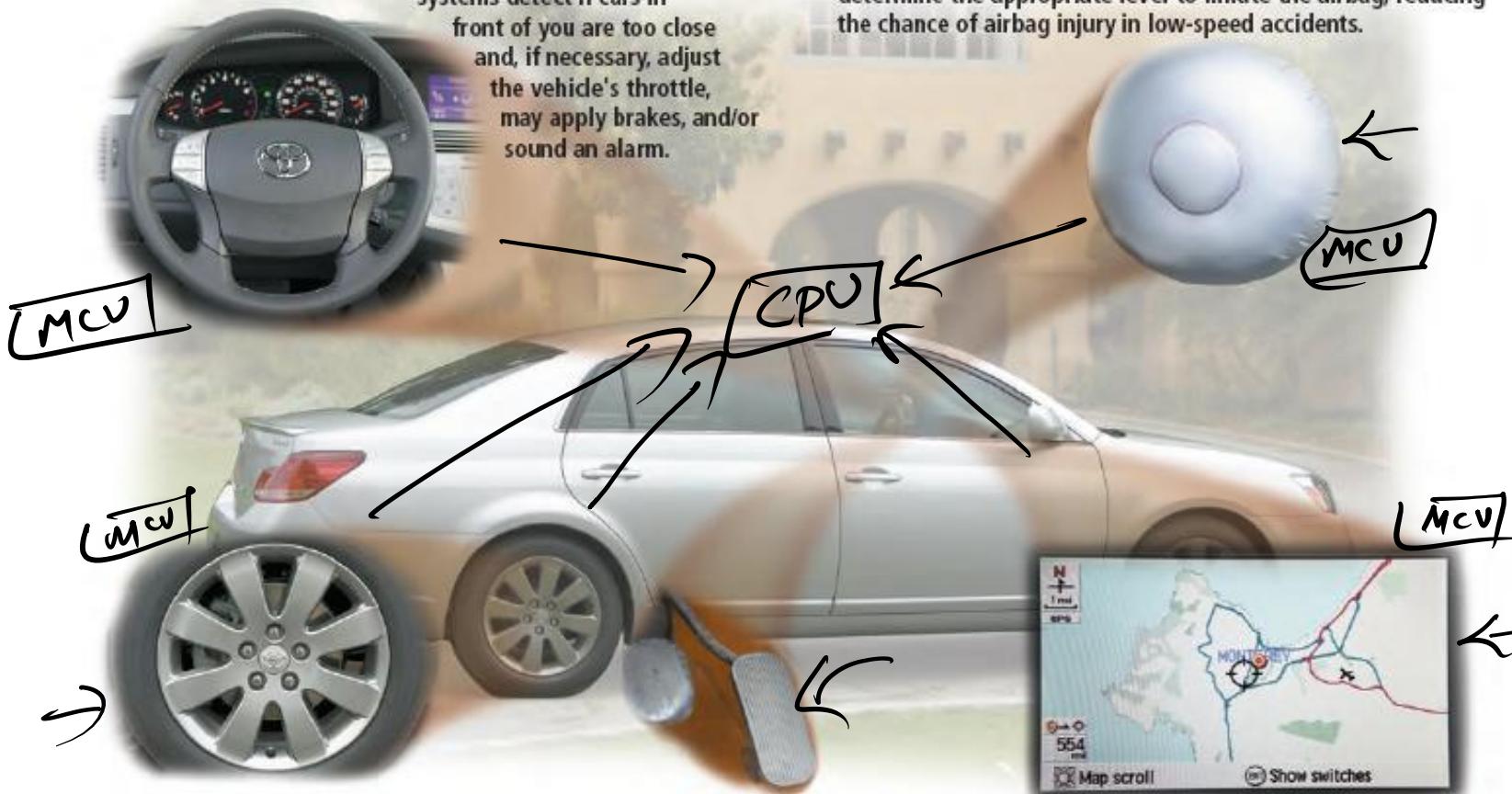


# Automotive Embedded Systems

(Self-drive car)

Adaptive cruise control systems detect if cars in front of you are too close and, if necessary, adjust the vehicle's throttle, may apply brakes, and/or sound an alarm.

Advanced airbag systems have crash-severity sensors that determine the appropriate level to inflate the airbag, reducing the chance of airbag injury in low-speed accidents.



Tire pressure monitoring systems send warning signals if tire pressure is insufficient.

Drive-by-wire systems sense pressure on the gas pedal and communicate electronically to the engine how much and how fast to accelerate.

Cars equipped with wireless communications capabilities, called telematics, include such features as navigation systems, remote diagnosis and alerts, and Internet access.

# Automotive Embedded Systems

- Today's high-end automobile may have 100+ microprocessors:  
Seat belt; dashboard devices; engine control; ABS; automatic stability control; navigation system; infotainment system; collision avoidance system; tire pressure monitoring; lane warning; adaptive cruise control; climate control; airbag control unit; electric window and central locking; parking aid; automatic wiper control; alarm and immobilizer; power seat; electric power steering; electronic transmission; active suspension

Gr.  
Project E

, pedestrian  
Peds  
Stoed li.  
Red li.  
Zebra Crone  
Other cars

side mirror  
Doft mirr-  
A/c

## Embedded Processor Market

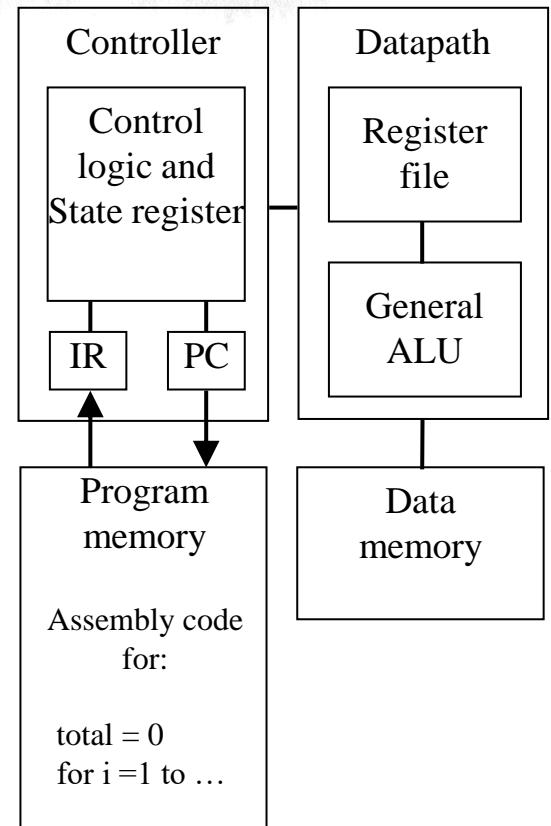
80 million PCs every year

3 billion embedded CPUs  
every year

Embedded systems  
market growing, while PC  
market mostly saturated

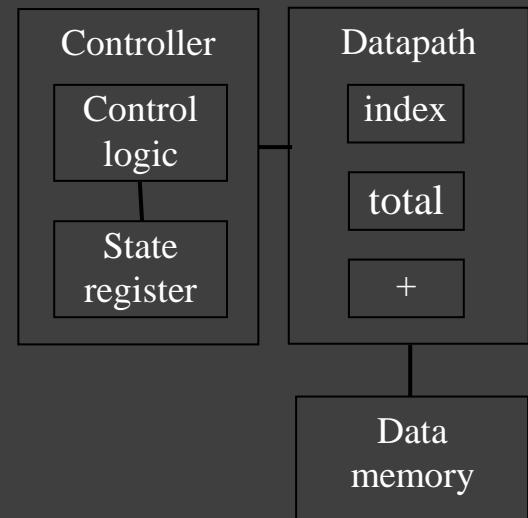
# General-Purpose Processor

- Programmable device, “microprocessor”
- Features
  - Program memory
  - General data path with large register file and general ALU
- User benefits
  - Low time-to-market and NRE costs
  - High flexibility
- Examples: Intel Core i7, AMD Ryzen 5, etc.



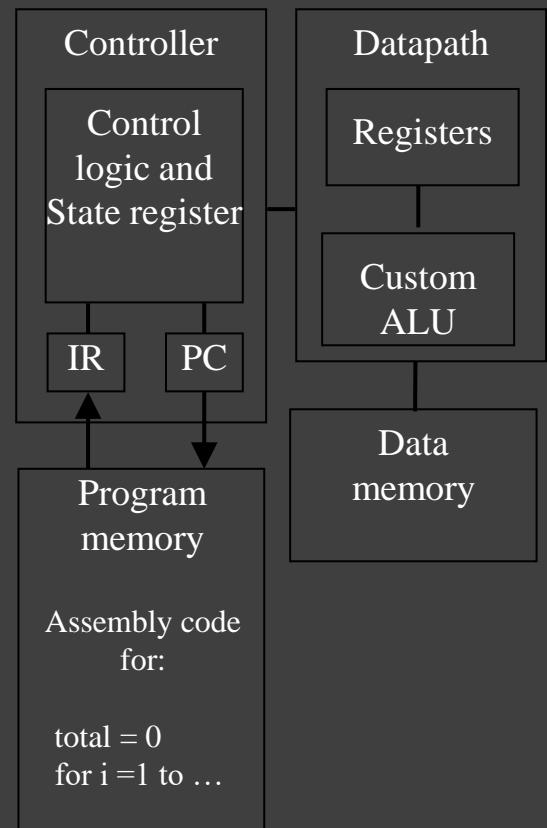
# Dedicated Processor

- Digital circuit designed specifically for one purpose
- Features
  - Contains only the components needed to execute a single program
  - No program memory
- Benefits
  - Fast
  - Low power
  - Small size



# Application-Specific Processor (ASIC)

- Programmable processor optimized for a particular class of applications that have common characteristics.
- Features
  - Program memory
  - Optimized data path
  - Special functional units
- Benefits
  - Some flexibility, good performance, size, and power, “reusable”

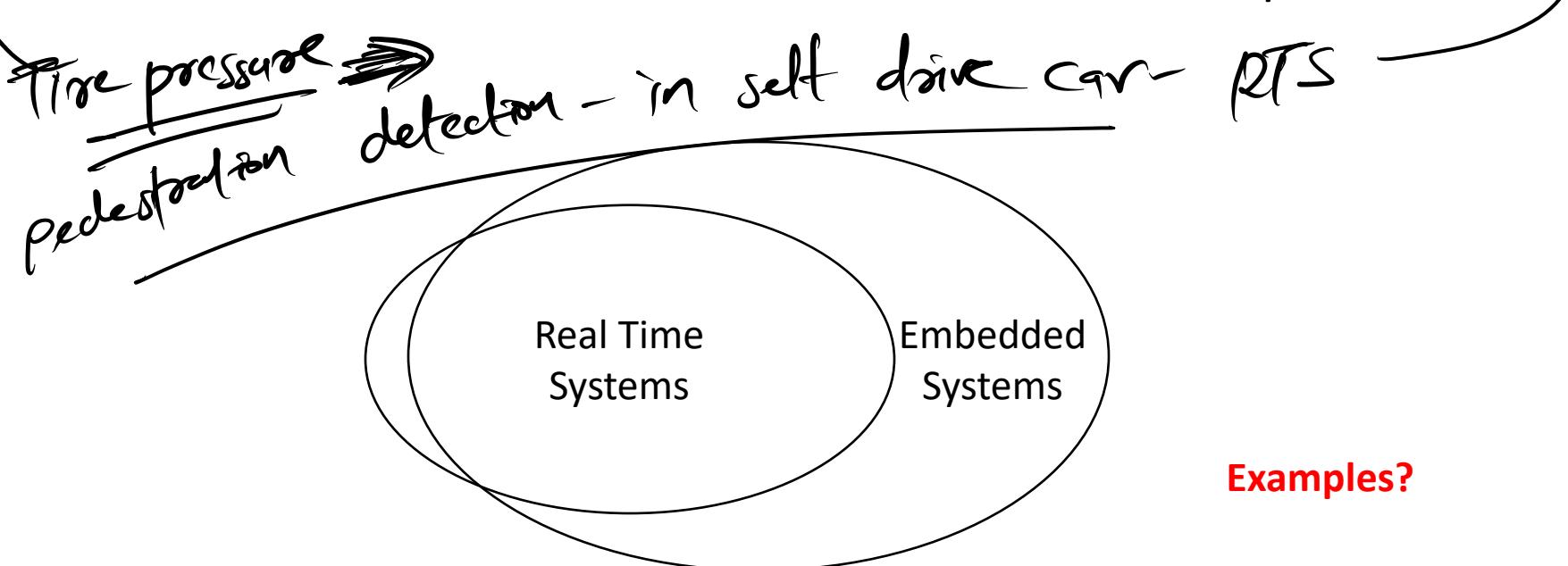


# Characteristics of Embedded Systems

- Dedicated functionality ↗
- Real-time operation ↗
- Small size and low weight ↗
- Low power ↗
- Harsh environments ↗
- Safety-critical operation ↗
- Cost sensitive ↗

# Embedded vs. Real Time Systems

- Embedded system: is a computer system that performs a limited set of specific functions; it often interacts with its environment
- RTS: Correctness of the system depends not only on the logical results, but also on the **time** in which the results are produced



## Examples

- Real Time Embedded:
    - Nuclear reactor control ✓
    - Flight control ✓
    - Basically, any safety critical system ✓
    - GPS ✓
    - MP3 player ✓
    - Mobile phone ✓
  - Real Time, but not Embedded:
    - Stock trading system
    - Skype
    - Pandora, Netflix
  - Embedded, but not Real Time:
    - Home temperature control ↗
    - Sprinkler system
    - Washing machine, refrigerator, etc.
- Software

# Benefits of embedded systems



## Greater performance and efficiency

More sophisticated control through software



## Lower cost

Cheaper components  
Reduced manufacturing costs  
Reduced operating and maintenance costs



## More features

Many not possible or impractical using other approaches

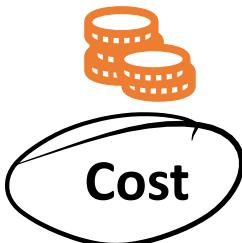


## Better dependability

Adaptive systems that can compensate for failures

Better diagnostics to improve repair time

# Constraints specific to embedded devices



## Cost

Competitive markets penalize products that do not deliver adequate value for money



## Size and weight limits

Mobile (aviation, automotive) and portable (e.g., handheld, wearable) systems



## Power and energy limits

Battery capacity, cooling limits



## Environment

Temperatures may range from -40 degrees C to 125 degrees C, or even more.

# Functions of embedded systems



## Closed-loop control system

Monitor a process, adjust an output to maintain the desired set point of operation (temperature, speed, direction, etc.)



## Sequencing

Step through different stages based on environment and system conditions



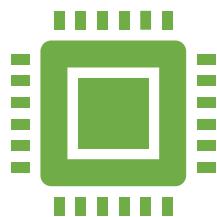
## Signal processing

Remove noise, select desired signal features



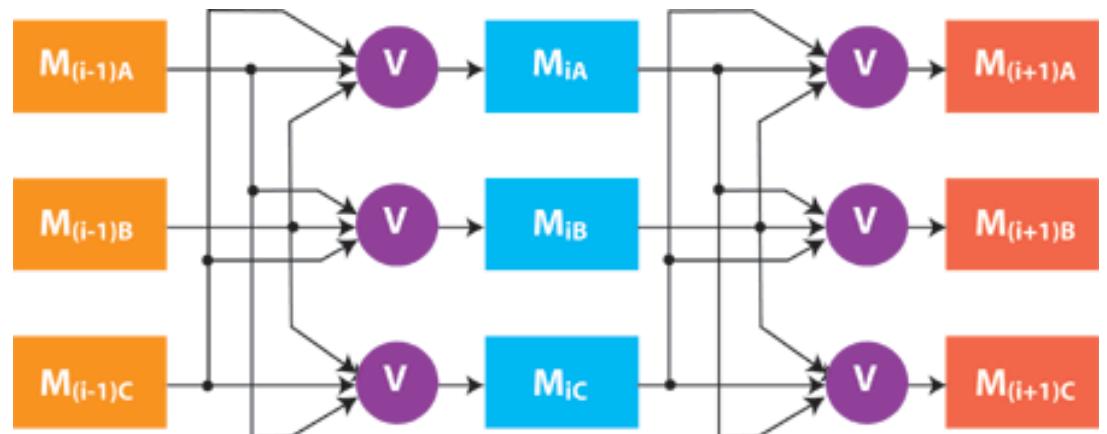
## Communications and networking

Exchange information reliably and quickly

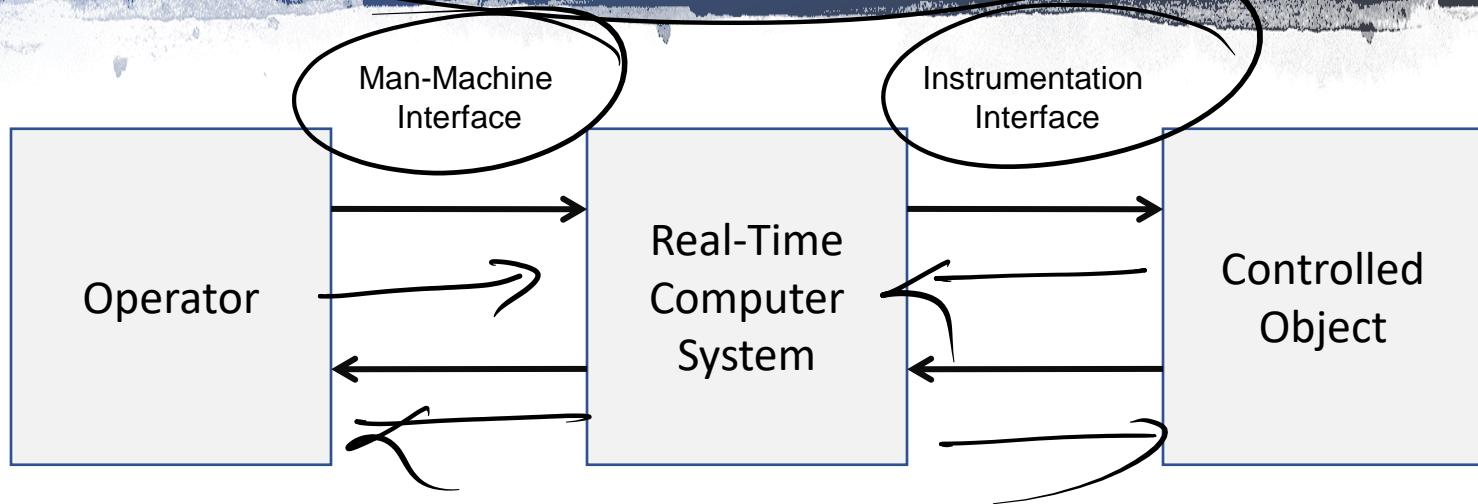


## Characteristics of RTS

- Event-driven (reactive) vs. time-driven
- Reliability/fault-tolerance requirements (example: triple modular redundancy)
- Predictability
- Priorities in multi-programmed systems



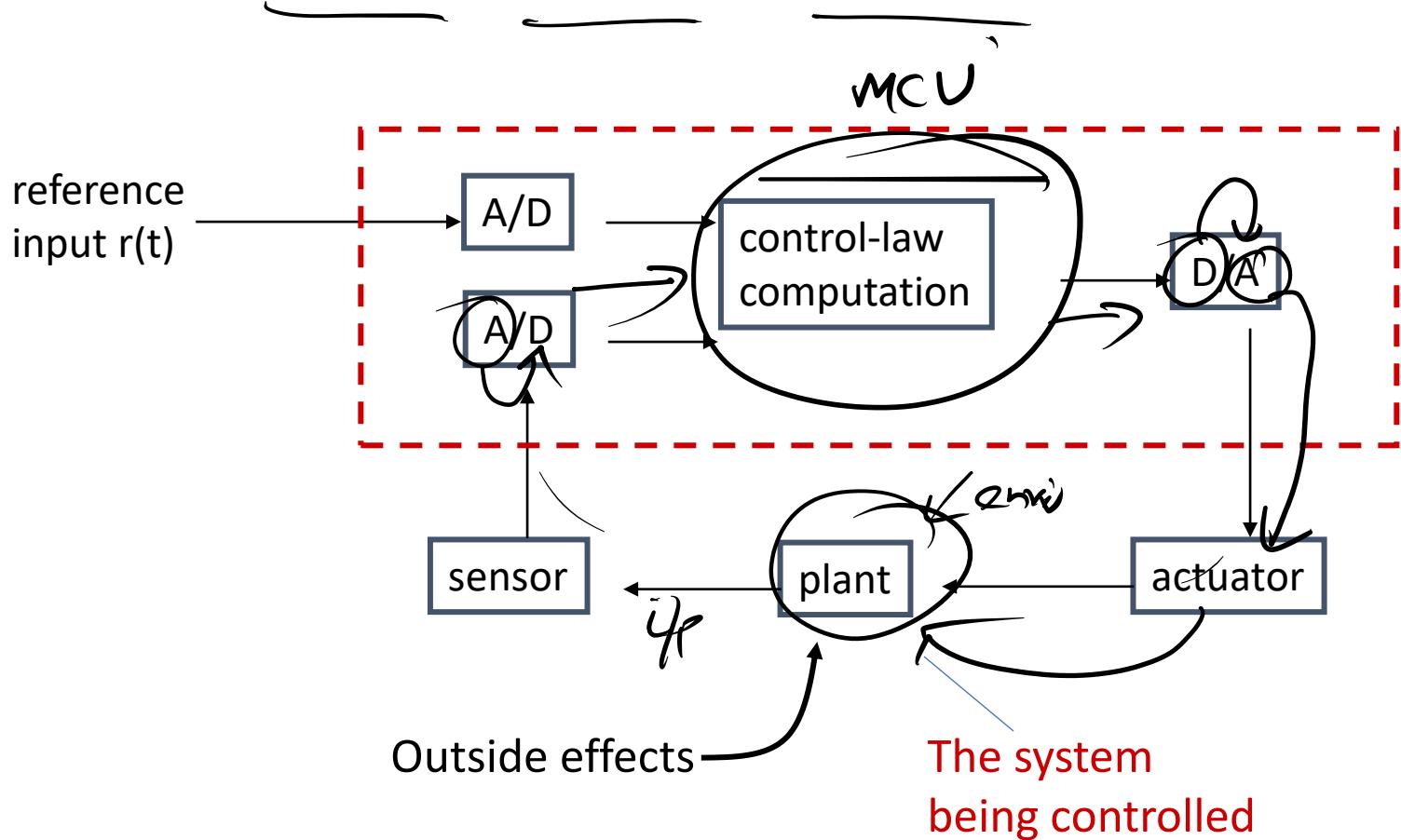
# Control Systems



- **Man-machine interface:** input devices, e.g., keyboard and output devices, e.g., display
- **Instrumentation interface:** sensors and actuators that transform physical signals and digital data.

# Control System Example

**Example:** A simple one-sensor, one-actuator control system.



# Control Systems Cont'd.

## Pseudo-code for this system:

```
set timer to interrupt periodically with period  $T$ ;  
at each timer interrupt do  
    do analog-to-digital conversion to get  $y$ ;  
    compute control output  $u$ ;  
    output  $u$  and do digital-to-analog conversion;  
end do
```

process it  
decide it

$T$  is called the sampling period.  $T$  is a key design choice. Typical range for  $T$ : seconds to milliseconds.

## Sensors and Actuators

### Sensors:

- They are mainly input components
- They sense and collect surrounding information

### Actuators:

- They are mainly output components
- They alter the surrounding

# Communications

- Connects devices with each other & the cloud
- Communication type:
  - Wireline (e.g., copper wires, optical fibers)
  - Wireless (e.g., RF, IR); RF-based communication is the most popular choice
- Popular RF-based communication solutions:
  - IEEE 802.15.4 (LR-WPANs) (Zigbee, 6LoWPAN) LoRaWAN
  - IEEE 802.11 (or Wifi)
  - Bluetooth (BLE)
  - Near Field Communication (NFC), e.g., RFID



Thank You!!

Next Session:  
**Human  
Computer  
Interaction**

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn:

<https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# Human Computer Interaction

Internet-of-Things (IoT)

---

COCOS20

# Smart Objects



Smart Refrigerator

- Objects that are able to **sense** the environment, **interpret** the environment, **self-configure**, **interact** with other objects and exchange information with people

# Traditional Computing System: HCI



***"Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them."*** -- Association for Computing Machinery.



(Bad) Examples of User Interfaces

TOO COOL TO DO DRUGS

COOL TO DO DRUGS

DO DRUGS

DRUGS



(Bad) Examples of User Interfaces



(Bad) Examples of User Interfaces



resume aborted transfer?

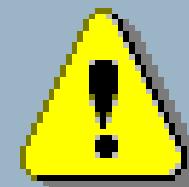
A file with this name already exists.  
Do you want to resume an aborted transfer?  
(Click No to overwrite your existing file  
and start a new transfer.)

Yes

No

Cancel

Microsoft Access



**Wrong button!**

This button doesn't work

**Solution**

Try another.

OK

(Bad) Examples of User Interfaces

# Why is HCI Important?

- It can affect
  - Effectiveness
  - Productivity
  - Morale
  - Safety
- Bad interfaces:
  - Confusing
  - Cumbersome
  - Time-consuming
  - Uninformative
  - Lead to errors
  - ...

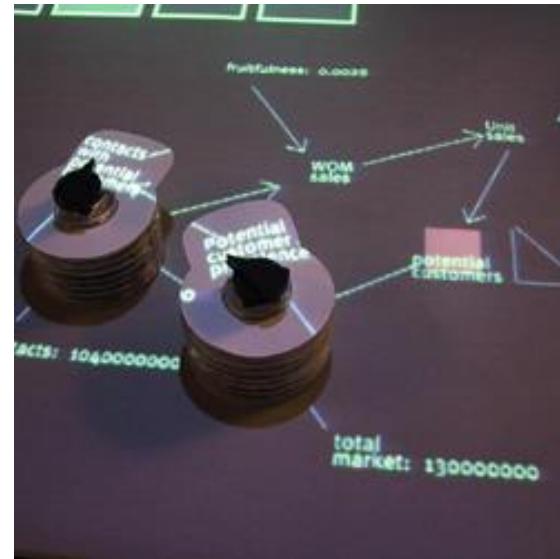


# Interfaces

- **Keyboard/mouse/screen/speakers**
- Pen input
- Touch
- Speech/audio/sound
- Gesture, eye movement
- Tangible interfaces
- Virtual/augmented reality (VR, AR)
- Wearable computing
- **Multi-modal** interactive interfaces: more than just one input/output channel

## Interface Discussion

- Ease-of-Use?
- Flexibility?
- Accuracy?
- Safety?
- Privacy?



# Touch as Input



# Gesture/Motion as Input

**1**

An eye tracker consists of cameras, projectors and algorithms.

**2**

The projectors create a pattern of near-infrared light on the eyes.

**3**

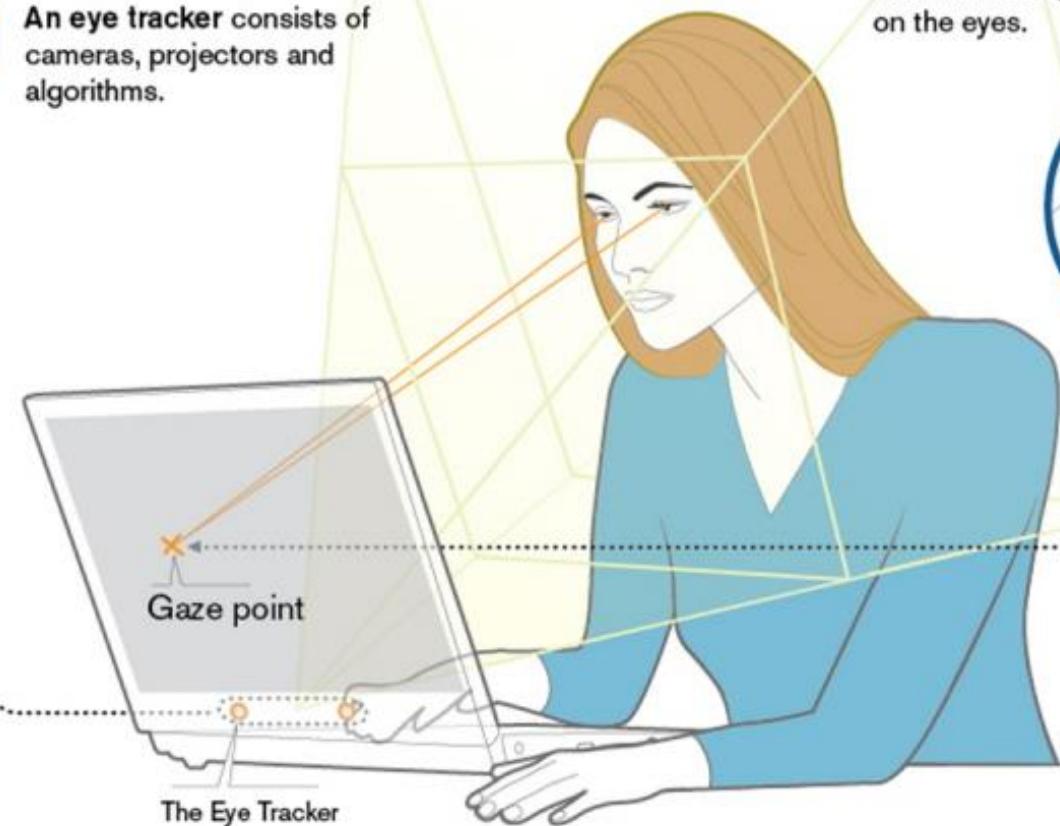
The cameras take high-frame-rate images of the user's eyes and the patterns.

**4**

The image processing algorithms find specific details in the user's eyes and reflections patterns.

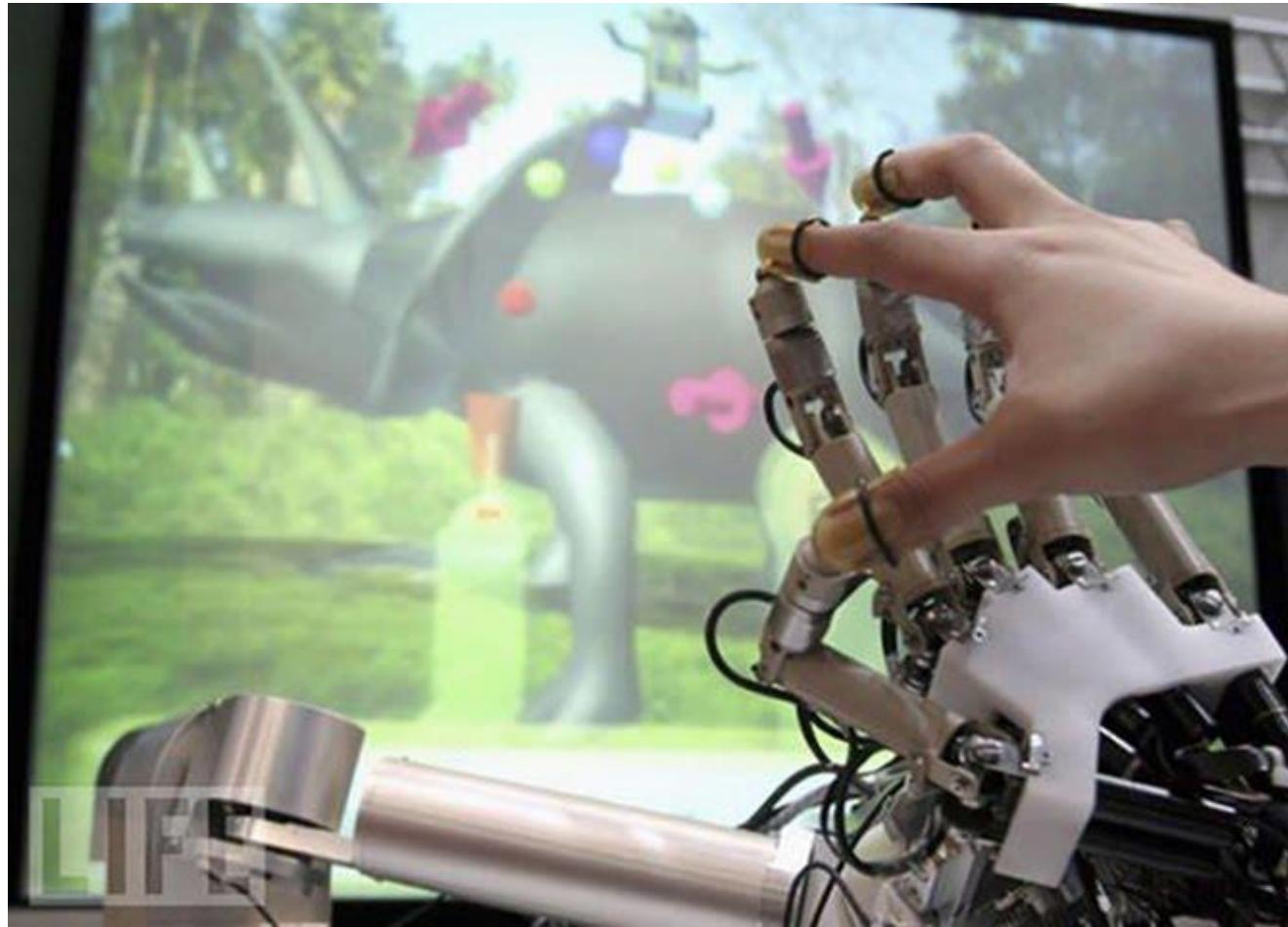
**5**

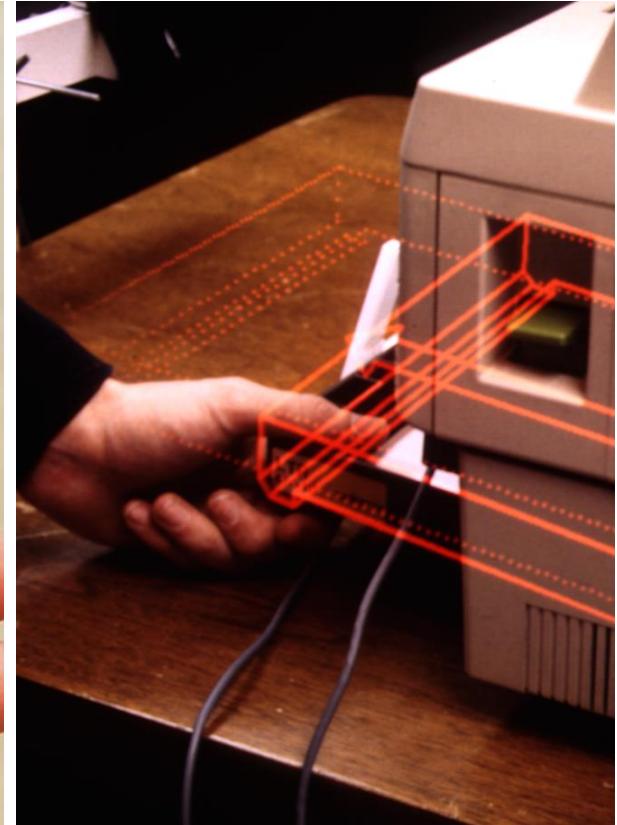
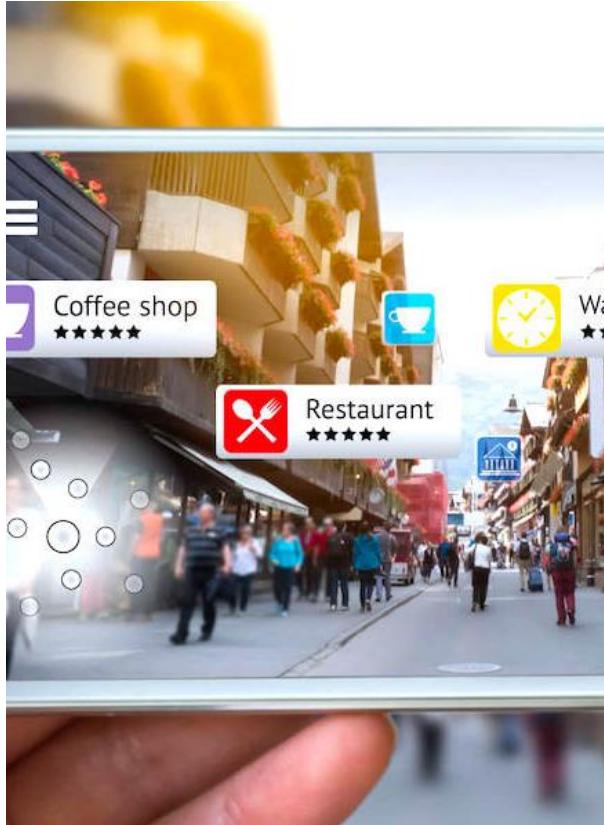
Based on these details, mathematical algorithms calculate the eyes' position and gaze point, for instance on a computer monitor.



## Eye Movement as Input

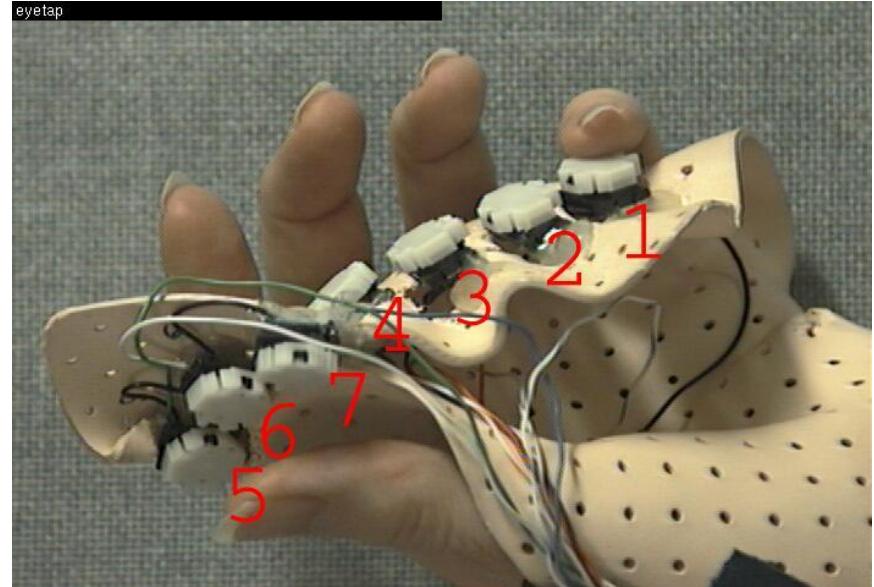
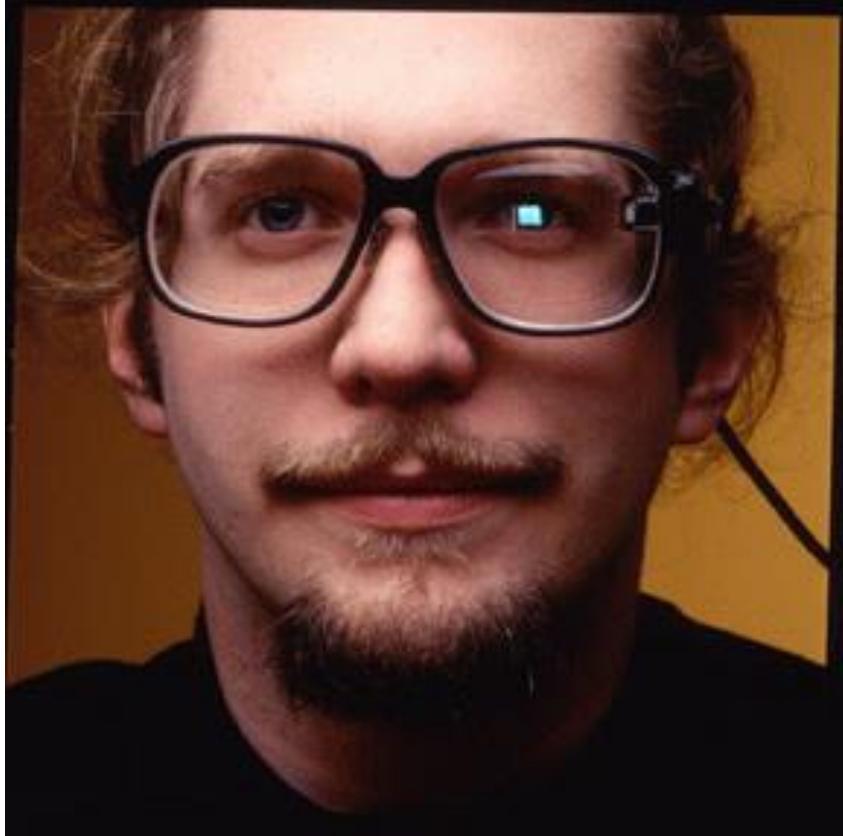
# Haptic Interfaces





# Augmented Reality

---



# Wearable Computing

---

Computation devices accompany you, rather than you seeking them out



“Hey Siri, what’s the best  
sushi place in town?”



## Speech Input

- Human beings have a great and natural mastery of speech
  - makes it difficult to appreciate the complexities
  - but it's an easy medium for communication

# Windows Speech Recognition

- Supplied with every Windows machine
  - From '98 on
  - Almost no one used it
- What was the problem?
  - Need to “train” users to use early virtual assistants (VAs)
  - Microphone expense determines quality
  - No app buy-in.

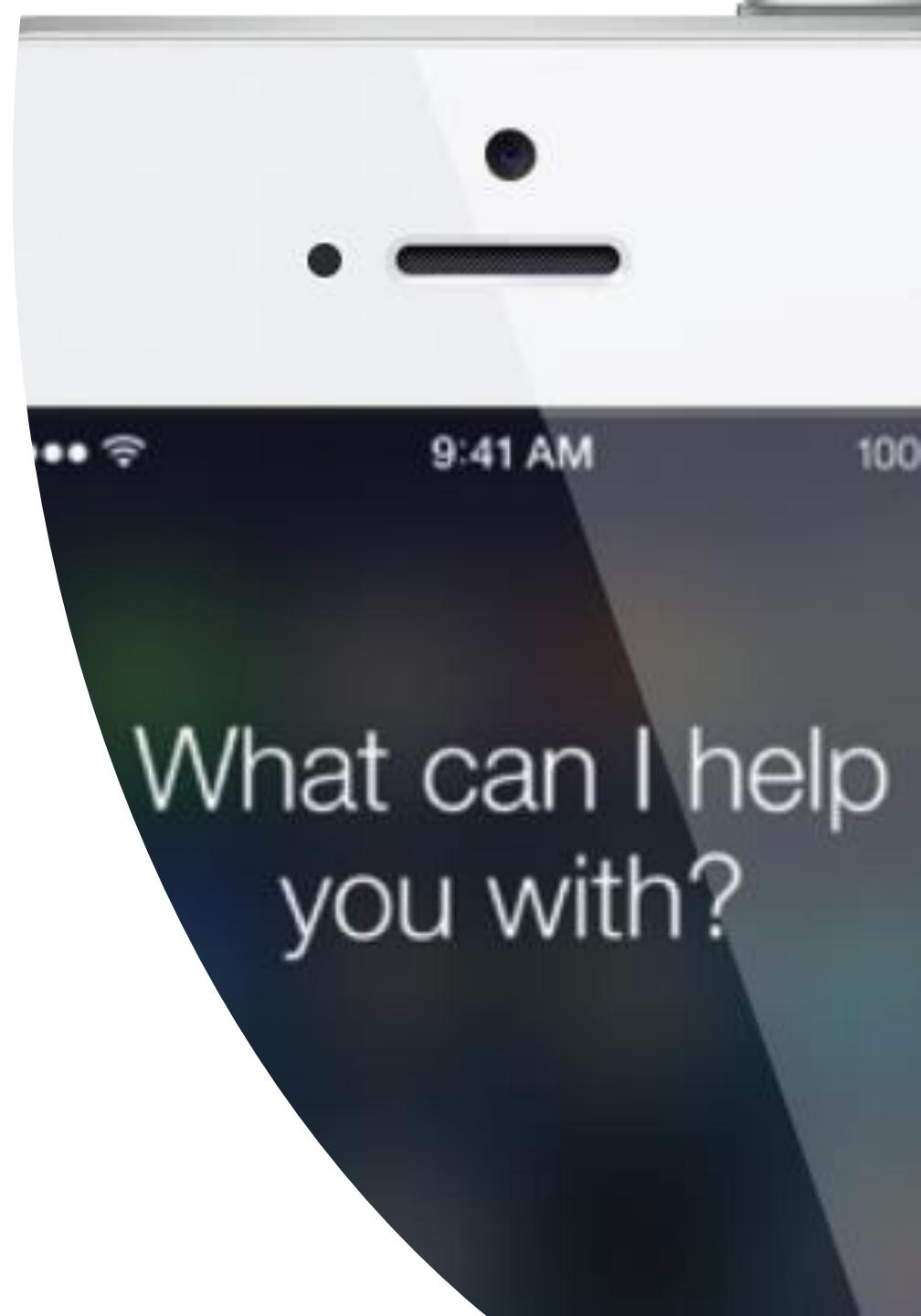


# And Then There Was Siri

---

## A Technical Success

- Consistent microphone gives predictable quality
- Inclusion on every iPhone made it mainstream



# And Then There Was Siri

- Misunderstandings
- Limited skills
- What Apple wants isn't always what users want
- No 3<sup>rd</sup> parties; limited innovation and evolution

“ I need to hide a body ”

What kind of place are you looking for?

reservoirs

metal foundries

mines

dumps

swamps



# Current Incarnations

- What these look like now
  - Specialized hardware
  - Domestic setting
  - Initially aimed at home automation
  - Mostly used for home entertainment
  - All open to 3<sup>rd</sup> parties



# Voice “Explodes” into Mainstream



IBM Watson™



# Seven Design Principles

## 1. **Equitable use**

- same means for all users, do not segregate/stigmatize users, make design appealing

## 2. **Flexibility in use**

- provide choice of methods & adapt to user's pace

## 3. **Simplicity and intuitiveness of use**

- support user's expectations
- accommodate different languages and literacy skills
- provide prompting and feedback

# Seven Design Principles

## 4. Perceptible information

- redundancy of information: use different forms/modes
- emphasize essential information

## 5. Tolerance for error

- minimize impact caused by mistakes
- remove potentially dangerous situations
- hazards should be shielded by warnings

# Seven Design Principles

## **6. Low physical effort**

- comfort; minimize fatigue and effort
- repetitive or sustained actions should be avoided

## **7. Size and space for approach and use**

- placement of system should be reachable by all users
- consider line of sight for standing and sitting user
- allow for variation in hand size
- provide room for assistive devices

# Disabilities

---

- Federal law to ensure access to IT, including computers and web sites.
  - Vision (low vision, blind, color blind)
  - Hearing (deaf, limited hearing)
  - Mobility
  - Learning (dyslexia, attention deficit)

# Disabilities

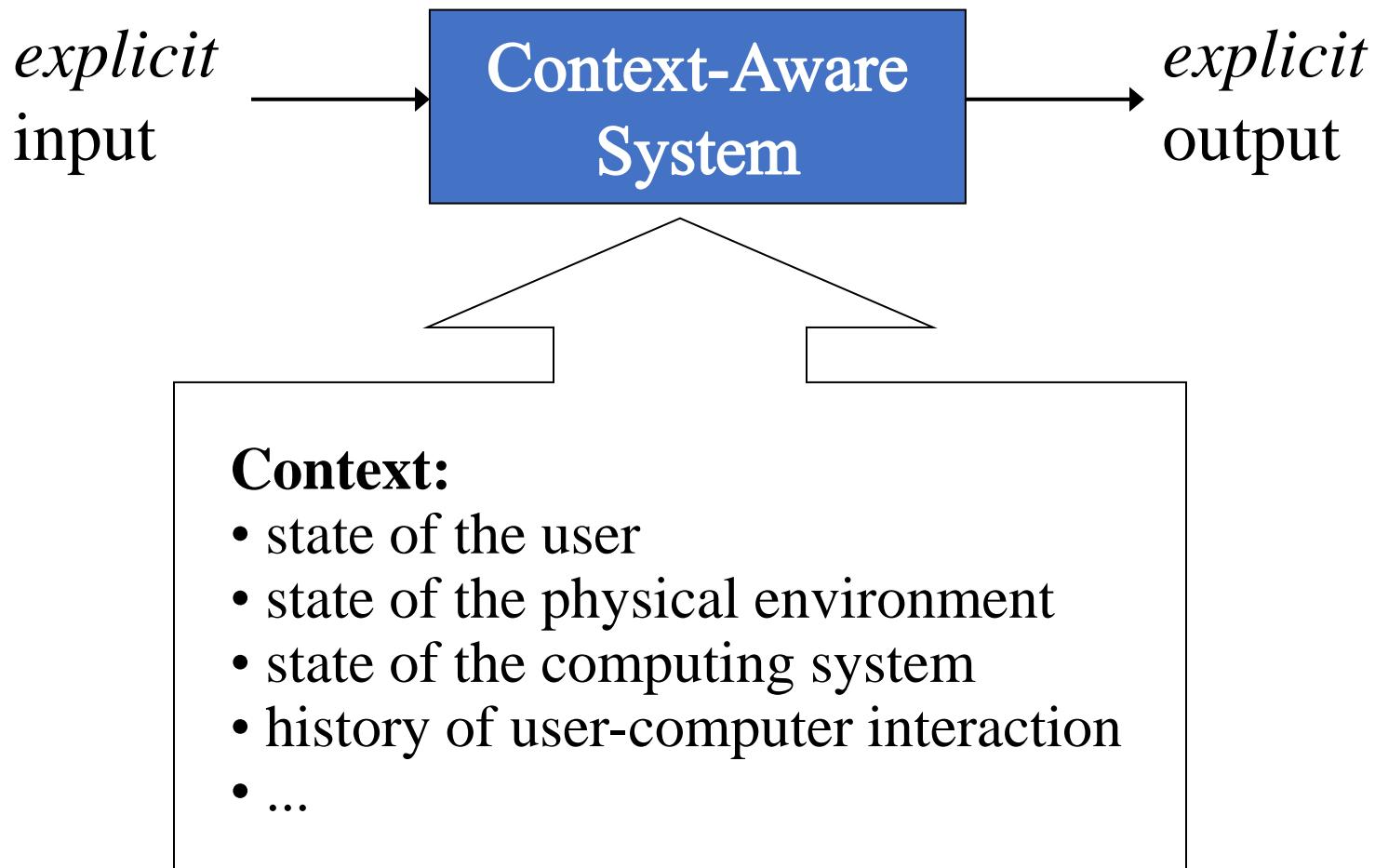
---

- Keyboard and mouse alternatives
- Color coding
- Font size
- Contrast
- Text descriptors for web images
- Magnification
- Text-to-speech; speech recognition
- Head-mounted optical mice
- Eye gaze control

# System Structure



# Context as Implicit Input



# What is Context?



# Examples of Context

- Identity (user, others, objects)
- Location
- Date/Time
- Environment
- Emotional state
- Focus of attention
- Orientation
- User preferences
- Calendar (events)
- Browsing history
- Behavioral patterns
- Relationships (phonebook, call history)
- ... the elements of the user's environment that the computer knows about...

# Relevance of Context Information

- Trying to arrange lunch meeting
- Going to a job interview
- Going home after work and making evening plans
- Shopping
- Tourist
- ...

# Definitions of Context

- “Context is **any information that can be used to characterize the situation of an entity**. An entity is a person, place, or object that is considered **relevant** to the interaction between a user and an application, including the user and applications themselves” [Dey et al. 2001]

# Classification

## External (physical)

- Context that can be measured by hardware sensors
- Examples: location, light, sound, movement, touch, temperature, air pressure, etc.

## Internal (logical)

- Mostly specified by the user or captured monitoring the user's interaction
- Examples: the user's goal, tasks, work context, business processes, the user's emotional state, etc.

# Context?



# Context?



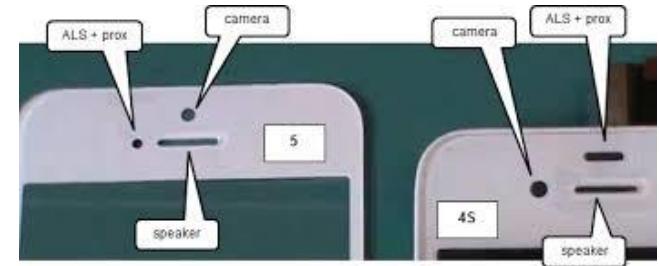
# Simple Everyday Examples

- Smartphone adjusts the screen to the orientation of the device
- Apple Watch turns on display if arm lifted/rotated
- Orientation is determined by using both a gyroscope and an accelerometer



# Simple Everyday Examples

- Phone display adjusts the brightness of the display based on the surrounding area
- Uses a light sensor



# Simple Everyday Examples

- Device displays user's location, shows route to a desired destination, find nearby stores, geotag images on social media, etc.
- Uses location sensor



# Simple Everyday Examples

- The time is displayed on the phone
  - Time zone change
  - Daylight savings time



# Simple Everyday Examples

- Device disables touch screen when the user speaks on the phone
- Uses a proximity sensor (infrared signal travel time)



# Challenges

- Lack of self-awareness
  - Knowing when to do or not to do something is hard
- Complexity
  - More rules do not necessarily yield more intelligence
  - But will become harder to maintain and understand
- Human-in-the-loop vs. automation
  - Loss of control vs. risk of human error
- Development
  - Sensing, aggregation, rules, etc., are complex issues
- Privacy
- User preferences
- Information overload

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn:

<https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# Sensors

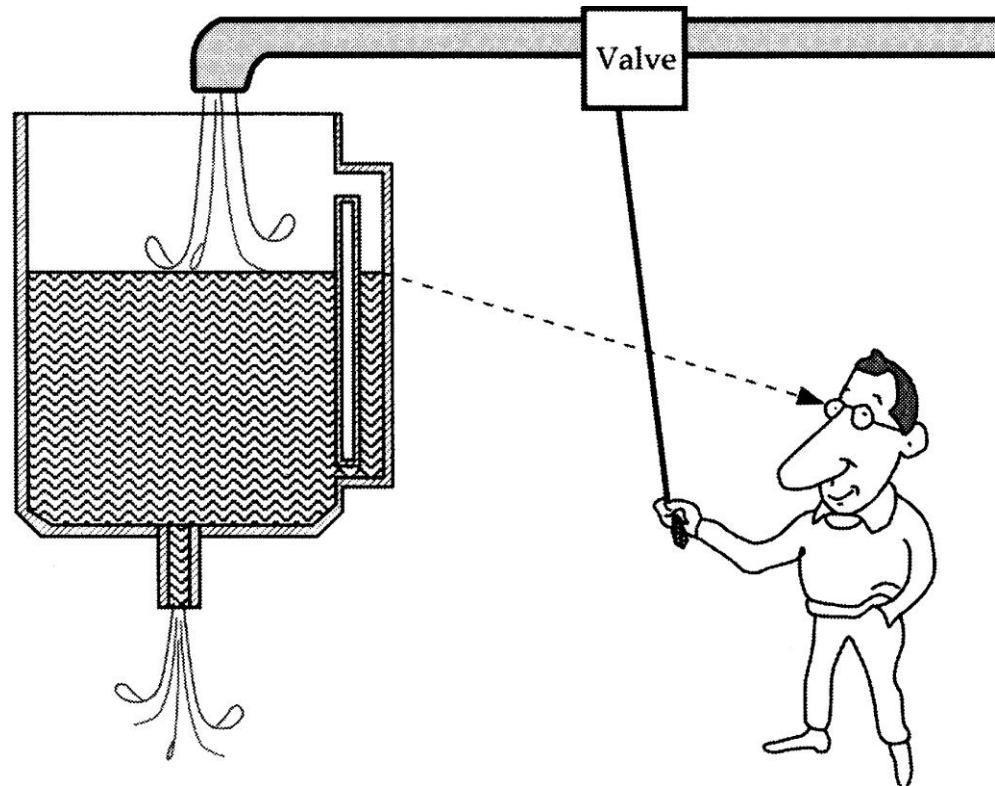
Internet-of-Things (IoT)

---

COCOS20

# Sensors

- A sensor is a device that receives a stimulus and responds with an electrical signal.



# Sensors

Sensors are devices that measure a particular property of the environment and convert that into an electrical output

Typically, sensors have a linear function between the physical property measured and the electrical output

The sensitivity of a sensor indicates the minimum value of the measured input that can produce a certain output signal

An analog-to-digital converter employed to turn a sensor output into signals that can be processed by MCUs

Different physical phenomena underpin the operation of sensors, e.g., the piezoelectric effect (accumulation of electric charge when mechanical pressure is applied)



## Computer-Process Interface

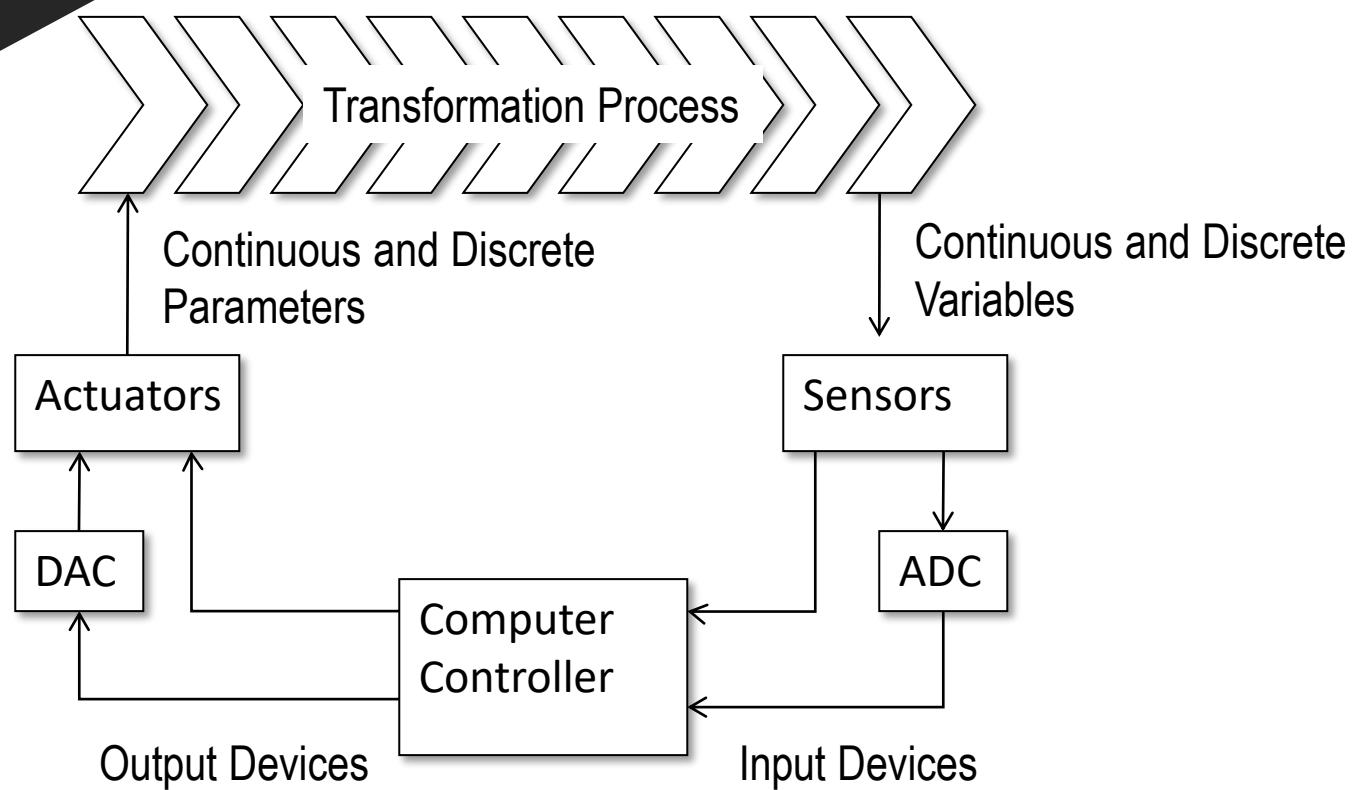
---

- To implement process control, the computer must collect data from and transmit signals to the production process
- Components required to implement the interface:
  - Sensors to measure continuous and discrete process variables
  - Actuators to drive continuous and discrete process parameters
  - Devices for ADC and DAC
  - I/O devices for discrete data

# Need for Sensors

- Sensors are omnipresent. They embedded in our bodies, automobiles, airplanes, cellular telephones, radios, chemical plants, industrial plants and countless other applications.
- Without the use of sensors, there would be no automation!!

# Computer Process Control System



# What is a Stimulus?

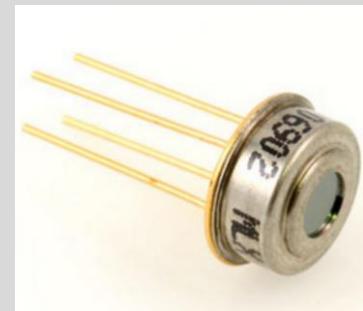
- Motion, position, displacement
- Velocity and acceleration
- Force, strain
- Pressure
- Flow
- Sound
- Moisture
- Light
- Radiation
- Temperature
- Chemical presence



Visual Sensor



Ultrasound Sensor



Infrared Sensor

# Types of sensors

- Push-button/switch



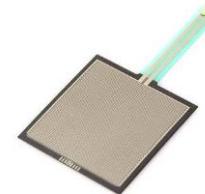
- Temperature



- Acceleration



- Pressure



- Optical/photodiode



- Humidity



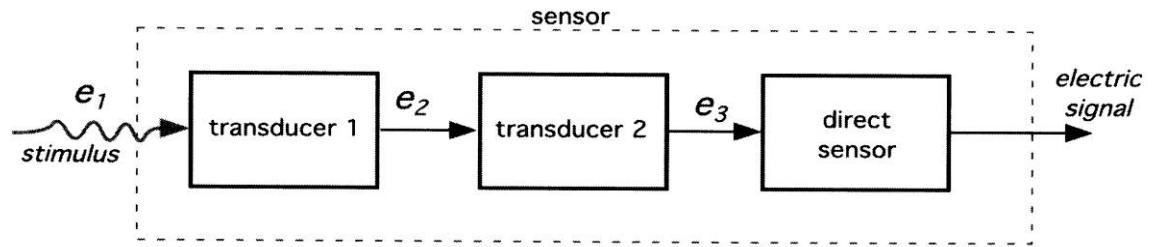
# What is a Response?

When we say electrical, we mean a signal which can be channeled, amplified, and modified by electronic devices:

- **Voltage**
- **Current**
- **Charge**

# Sensor as Energy Converter

- This conversion can be direct or it may require transducers



- Example:

A chemical sensor may have a part which converts the energy of a chemical reaction into heat (transducer) and another part, a thermopile, which converts heat into an electrical signal.

Microphone, Loud Speaker, Biological Senses (e.g. touch, sight,..., etc)

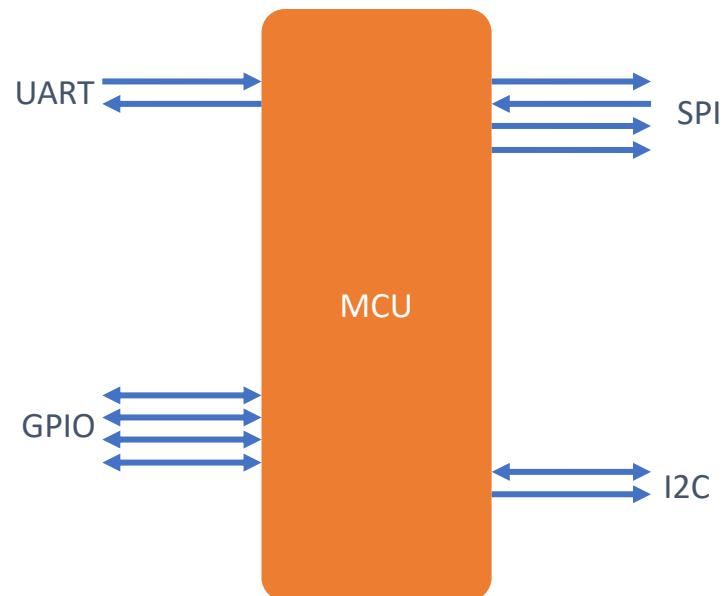
# Physical Principles of Sensing

- Charges, fields, & potentials
- Capacitance
- Magnetism
- Induction
- Resistance
- Piezoelectric effect
- Seebeck and Peltier effects
- Thermal properties of materials
- Heat transfer
- Light

# Input/Output

I/O defines how an MCU can interact with the environment

- Different I/O protocols are available
- Universal Asynchronous Receiver/Transmitter (UART) – two wires to send/receive data between devices
- Serial Peripheral Interface (SPI) – any number of bits can be sent/received without interruption
- Inter-Integrated Circuit (I<sup>2</sup>C) – combines features of UART and SPI
- General purpose I/O (GPIO) – controllable by user at run time.



# GPIO



Can be set up to accept or source different logic voltage levels, through which MCU can control peripherals or receive external input/interrupts.



If pins are configured for interrupts, they can be used to move wake-up from low-power/sleep modes.



Can be grouped into a GPIO port and controlled as such.



Pulse-width modulation (PWM) employed when the linear processes must be controlled (e.g., fans)



Limited to low-current applications → transistors and relays used to help drive higher current loads.

# Pulse-width modulation

Reducing average power output by switching on/off at high rates.

- Duty Cycle – the fraction of one period  $T$  during which a signal is active:

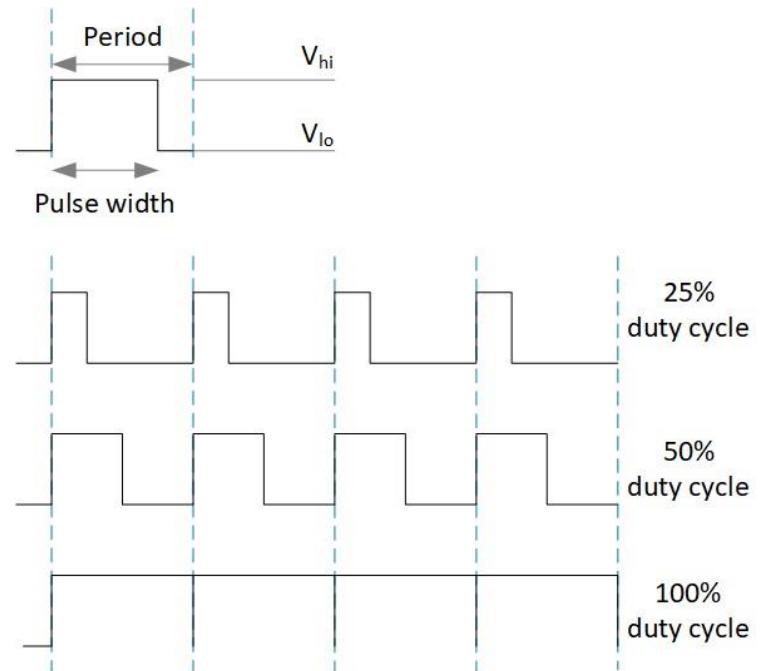
$$D = \frac{PW}{T} \times 100 [\%]$$

- Frequency – rate of periods:

$$f = \frac{1}{T} [\text{Hz}]$$

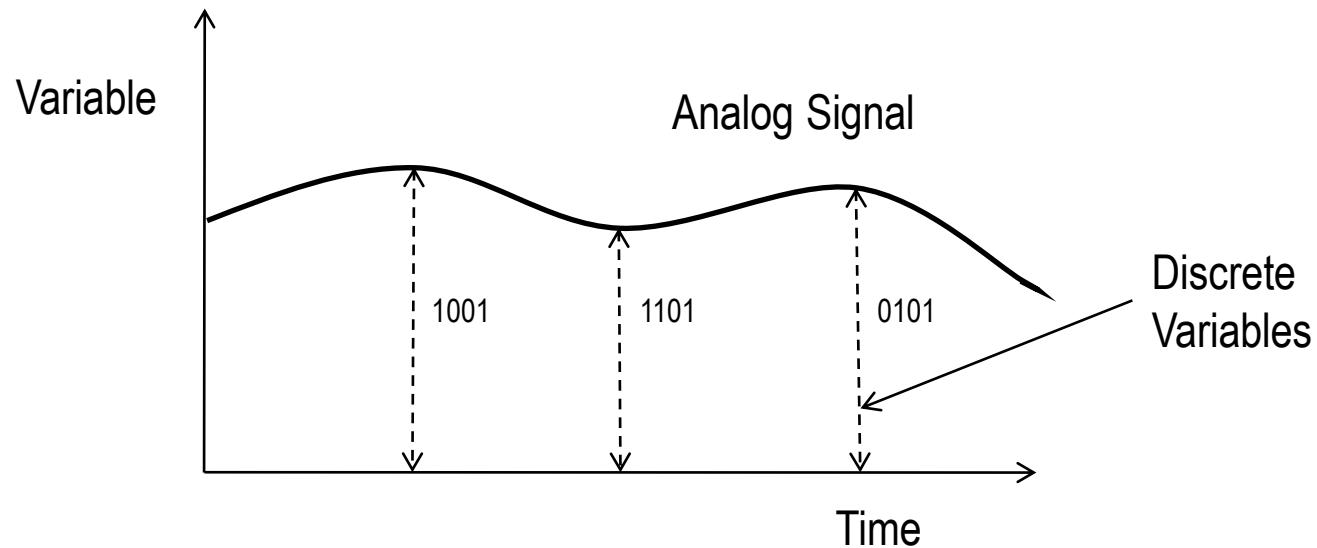
- Average voltage:

$$V_{avg} = V_{hi} \times \frac{D}{100}$$



# Analog-to-Digital Conversion

- **Sampling** – converts the continuous signal into a series of discrete analog signals at periodic intervals
- **Quantization** – each discrete analog is converted into one of a finite number of (previously defined) discrete amplitude levels
- **Encoding** – discrete amplitude levels are converted into digital code



# Features of an ADC

- **Sampling rate** – rate at which continuous analog signal is polled (e.g., 1000 samples/sec)
- **Quantization** – divide analog signal into discrete levels
- **Resolution** – depends on number of quantization levels
- **Conversion time** – how long it takes to convert the sampled signal to digital code
- **Conversion method** – means by which analog signal is encoded into digital equivalent
  - Example: Successive approximation method

# Successive Approximation Method

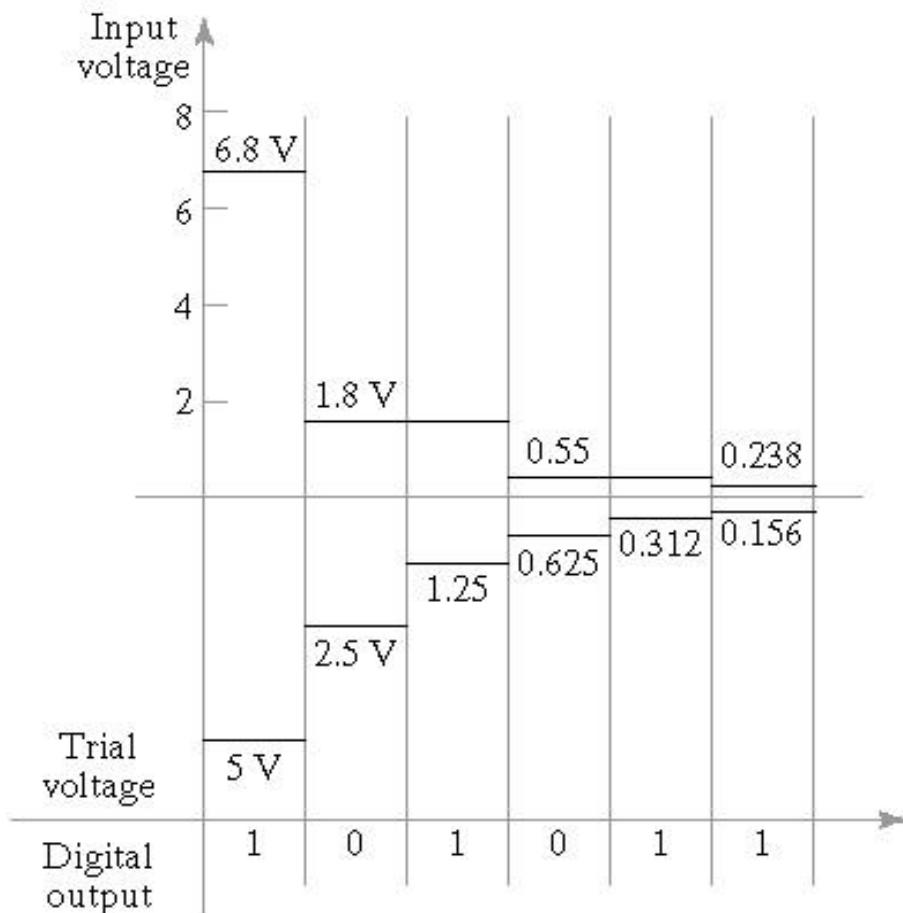
- A successive approximation ADC is a type of ADC that converts a continuous analog waveform into a discrete digital representation via a binary search through all possible quantization levels before finally converging upon a digital output for each conversion

# Algorithm

- A series of trial voltages are successively compared to the input signal whose value is unknown
- Number of trial voltages = number of bits used to encode the signal
- First trial voltage is  $1/2$  the full scale range of the ADC
- If the remainder of the input voltage exceeds the trial voltage, then a bit value of 1 is entered, if less than trial voltage then a bit value of zero is entered
- The successive bit values, multiplied by their respective trial voltages and added, becomes the encoded value of the input signal

# Example

- Analogue signal is 6.8 volts. Encode, using SAM, the signal for a 6 bit register with a full scale range of 10 volts.



For six digit precision,  
the resulting binary  
digital value is 101011,  
which is interrupted as:

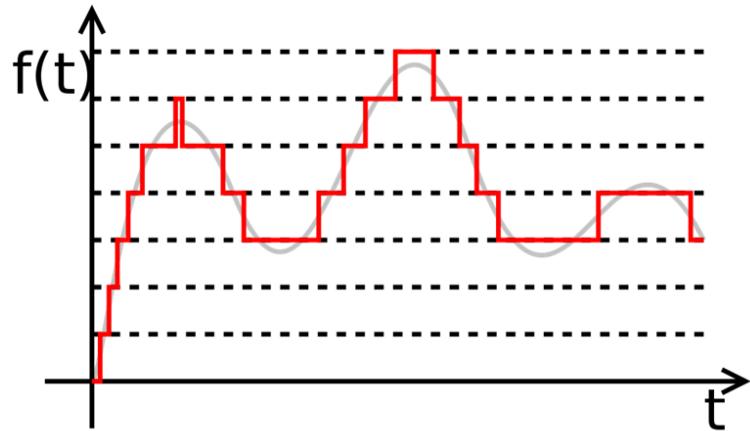
$$\begin{aligned} &1 \times 5.0 \text{ V} \\ &0 \times 2.5 \text{ V} \\ &1 \times 1.25 \text{ V} \\ &0 \times 0.625 \text{ V} \\ &1 \times 0.312 \text{ V} \\ &1 \times 0.156 \text{ V} \end{aligned}$$

$$\text{Total} = 6.718 \text{ V}$$

# Example

$$Q = \frac{E_{FSR}}{2^M} = \frac{E_{FSR}}{N}$$

- Q = resolution in volts per step
- M = resolution in bits
- N = Number of intervals (steps, quantization levels)
- $E_{FSR}$  = Full scale voltage range
- Quantization error =  $\frac{1}{2}$  of interval
  
- Voltage range 0 – 10V; M = 12 bits
- N = 4096 intervals (steps)
- Q = 2.44 mV/code



## Example

- Using an analogue-to-digital converter, a continuous voltage signal is to be converted into its digital counterpart. The ADC has a 16-bit capacity, and full scale range of 60 V. Determine:
  - number of quantization levels
  - resolution
  - quantization error

# Solution

(1) Number of quantization levels:

$$= 2^{16} = 65,536$$

(2) Resolution:

$$= 60 / 65,536 = \sim 0.00092 \text{ Volts}$$

(3) Quantization error:

$$= 0.00092/2 = 0.00046 \text{ Volts}$$

# Digital-to-Analog Conversion

- Convert digital values into continuous analogue signal
  - Decoding digital value to an analogue value at discrete moments in time based on value within register

$$E_0 = E_{ref} \left\{ 0.5B_1 + 0.25B_2 + \dots + (2^n)^{-1} B_n \right\}$$

Where  $E_0$  is output voltage;  $E_{ref}$  is reference voltage;  $B_n$  is status of successive bits in the binary register

# Example

- A DAC has a reference voltage of 100 V and has 6-bit precision. Three successive sampling instances 0.5 sec apart have the following data in the data register:

Instant	Binary Data
1	101000
2	101010
3	101101

- Output Values:

$$E_{01} = 100\{0.5(1)+0.25(0)+0.125(1)+0.0625(0)+0.03125(0)+0.015625(0)\}$$

$$E_{01} = 62.50V$$

$$E_{02} = 100\{0.5(1)+0.25(0)+0.125(1)+0.0625(0)+0.03125(0)+0.015625(0)\}$$

$$E_{02} = 65.63V$$

$$E_{03} = 100\{0.5(1)+0.25(0)+0.125(1)+0.0625(0)+0.03125(0)+0.015625(0)\}$$

$$E_{03} = 70.31V$$

# Sensor Types: HW & SW

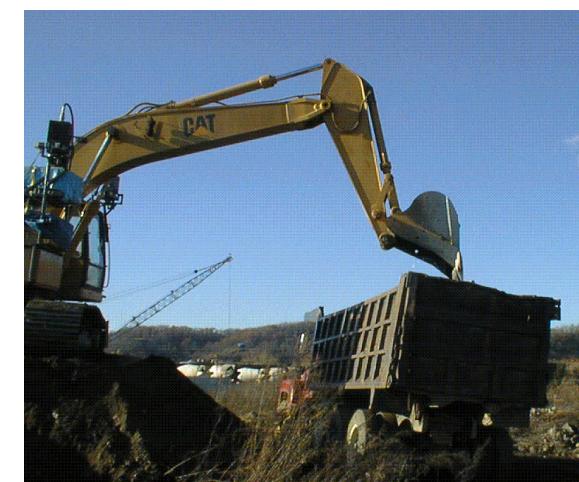
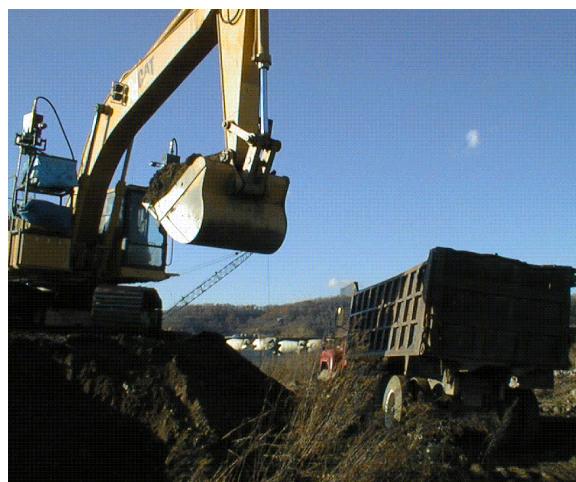
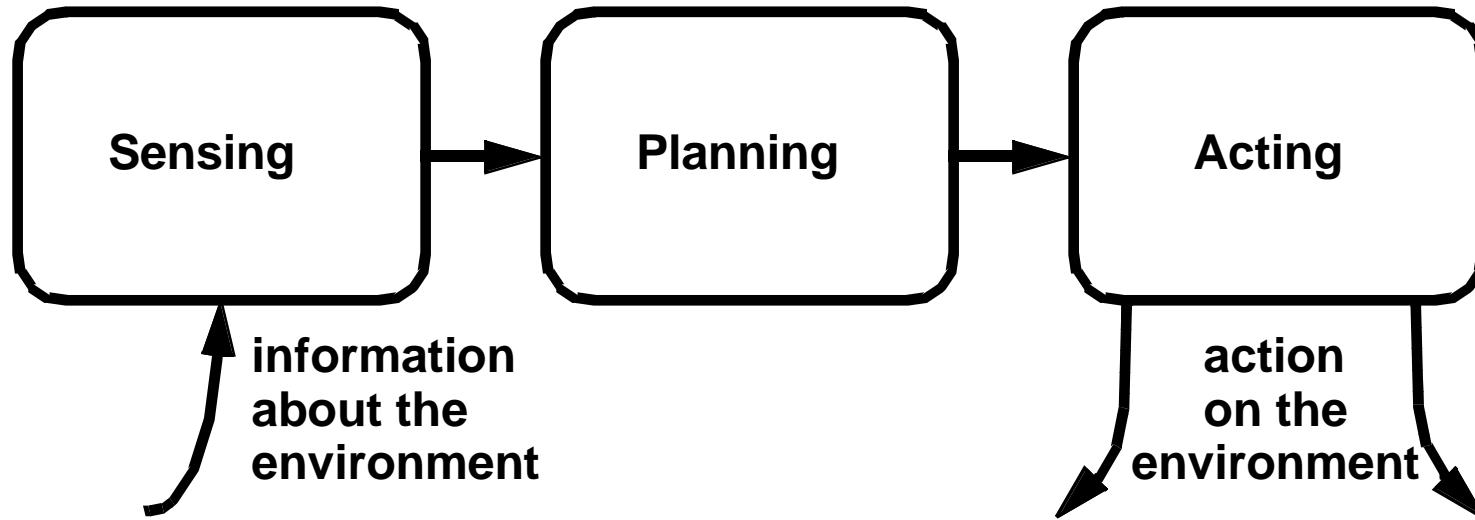
- **Hardware-based sensors**
  - Physical components built into a device
  - They derive their data by directly measuring specific environmental properties
- **Software-based sensors**
  - Not physical devices, although they mimic hardware-based sensors
  - They derive their data from one or more hardware-based sensors

# Sensor List of Smartphone

Sensor	Function Type	Software-based or Hardware-based
Accelerometer	Motion Sensor	Hardware-based
Gyroscope	Motion Sensor	Hardware-based
Gravity	Motion Sensor	Software-based
Rotation Vector	Motion Sensor	Software-based
Magnetic Field	Position Sensor	Hardware-based
Proximity	Position Sensor	Hardware-based
GPS	Position Sensor	Hardware-based
Orientation	Position Sensor	Software-based
Light	Environmental Sensor	Hardware-based
Thermometer	Environmental Sensor	Hardware-based
Barometer	Environmental Sensor	Hardware-based
Humidity	Environmental Sensor	Hardware-based

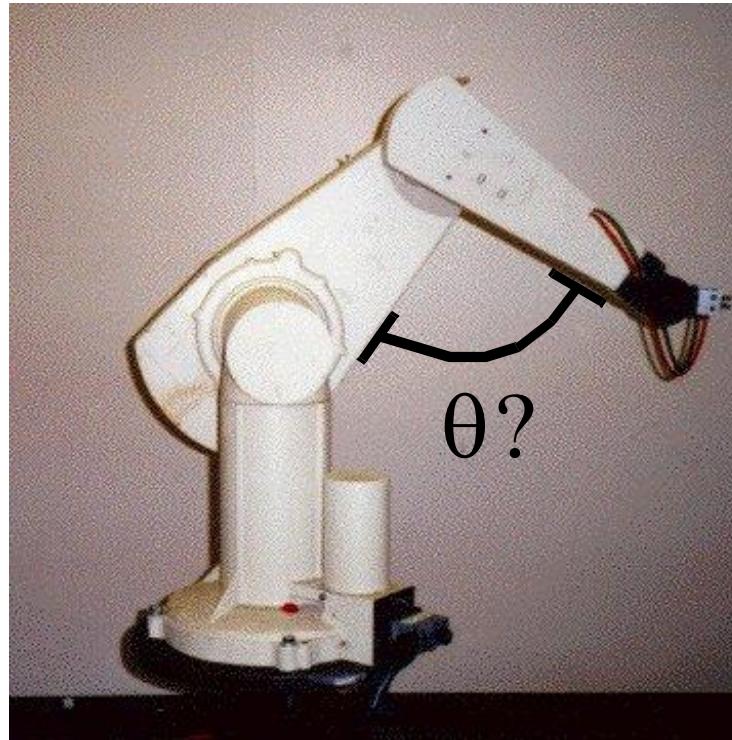
# **Overview of Our Sensors For Robotics**

# What makes a machine a robot?



Why do robots need sensors?

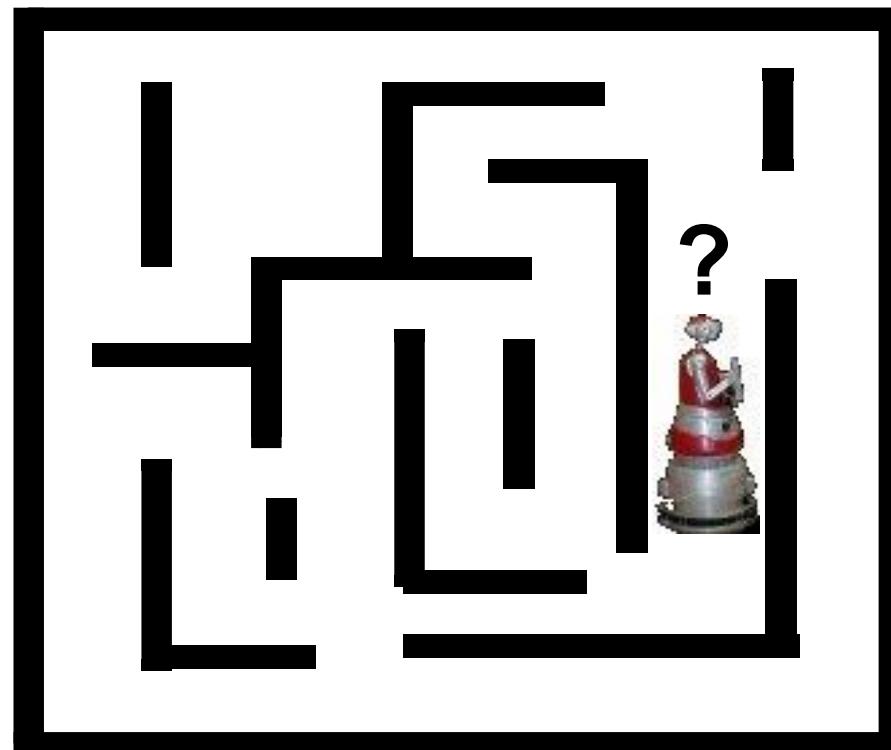
**What is the angle of my arm?**



**internal information**

# Why do robots need sensors?

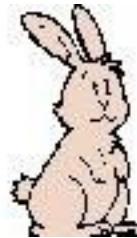
**Where am I?**



**localization**

# Why do robots need sensors?

**Will I hit anything?**



**obstacle detection**

Sensing for specific tasks

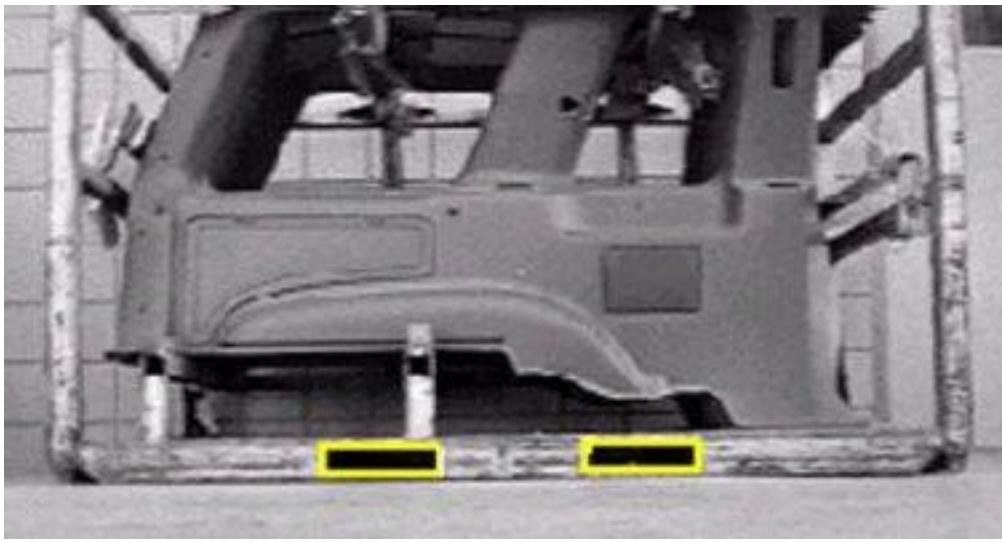
**Where is the cropline?**



**Autonomous  
harvesting**

Sensing for specific tasks

**Where are the forkholes?**



**Autonomous material handling**

Sensing for specific tasks

**Where is the face?**



**Face detection & tracking**

# Types of Sensors

- Active

- send signal into environment and measure interaction of signal w/ environment
- e.g. radar, sonar

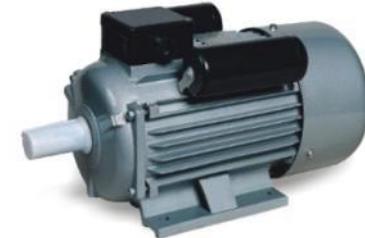
- Passive

- record signals already present in environment
- e.g. video cameras

# Actuators

- Hardware devices that convert a controller command signal into a change in a physical parameter
  - The change is usually mechanical (e.g., position or velocity)
  - An actuator is also a transducer because it changes one type of physical quantity into some alternative form
  - An actuator is usually activated by a low-level command signal, so an amplifier may be required to provide sufficient power to drive the actuator

# Types of Actuators



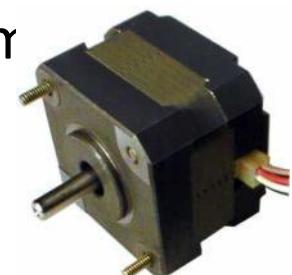
## 1. Electrical actuators

- Electric motors
  - DC servomotors
  - AC motors
  - Stepper motors
- Solenoids



## 2. Hydraulic actuators

- Use hydraulic fluid to amplify the controller command signal



## 3. Pneumatic actuators

- Use compressed air as the driving force



Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn:

<https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# IoT System Architectures and Standards

COCSC20

# Syllabus

This module will cover the following aspects

- Key considerations for IoT architectures
- Cloud, fog, and edge paradigms
- The role of gateways in IoT
- IoT internetworking approaches
- Standards that enable practical IoT deployment and interoperability

# The IoT architectural landscape



- Thousands of new applications exist, spanning countless domains (verticals).
- Each application has its unique requirements → combining these leads to systems that are complex, difficult to manage, and often proprietary.
- Defining a unified architecture is challenging and interoperability problematic, if there are too many standards to choose from.
- Efforts by multiple entities to define common frameworks, including international standardization bodies, multi-national collaborative research projects, industry consortiums, and large commercial actors.
- Device/protocol documentation is scattered and often difficult to navigate.
- We will focus on the key principles different architectural patterns share and examine some examples.

# Key considerations for IoT architectures



What application domains should be covered?



Where to place the “intelligence”?



What networking structure should be employed?



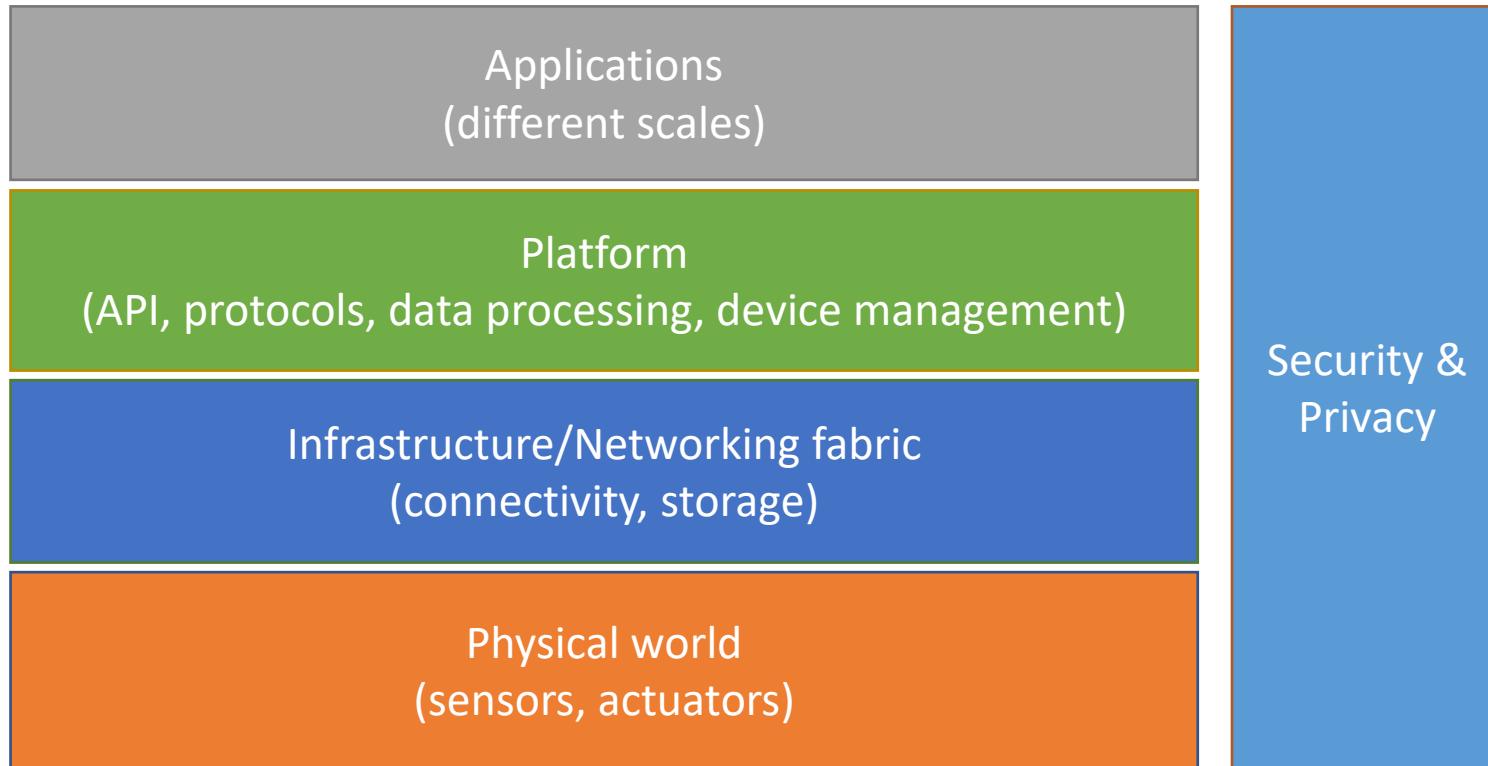
How to modularize systems, so as to manage complexity and enable programmability?



What are the cost and scalability implications?

# A layered view to IoT architecture

At a high level, stakeholders may converge to a shared vision



This approach enables to break up complexity, share resources more easily, and promote interoperability

# Advantages of the layer approach

- Allows IoT device manufacturers to focus strictly on improving their performance, power consumption, etc. – expose only well-defined interfaces to software platforms.
- Easier to share and partition strictly the network and computing resources (slicing); reducing the burden on service providers to build and manage networks – Infrastructure/Network as a Service (IaaS/NaaS)
- Enables software/app developers to build applications without having to understand the specifics of a device – Platform as a Service (PaaS)

# Security challenges

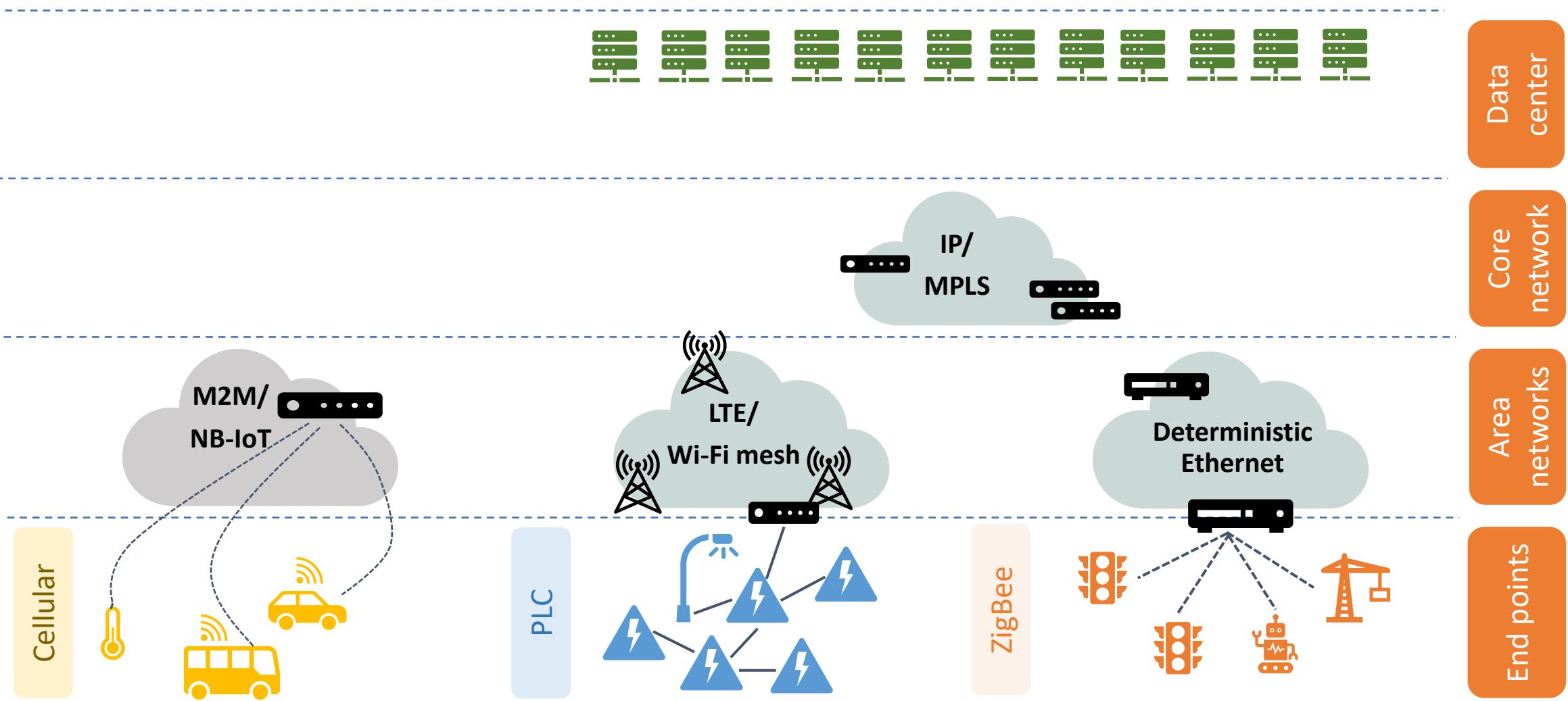
How to secure the entire ecosystems, from hardware to application?

- Hardware isolation (Arm TrustZone)
- Middleware (Speculative Store Bypass Barrier – SSBB)
- Network isolation (Software-defined Networking – SDN)
- Data confidentiality in transit (Transport Layer Security – TLS)
- Software isolation (containers)



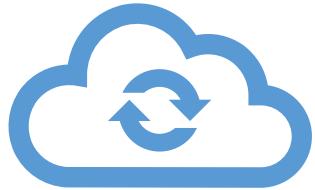
End-to-end security  
not straightforward

# A practical network-centric view



# Cloud vs. Fog vs. Edge

The information processing view



## Cloud computing



Cloud dominated the networked systems landscape until recently



All intelligence on powerful servers, including relational databases, control functions, data analytics engines, web interfaces, etc.

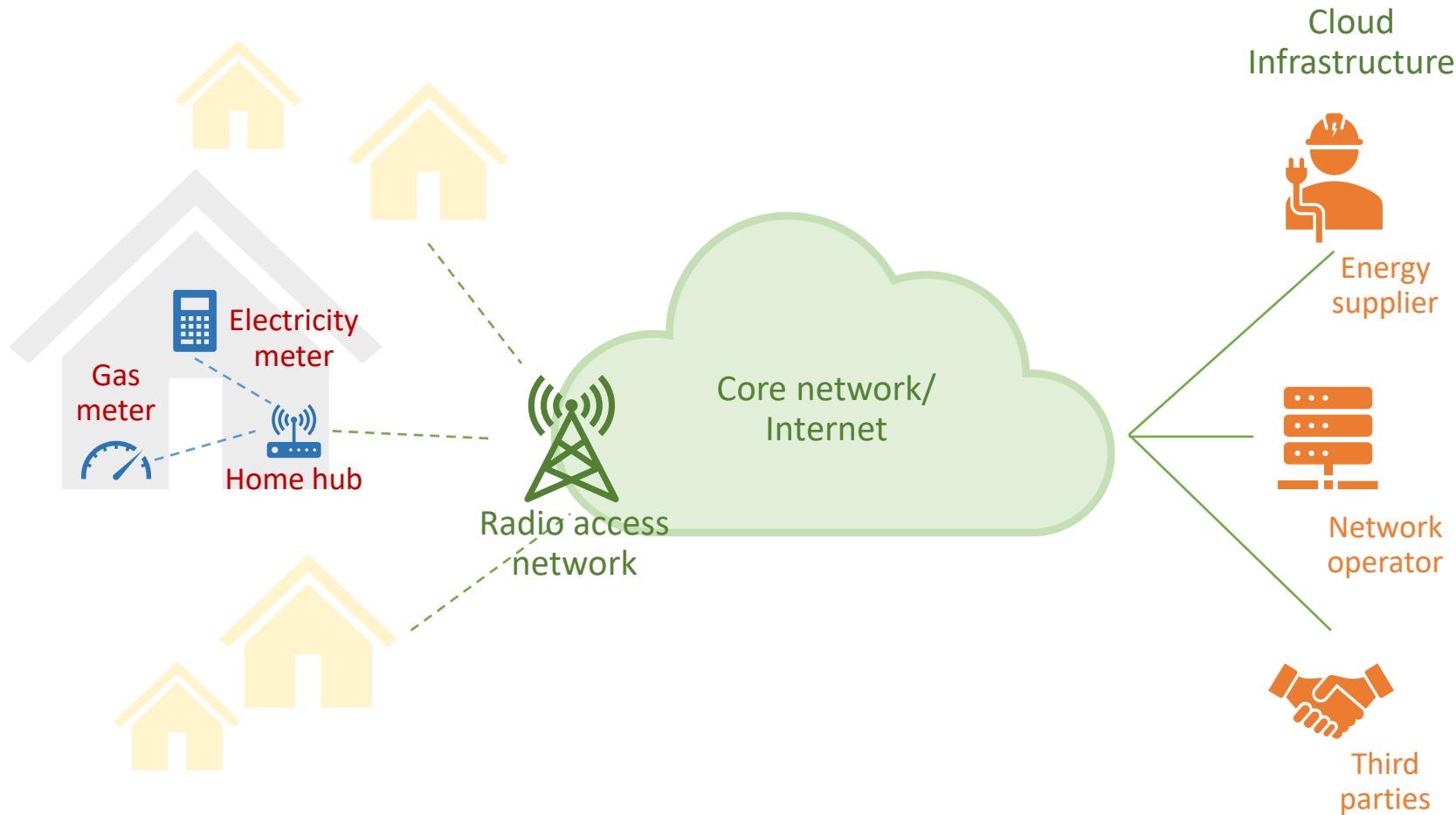


End devices merely information gatherers



Might not scale as the number of IoT devices grows, and applications continue to diversify and generate more data

# Example: Smart metering



- Sensing performed by simple sensors
- Information relayed by home hub over cellular network
- Data processed in the cloud by different stakeholders

# Cloud vs. Fog vs. Edge

The information processing view



Pushing some of the intelligence closer to the device, for e.g., to access networks or gateways



This includes data aggregation, compression, (partial) processing, making localized decisions



IoT devices kept simple, no direct communication with end servers, still battery powered



Resource management implemented across different network layers – management could be regarded as an application



**Fog computing**

# The role of gateways in fog architectures



Data filtering and processing (for e.g., aggregation of summaries, compression, etc.)



Protocol translation and interfacing between different connectivity technologies



Data flow multiplexing, packet routing



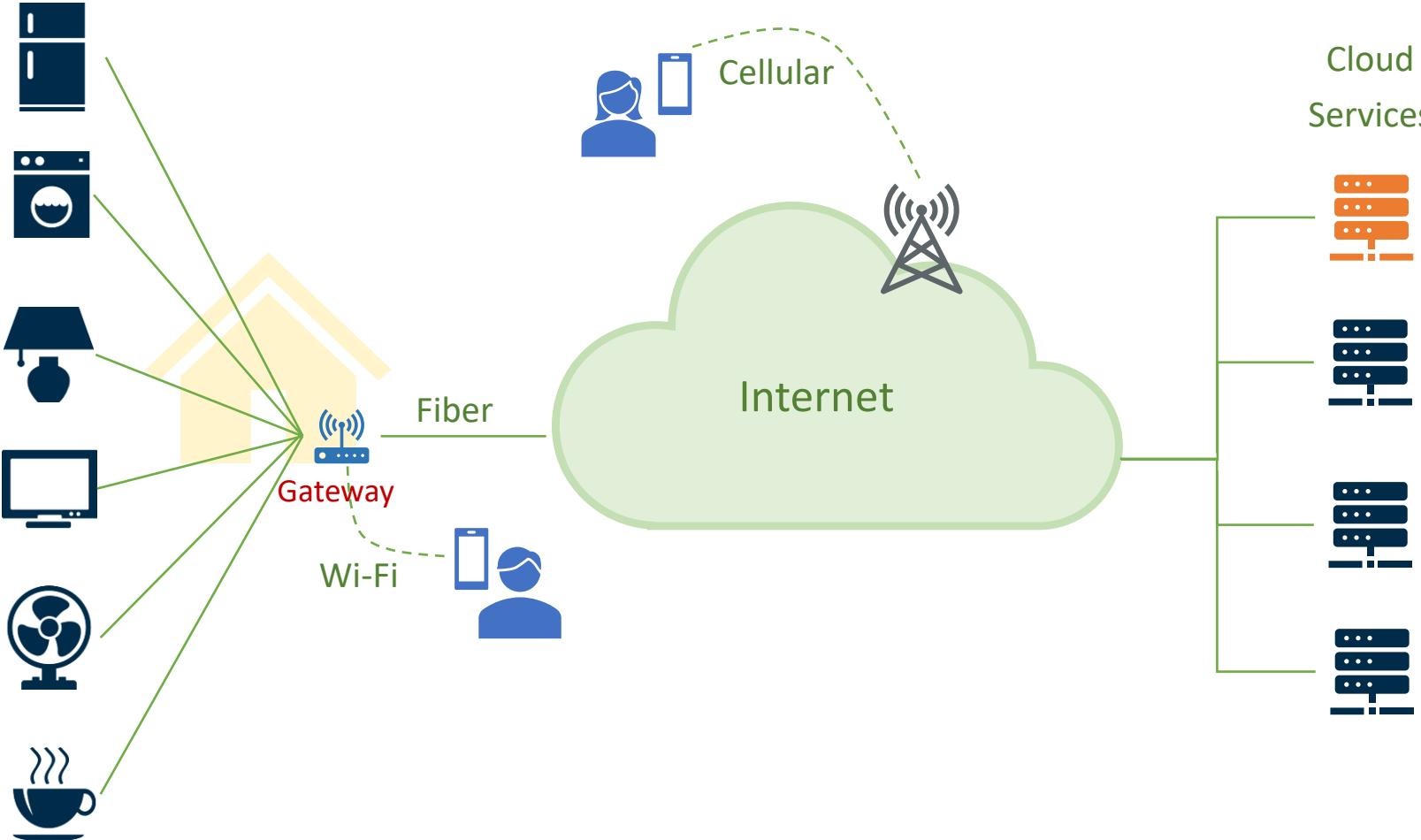
Security (for e.g., data encryption, firewalling)



Scalability problem: as the number of devices grows, so will the number of gateways that are required

# Example: Home automation

## Home appliances



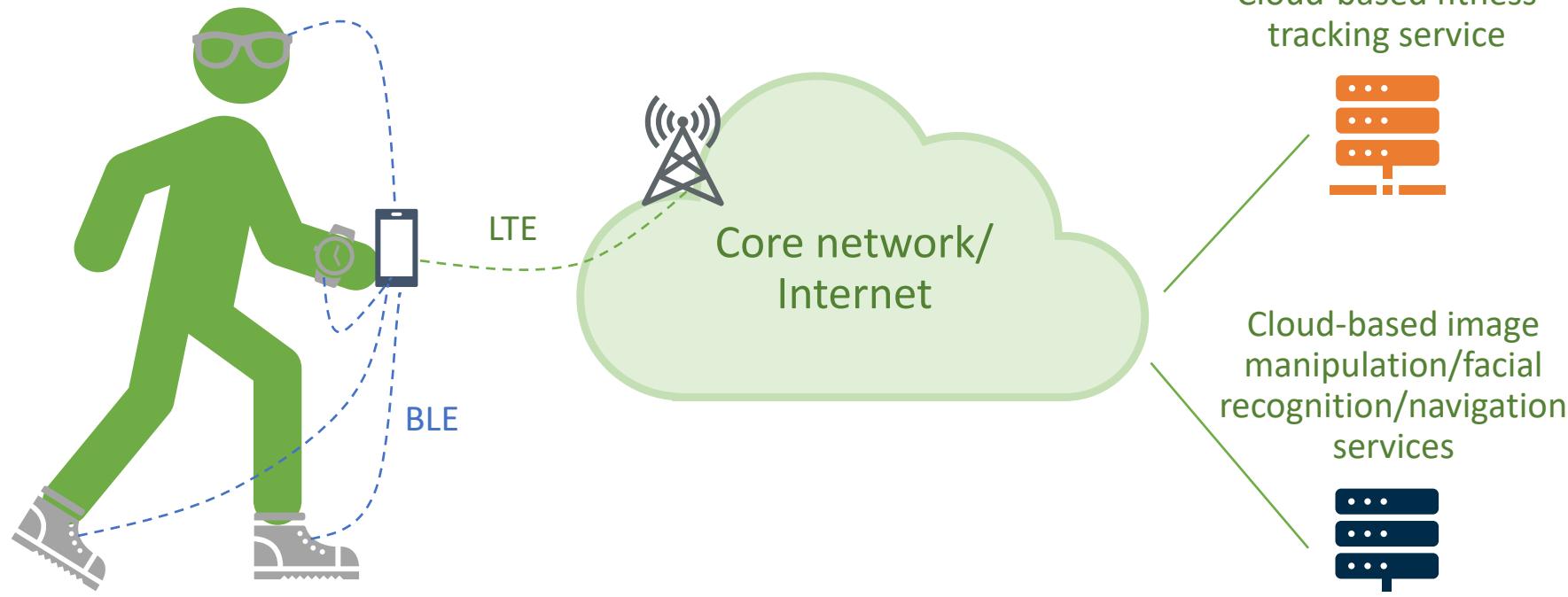
- Gateway performs protocol translation
- Incorporates basic network intrusion detection system
- Cloud services continue to perform analytics

# Smartphones as gateways

The fitness and healthcare domain

- Embed multiple networking technologies (Wi-Fi, 3G/4G, Bluetooth/BLE, NFC, etc.)
- Run full TCP/IP stacks, thus maintain end-to-end connectivity with cloud
- Can connect to multiple devices within close proximity simultaneously
- Ability to enforce secure transport (e.g., TLS/HTTPS)
- Sufficient computing power to pre-process/augment collected data

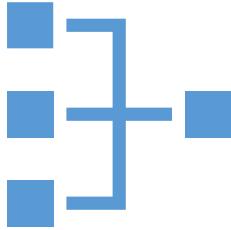
# Example: Wearables



- Smartphone communicates over BLE with wearable devices
- Performs minimal information pre-processing
- Relays data to cloud-based services

# Cloud vs. Fog vs. Edge

The information processing view



## Edge computing



Pushing compute power, communication capabilities, intelligence down at device level



Processing as much as possible where data is collected

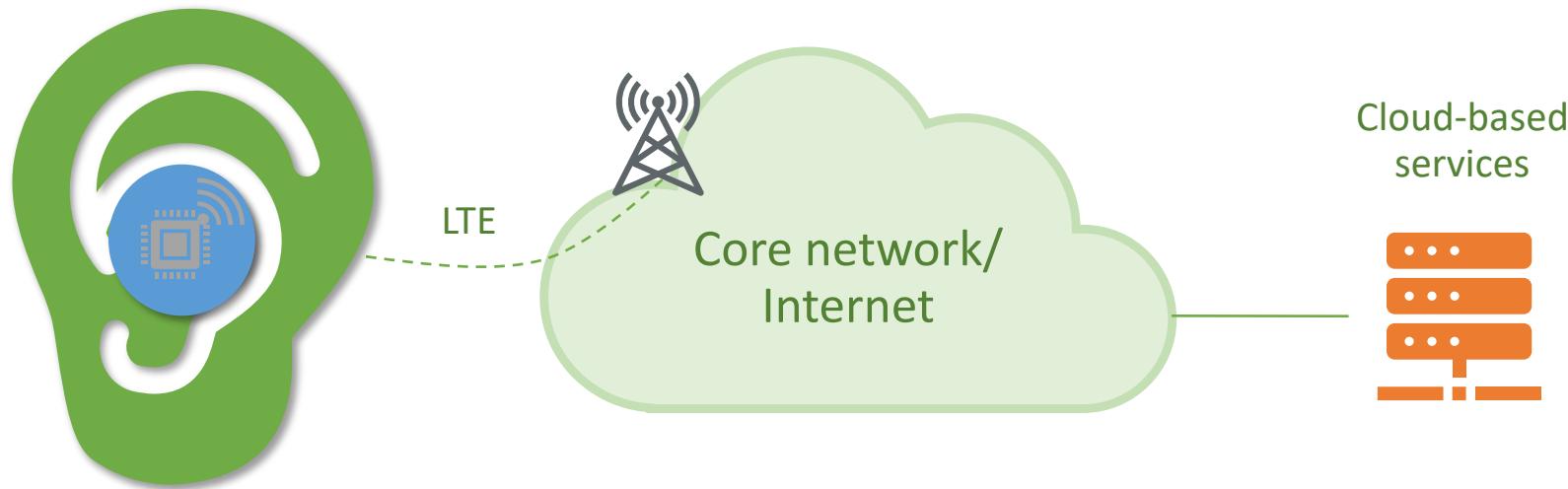


Transmitting only key information or summaries



Enabling new applications: automotive IoT, virtual/augmented reality, in-ear computing

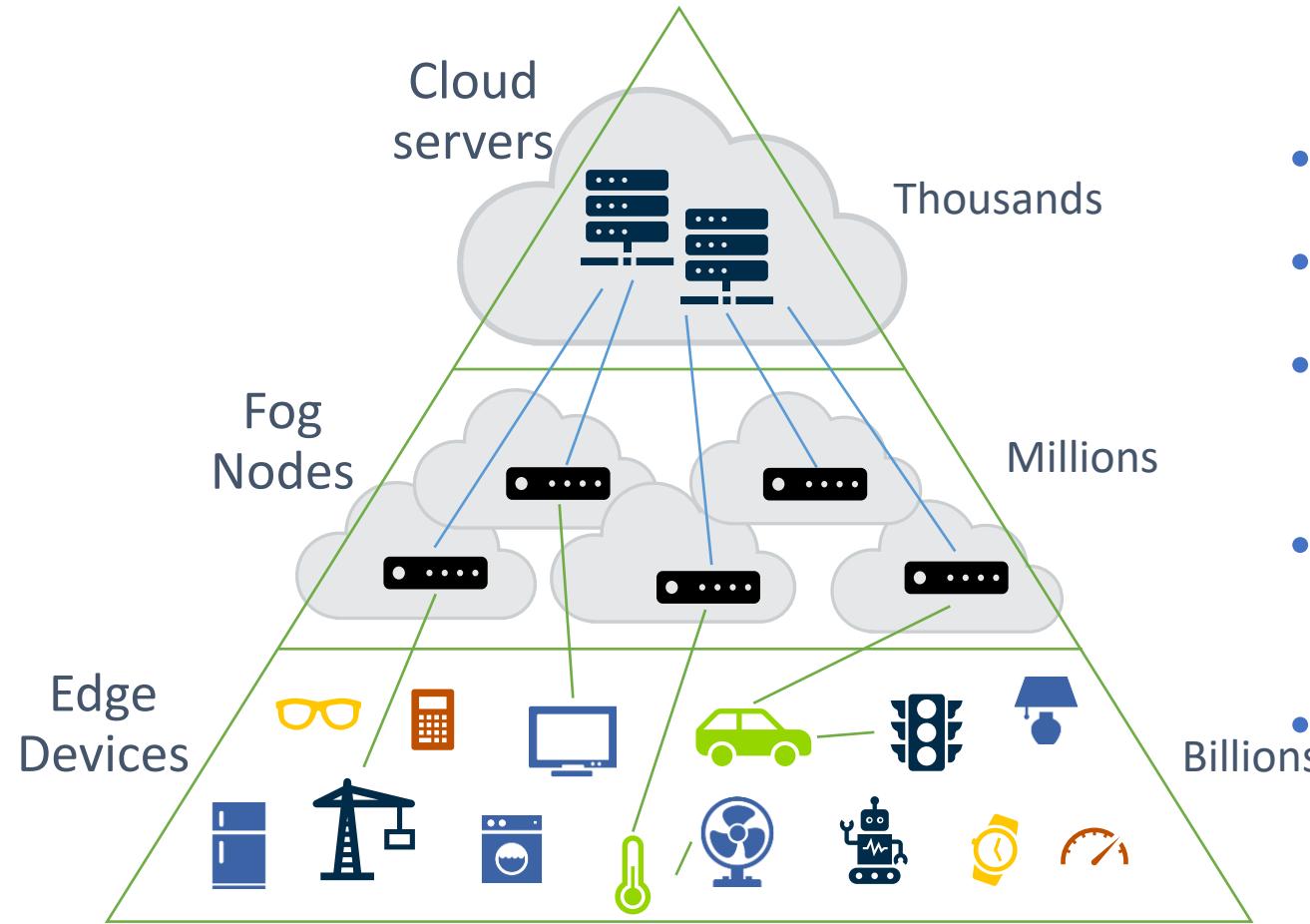
# Example: Hearables



- **Hardware:** Low-power chips specialized in computationally intensive tasks (Arm Ethos)
- **Software:** AI libraries optimized for constrained devices (uTensor)
- **Neural networks:** compressed/pruned models

# Choosing the right IoT architecture

Performance and cost remain the dominant architectural drivers



- What are the application requirements?
- What data needs to be acted on locally?
- Where is most of the computing power?
- How much networking infrastructure should be deployed/used?
- Where are the trust boundaries?

# Standards for IoT

Multiple regulation bodies and industry alliances are standardizing the means by which devices can interact with each other and with gateways or cloud services.

## The Institute of Electrical and Electronics Engineers (IEEE)

- Primarily dealing with defining protocols for (wireless) access networks
- Targeting the Industrial, Scientific, and Medical (ISM) bands (e.g., 2.4GHz, 5GHz, 900MHz in some regions, etc.)
- From an IoT perspective, the most relevant technologies include
  - IEEE 802.15.4, on which ZigBee builds, and
  - IEEE 802.11ah (HaLow) that is an amendment to the IEEE 802.11 specification (typically used for Wi-Fi) that enables low-power wide-area networking

# Standards for IoT

Multiple regulation bodies and industry alliances are standardizing the means by which devices can interact with each other and with gateways or cloud services.

## The 3<sup>rd</sup> Generation Partnership Project (3GPP)

- Focuses on specifying cellular network architectures and protocols (e.g., GSM, 3G, 4G-LTE, etc.)
- Developing standards for cellular communications tailored to IoT applications
  - LTE-M – compatible with existing LTE networks, easy to roll out, limited to 1Mb/s speeds
  - NB-IoT – deployed in same or different frequency bands, lower capacity (200Kb/s), different modulation and coding schemes, and does not require gateways.

# Standards for IoT

## The Internet Engineering Task Force (IETF)

- Focuses on specifying protocols that are used across the Internet; these standards are known as Requests for Comments (RFCs)
- IoT relevant standards include
  - Addressing/internetworking for low power devices (IPv6 over Low-Power Wireless Personal Area Networks – 6LoWPAN)
  - Routing (Routing Over Low-power and Lossy networks – ROLL)
  - End-to-end communications (Constrained Application Protocol – CoAP)
  - Security (Datagram Transport Layer Security – DTLS)
  - Software updating (Software Updates for Internet of Things – SUIT)
- Also offers experience-based guidance
  - Example: The JavaScript Object Notation (JSON) Data Interchange Format – RFC 8259

# Standards for IoT

## Industry alliances

- Bluetooth – wireless personal area networks (WPANs); defines application profiles
- ZigBee – WPANs building on IEEE 802.15.4; inexpensive consumer/industrial applications
- LoRaWAN – LPWAN based on chirp spread spectrum technology

## Collaborative associations

- The Alliance for IoT Innovation (AIoTI) – European Commission framework supporting interaction between IoT players to drive innovation, standardization, and policy.
- Open Connectivity Foundation (OCF) – Industry-led framework aiming to develop IoT standards, interoperability guidelines, and provide a device certification program.

# Standards for IoT

## Other standardization bodies relevant to IoT

- National Institute of Standards and Technology (NIST) – works on a range of science, technology, and engineering topics
  - Example: Advanced Encryption Standard (AES)
- International Organization for Standardization (ISO) – promotes a broad range of proprietary, industrial, and commercial standards
  - Example: Internet of Things (IoT) – Reference Architecture (ISO/IEC 30141:2018)
- International Telecommunication Union (ITU) – recommendations, reference models
  - Example: ITU-T Y.4000/Y.2060 - Overview of the Internet of things

# Choosing among IoT standards is not straightforward



# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# Computer Network Recap

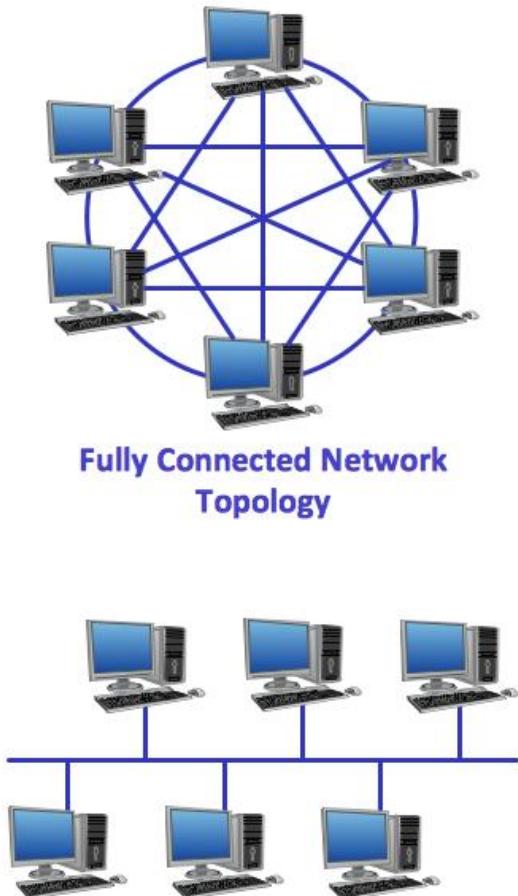
Internet-of-Things (IoT)

COCOS20

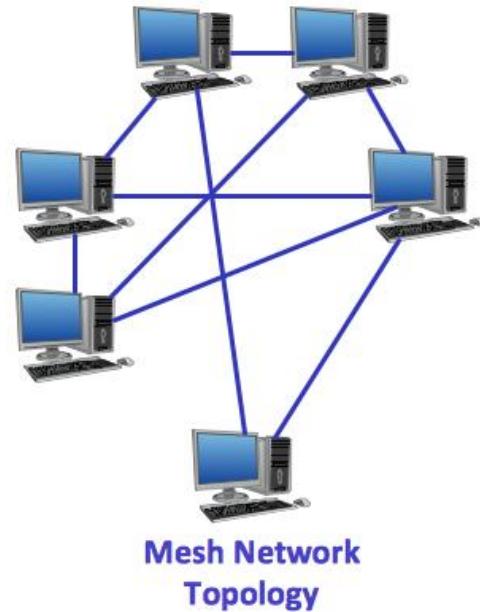
# Computer Network Terminology

- **Network:** group of computers and associated devices that are connected by communication facilities
- **Wide Area Network (WAN):** world-wide (Internet)
- **Metropolitan Area Network (MAN):** city-scale.
- **Local Area Network (LAN):** laboratory/office-scale (Ethernet).
  - **WLAN:** wireless LAN (Wi-Fi).
  - **WPAN:** wireless personal area network (Bluetooth).
  - **WBAN:** wireless body area network.

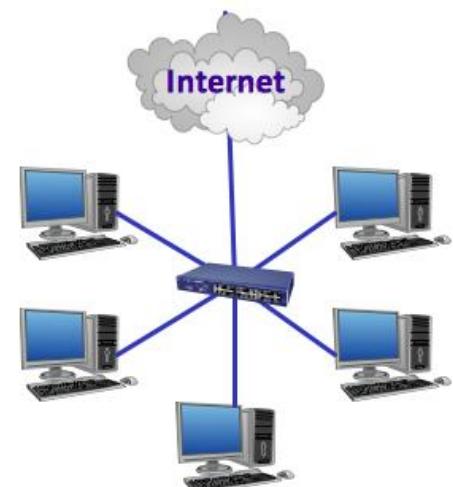
# Network Topologies



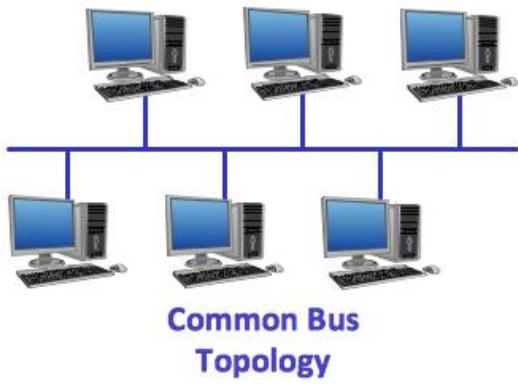
Fully Connected Network  
Topology



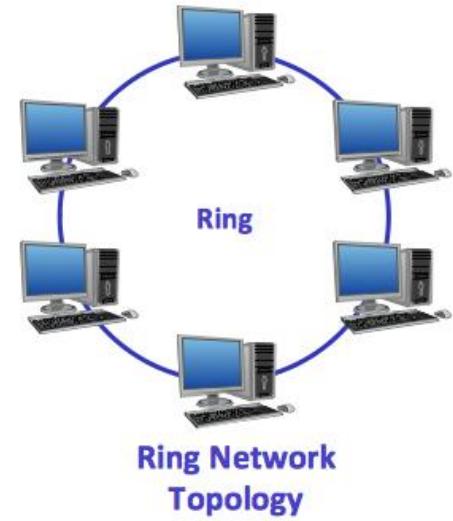
Mesh Network  
Topology



Star Network  
Topology



Common Bus  
Topology



Ring  
Topology

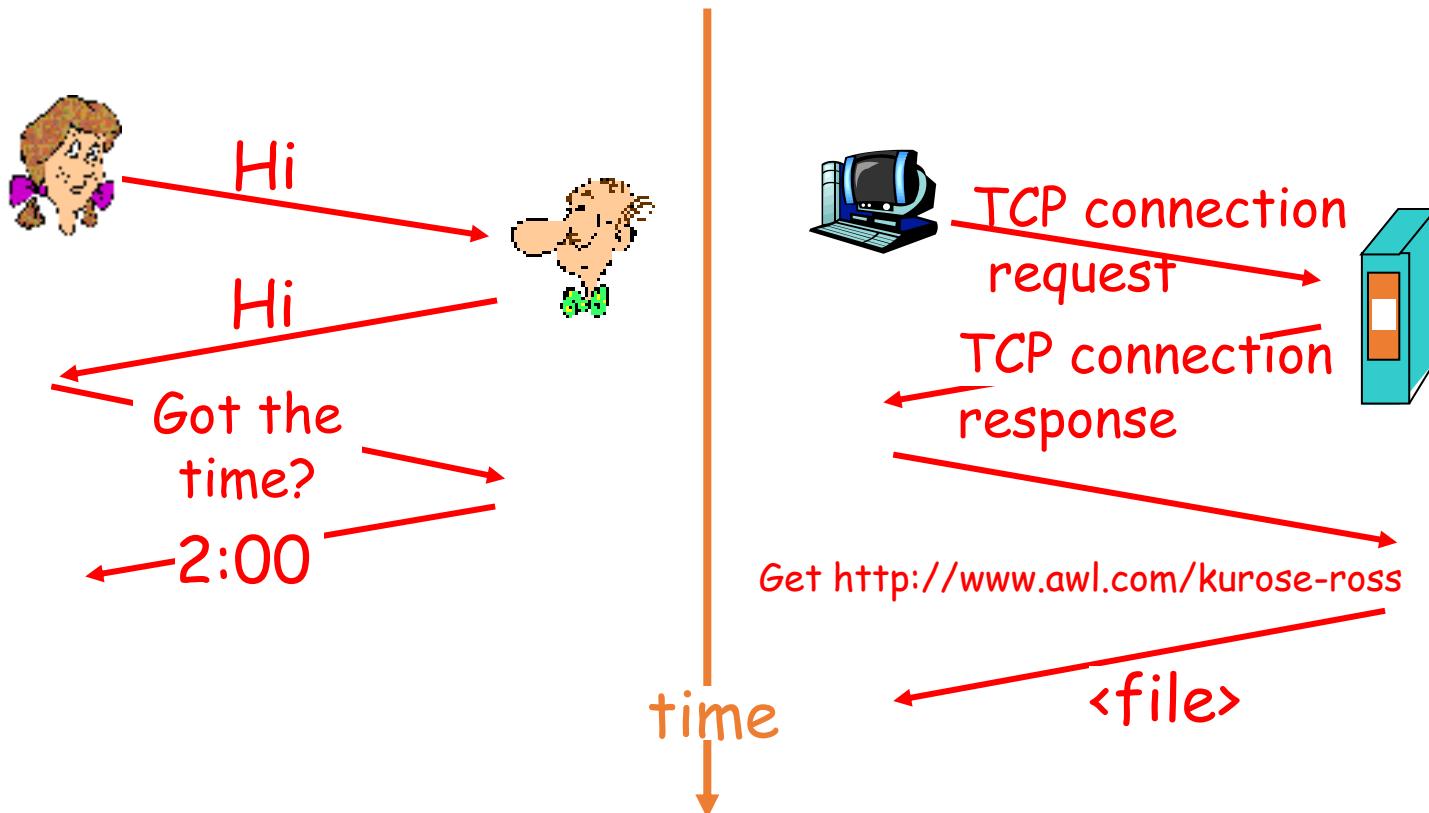
# Network Protocols

- Protocols are the **building blocks** of a network architecture.
- Formal standards and policies enabling communication.
- IEEE (Institute of Electrical and Electronics Engineers): standardization
  - Example: Project 802
    - 802.3: Ethernet
    - 802.11: WLAN
    - 802.15: WPAN

# Communication

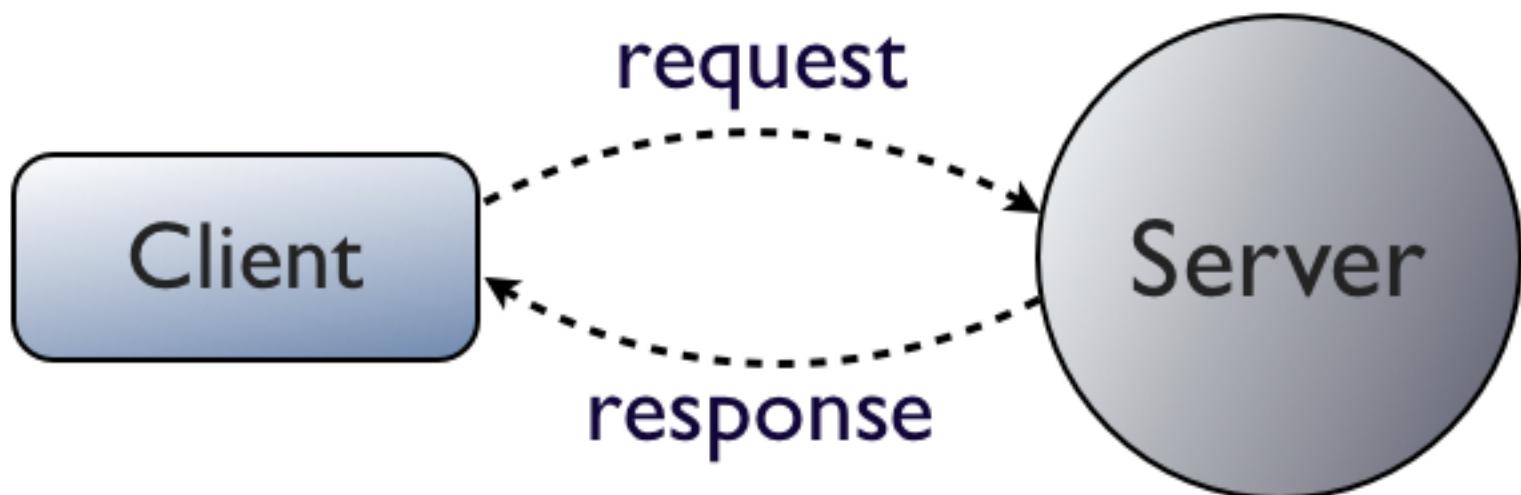
- Who initiates communication?
- Order of communication?
- How long can I talk?
- How loud can I speak?
- Do I have to say something specific at beginning or end?
- Do I have to add meta information?
- What do I do if I get interrupted?
- What do I do if I was not understood?

# Protocols



# Client/Server Model

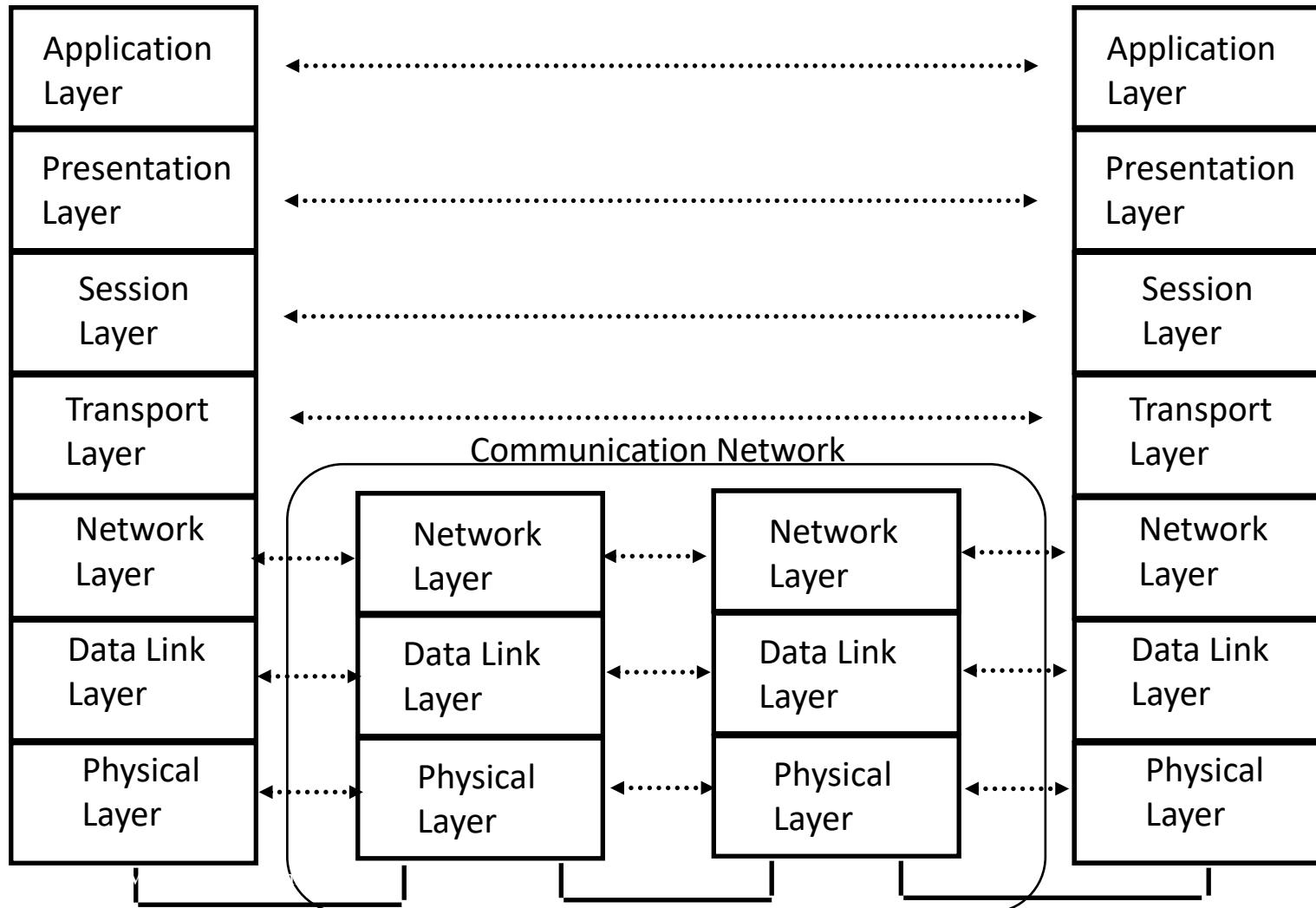
- Client: “active” (initiates communication)
- Server: “passive” (listens and responds)



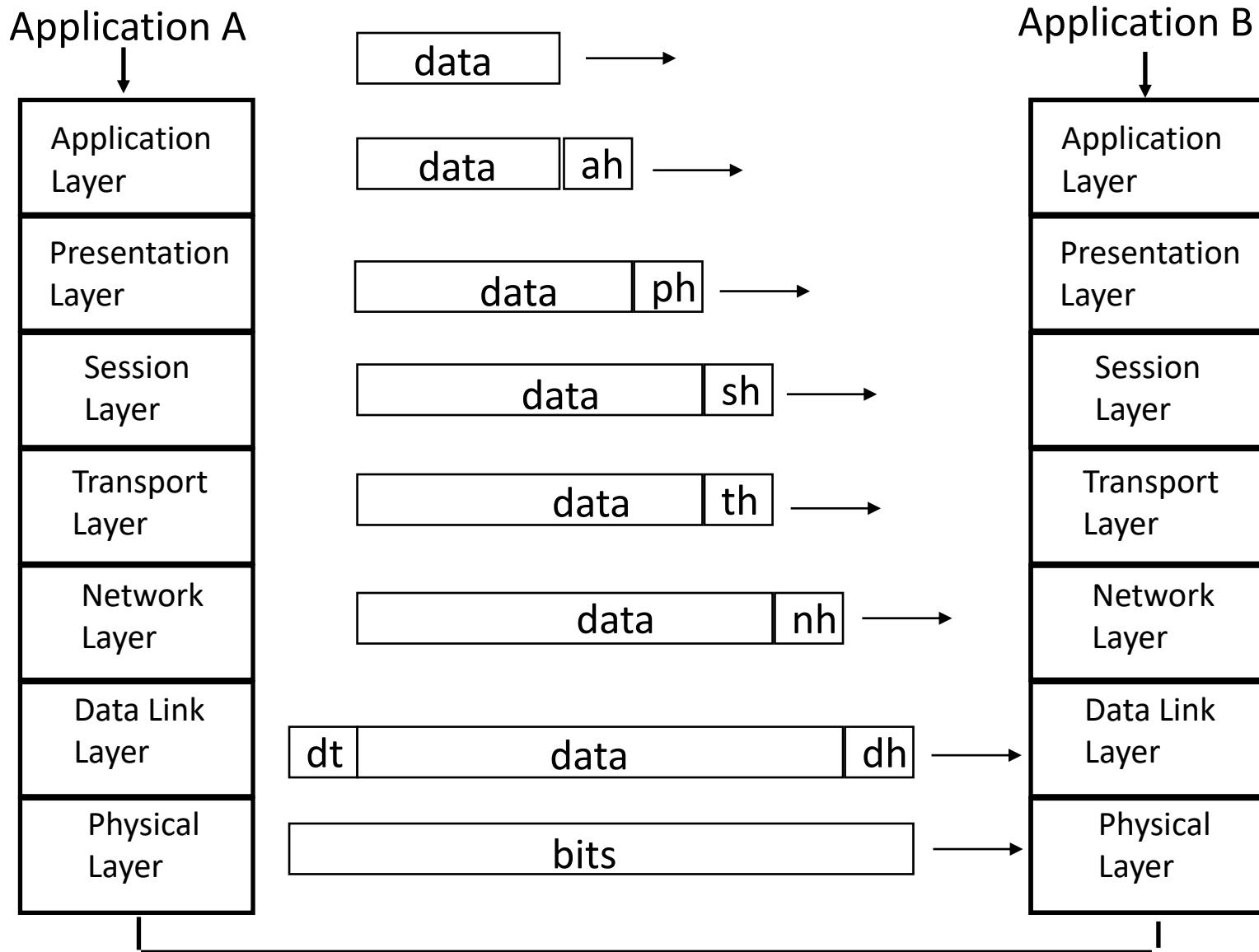
# Client/Server Model Examples

- HTTP (Hypertext Transfer Protocol)
- SMTP (Simple Mail Transfer Protocol)
- SSH (Secure Shell)
- DNS (Domain Name System)
- NFS/AFS (Network/Andrew File System)

# Network Protocols (“Protocol Stack”)



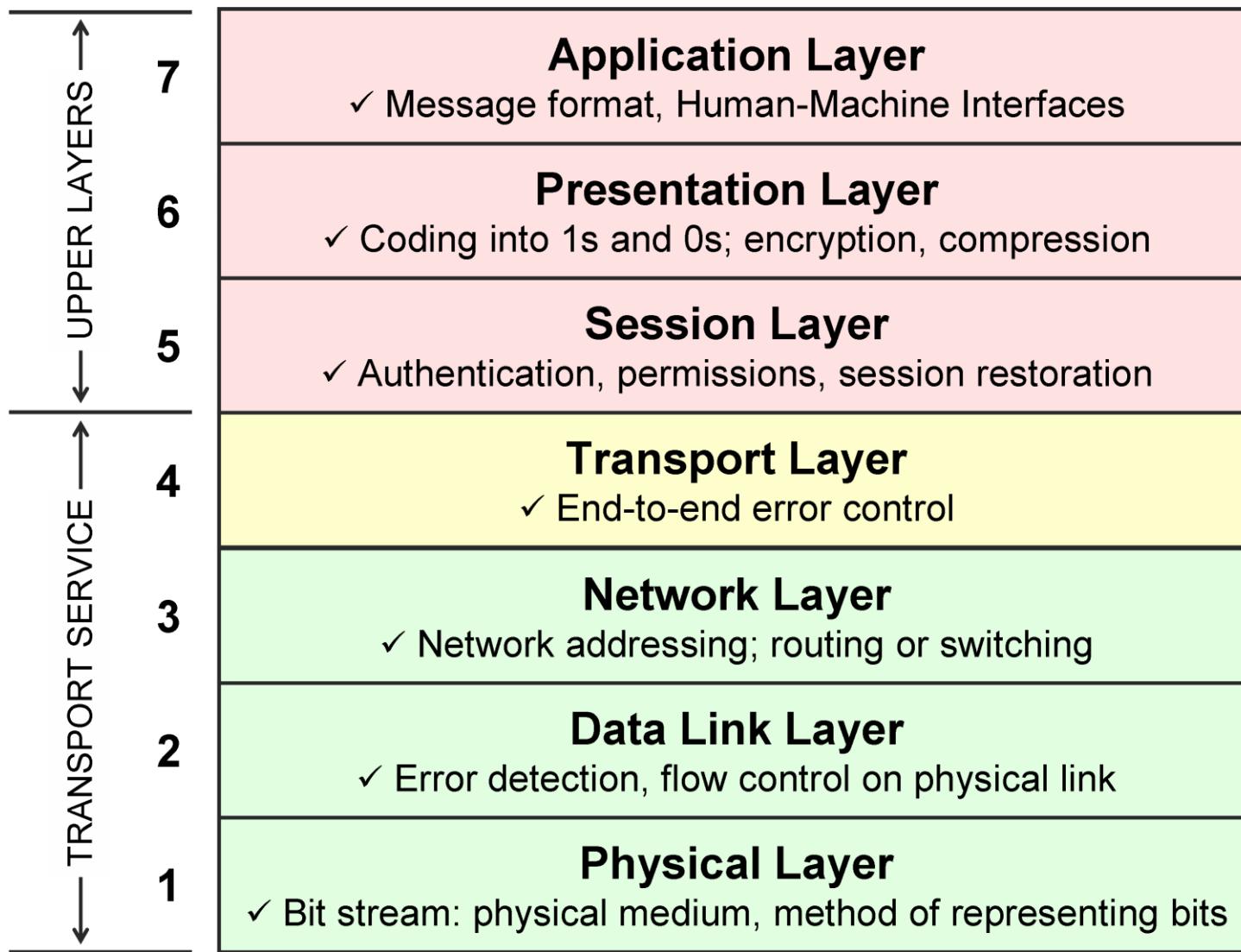
# Network Protocols (Headers/Trailers)



# Why a Layered Design?

- An explicit structure for dealing with a complex system
- Simplifies the design process
- Modularity of layers eases maintenance and updating of system components
- Accommodates incremental changes

# Open System Interconnection (OSI)



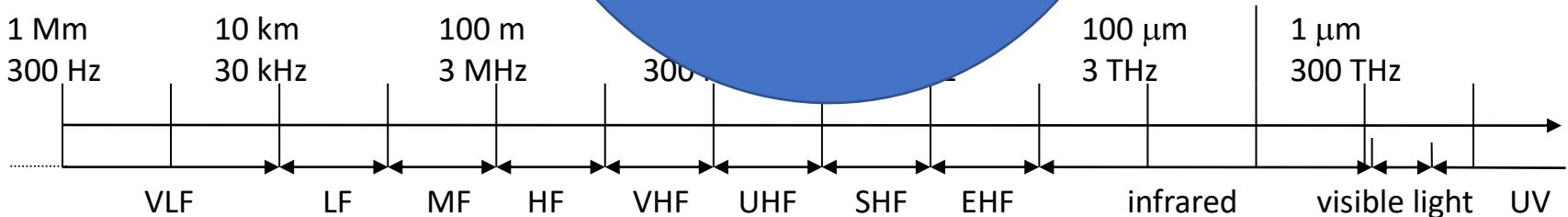
# Physical Layer (Layer 1)

- **Physical/electrical characteristics**
- Cable type, length, connectors, voltage levels, signal durations, ...
- Binary data (bits) as electrical or optical signals
- Frequencies (wireless)

# Wireless Characteristics

- VLF = Very Low Frequency
- LF = Low Frequency
- MF = Medium Frequency
- HF = High Frequency
- VHF = Very High Frequency
- Frequency and wave length
  - $\lambda = c/f$
  - wave length  $\lambda$ , speed of light

What is  
Frequency?



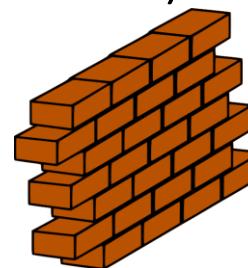
# Frequencies for Mobile Communication

- Low Frequencies:

- low data rates
- travel long distances
- follow Earth's surface
- penetrate objects and water (submarine communication)

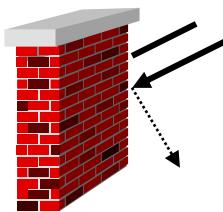
- High Frequencies:

- high data rates
- short distances
- straight lines
- cannot penetrate objects ("Line of Sight" or LOS)

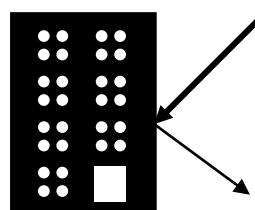


# Other Propagation Effects

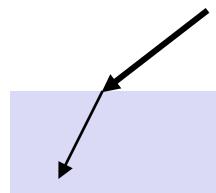
- **Shadowing**
- **Reflection** at large obstacles
- **Refraction** depending on the density of a medium
- **Scattering** at small obstacles
- **Diffraction** at edges



shadowing



reflection



refraction



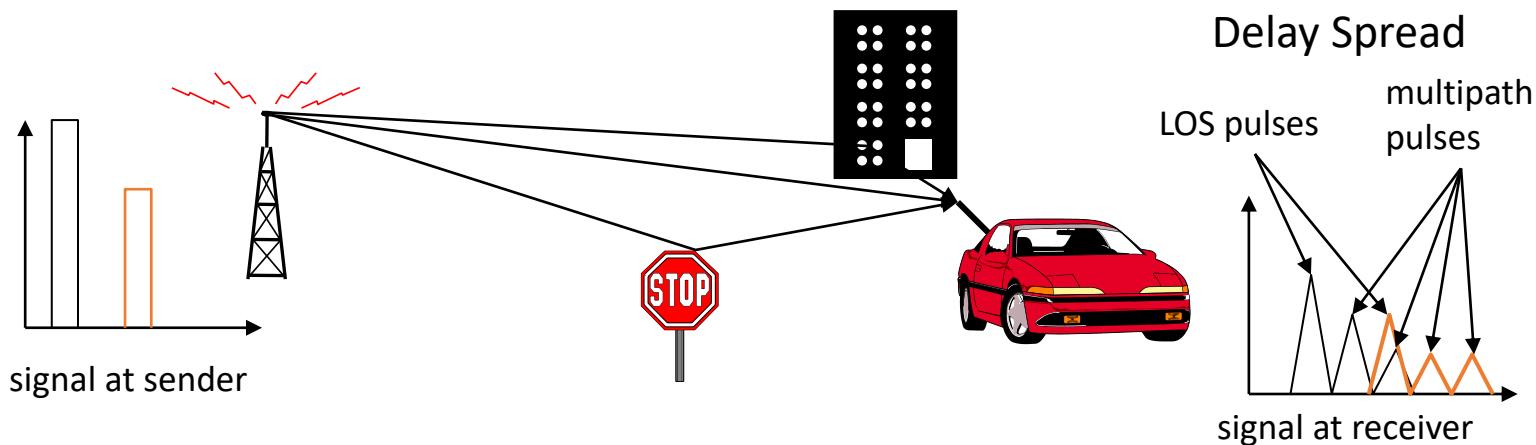
scattering



diffraction

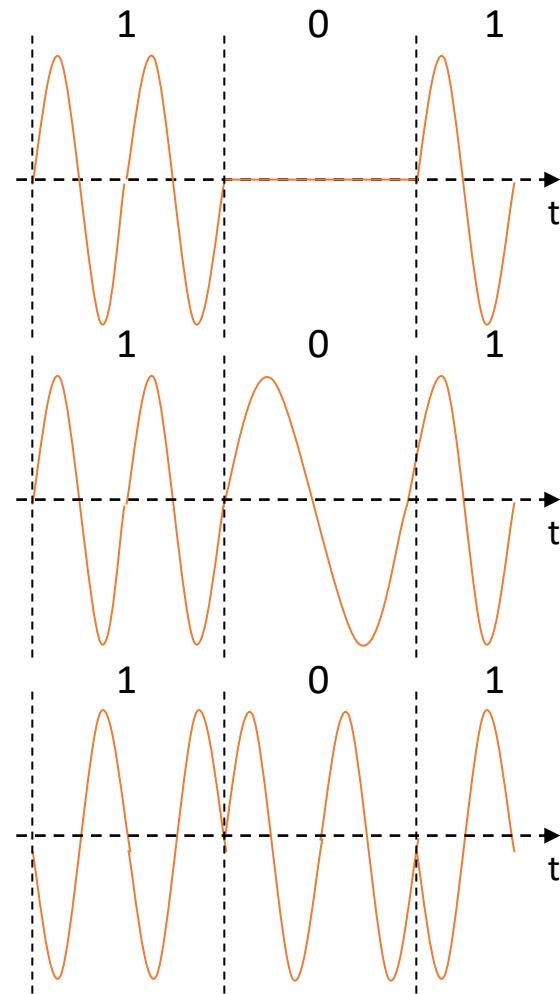
# Multipath Propagation

- Signal can take **many different paths** between sender and receiver due to reflection, scattering, diffraction



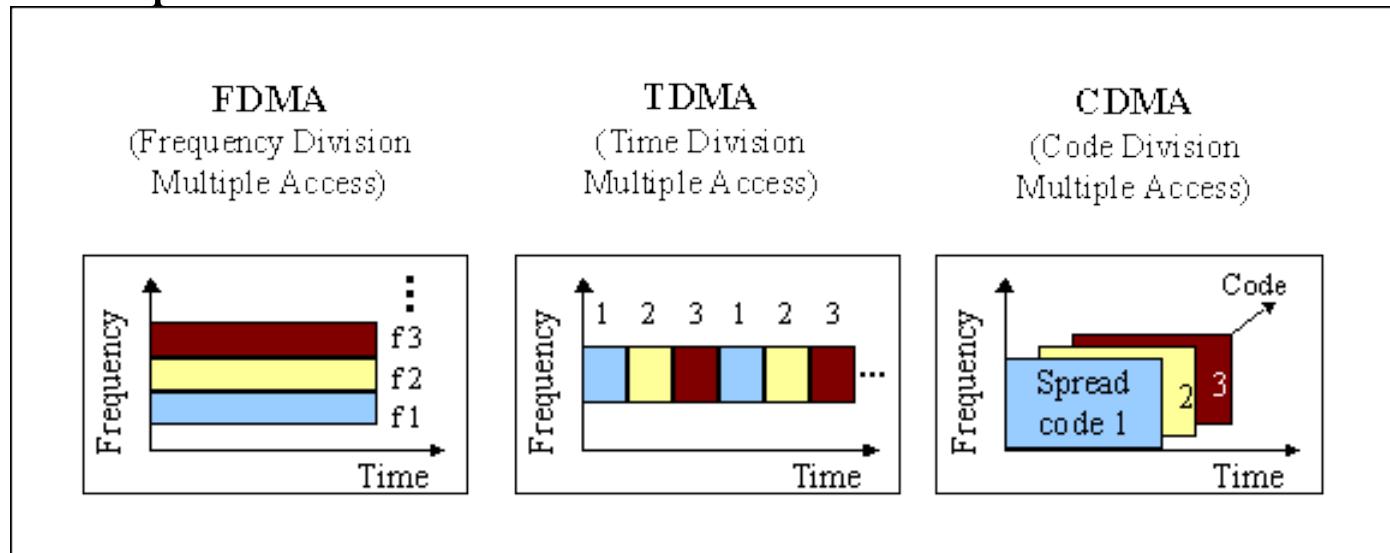
# Digital Modulation

- Amplitude Shift Keying (ASK)
- Frequency Shift Keying (FSK)
- Phase Shift Keying (PSK)



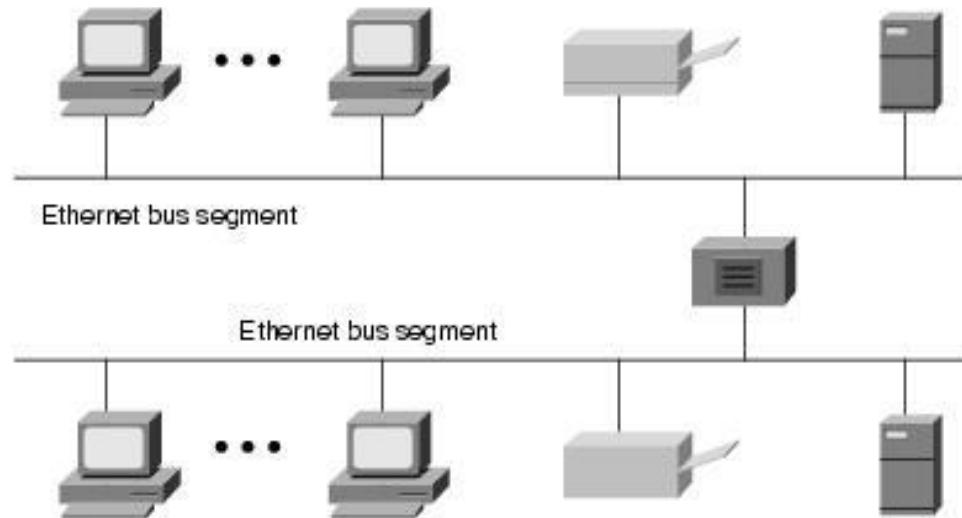
# Data Link Layer (Layer 2)

- **Defines when/how medium will be accessed for transmission**
- Units typically called “frames”; error detection/correction; divided into sublayers, including: **MAC = Medium Access Control** (MAC address 6f:00:2b:23:1f:32)
- Cell phone example:



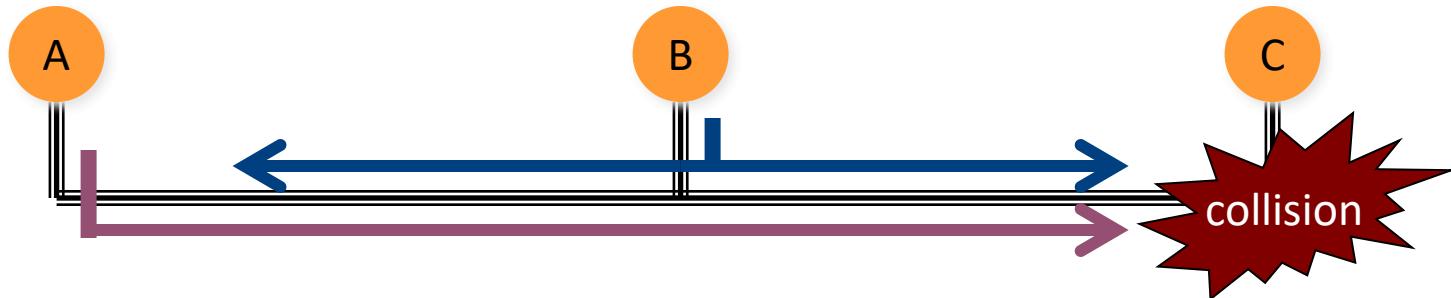
# Example: Ethernet (802.3)

- Most popular LAN technology, uses bus architecture
- Easy to install, inexpensive
- Data is broken into **packets**

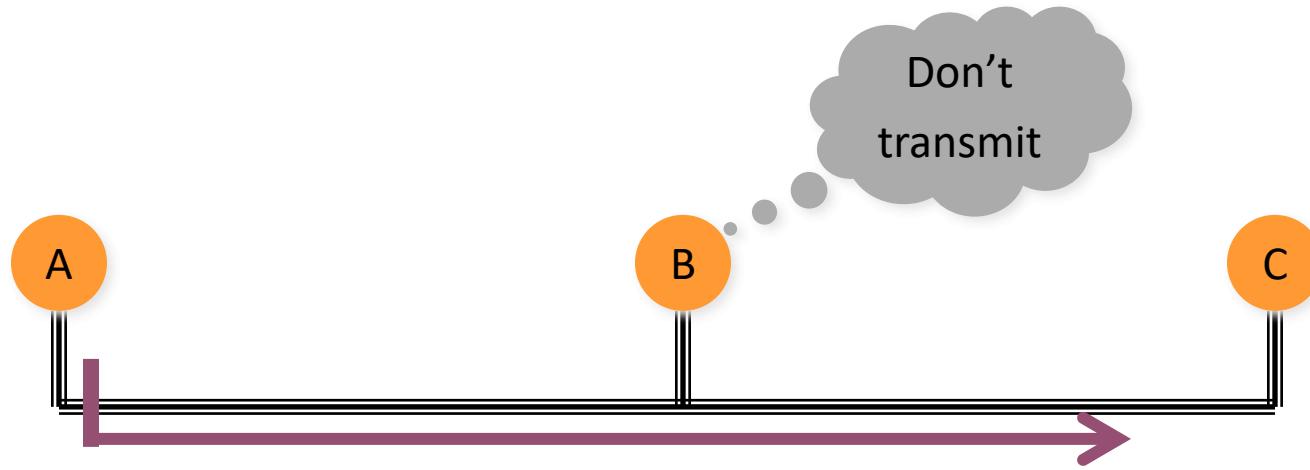


# Example: Ethernet

- Medium Access Control (MAC) protocol
- **CSMA/CD Protocol**
  - Carrier Sense
  - Multiple Access
  - Collision Detection



# Example: Ethernet

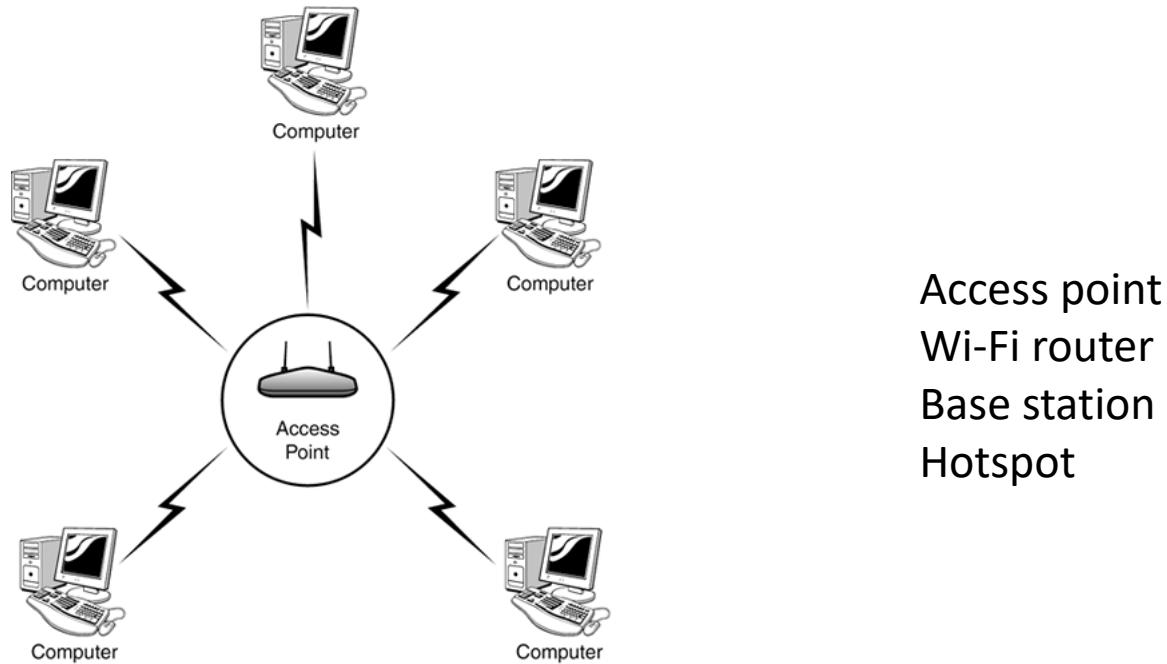


Can collisions still occur?

- “Sense” (listen) carrier (“is anyone else talking right now?”)
- If “busy”: wait; if “idle”: transmit
- CD: Keep listening while transmitting
  - If collision detected: retry at a later time

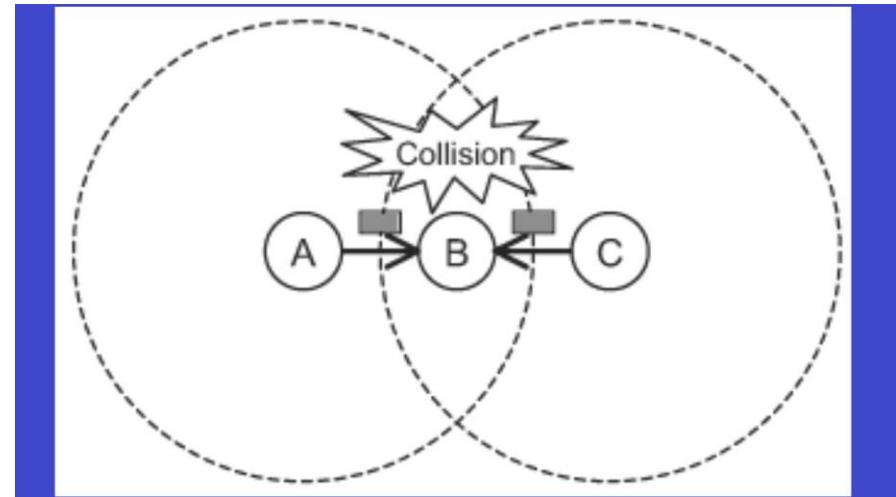
# Example: Wi-Fi (802.11)

- Most popular wireless LAN architecture



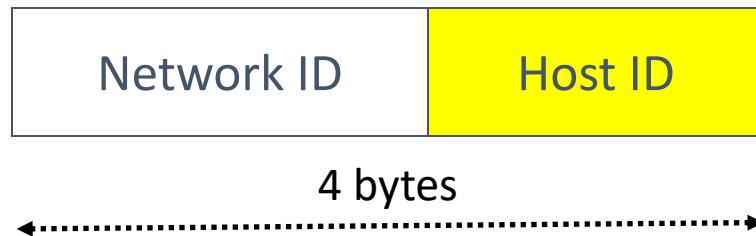
# Example: Wi-Fi (802.11)

- **CSMA/CA Protocol**
  - Carrier Sense
  - Multiple Access
  - Collision Avoidance
    - Channel reservations:
      - Transmitter sends request-to-send (RTS)
      - Receiver sends clear-to-send (CTS)
    - Advantages:
      - Nodes hearing RTS and/or CTS keep quiet
      - If collision, only small RTS or CTS packets are lost.



# Network Layer (Layer 3)

- **Dominant protocol: IP = Internet Protocol**
- Addressing and routing (sender & receiver IP address)
- Uses 32-bit **hierarchical address space** with location information embedded in the structure



- IPv4 address is usually expressed in dotted-decimal notation, e.g.:

**128.100.11.56**

# IPv4

**Class A**  
Subnet Mask

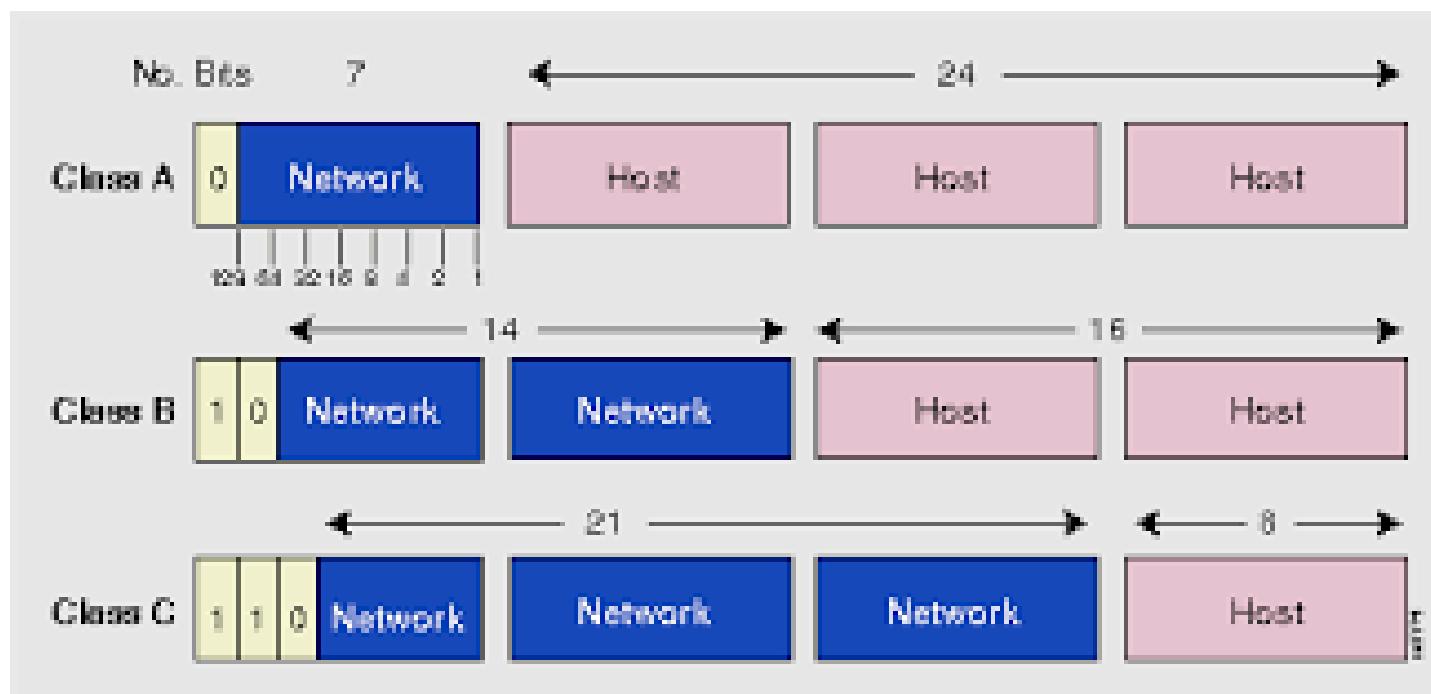
Netwok	Host	Host	Host
255	0	0	0

**Class B**  
Subnet Mask

Netwok	Network	Host	Host
255	255	0	0

**Class C**  
Subnet Mask

Netwok	Network	Network	Host
255	255	255	0

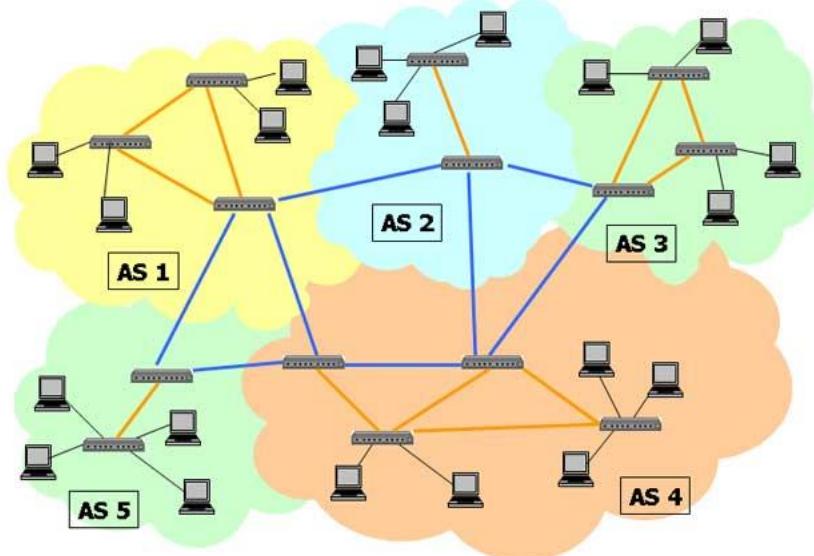


# IPv6

- IPv6 addresses are 128 bits long
- 16 bytes of IPv6 address are represented as a group of hexadecimal digits, separated by colons, e.g.:  
**2000:fdb8:0000:0000:0001:00ab:853c:39a1**
- Shorthand – leave out groups of zeros and leading zeros:  
**2000:fdb8::1:ab:853c:39a1**
- IPv4 Address space: 4,294,967,296 Addresses
- IPv6 Address space:  $(3.4 * 10^{38})$   
340,282,366,920,938,463,463,374,607,431,768,211,456

# Routers

- Form backbone of the Internet
- Use IP layer to identify source and destination of packets
- Look up **routing tables** that determines “**next hop**”



Destination	Next Hop
147.39.21.X	131.19.18.121
89.44.X.X	131.19.22.119
203.21.X.X	137.18.47.48

# Transport Layer (Layer 4)

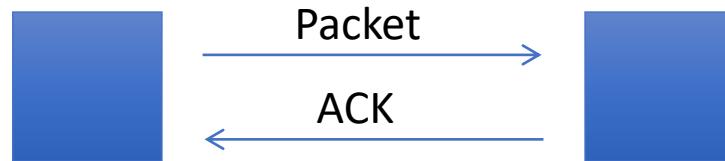
- **UDP (User Datagram Protocol)**



- Adds more addressing: “**ports**”
  - IP address tell you which computer
  - Ports tell you which application on that computer
  - Example: a web server “listens” to requests on port 80
  - Web browser: <http://www.google.com:80> = <http://216.58.216.100:80>
    - “:80”: optional
- **Unreliable!**
  - Packets can get lost; packets can arrive out of order

# Transport Layer

- **TCP** (Transmission Control Protocol)
- **Reliable** protocol!
- Adds ports (just like UDP), but also provides:
  - In-order delivery of packets (using sequence numbers)
  - Reliable delivery: using acknowledgment (ACK) packets



- **Flow control & congestion control:**
  - Allows receiver to slow down sender
  - Allows “network” to slow down sender

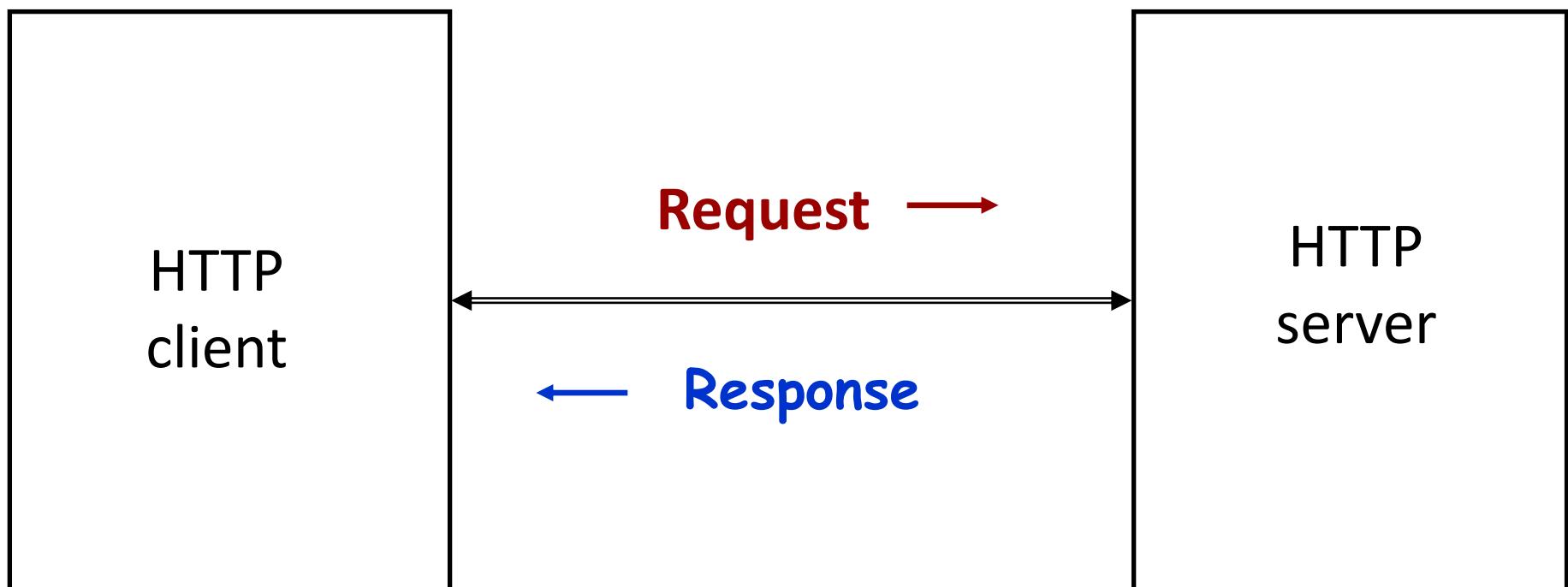
# UDP vs TCP

- TCP:
  - typical choice of most applications
  - do not want to lose data, out-of-order arrival, etc.
  - email, web traffic, financial transactions, etc.
- UDP:
  - can be “faster”
    - no flow/congestion control “slowing down” traffic
    - no retransmissions
    - good for “real-time” traffic
  - out-of-order arrival: can also “reorder” at application level
  - loss of data: can be acceptable
    - missing frames in video/audio stream

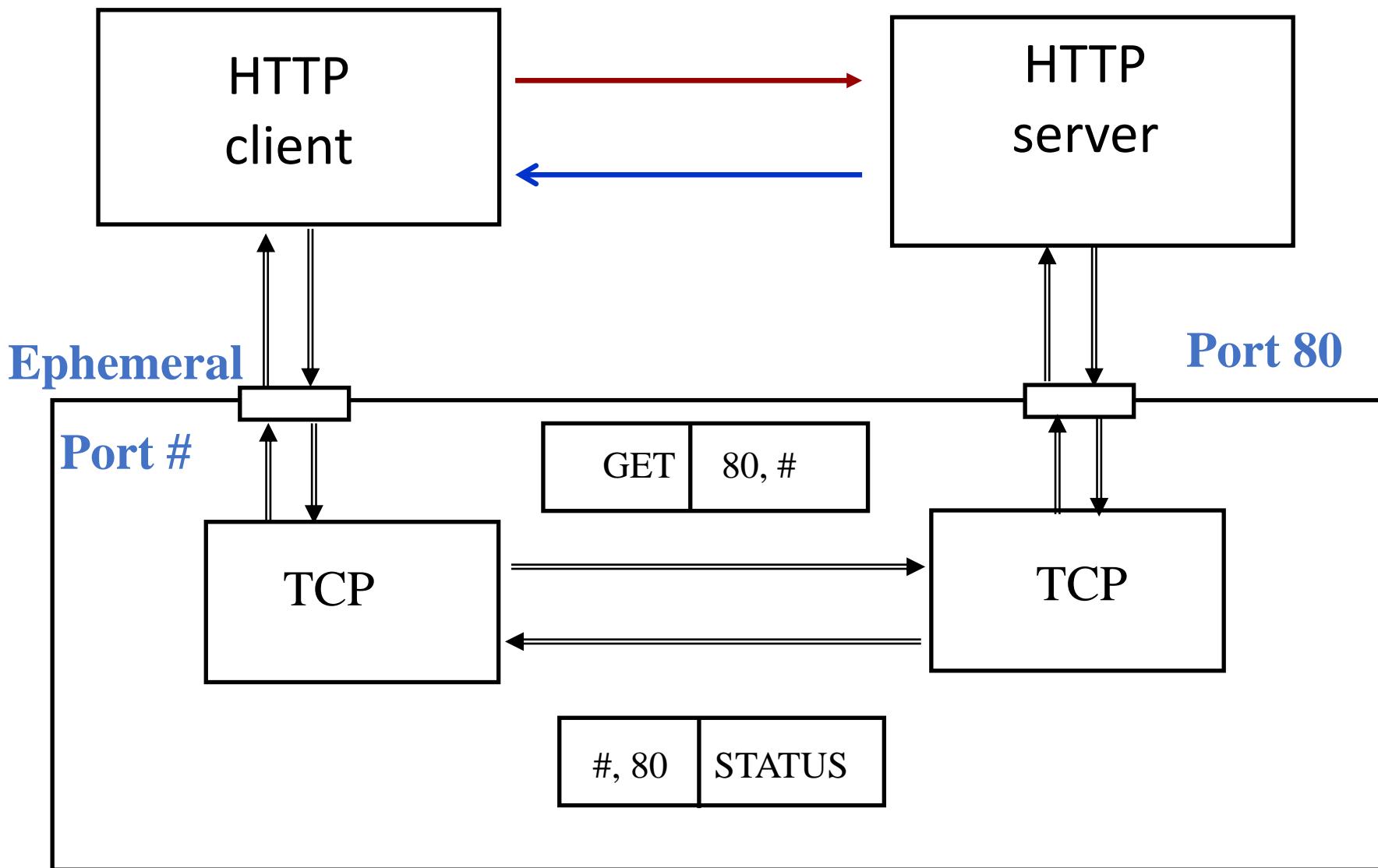
# Upper Layers (Layers 5-7)

- Session Layer
  - Management of “sessions”
- Presentation Layer
  - Data translation, formatting, encryption, compression
- Application Layer
  - Interface between user applications and lower network services

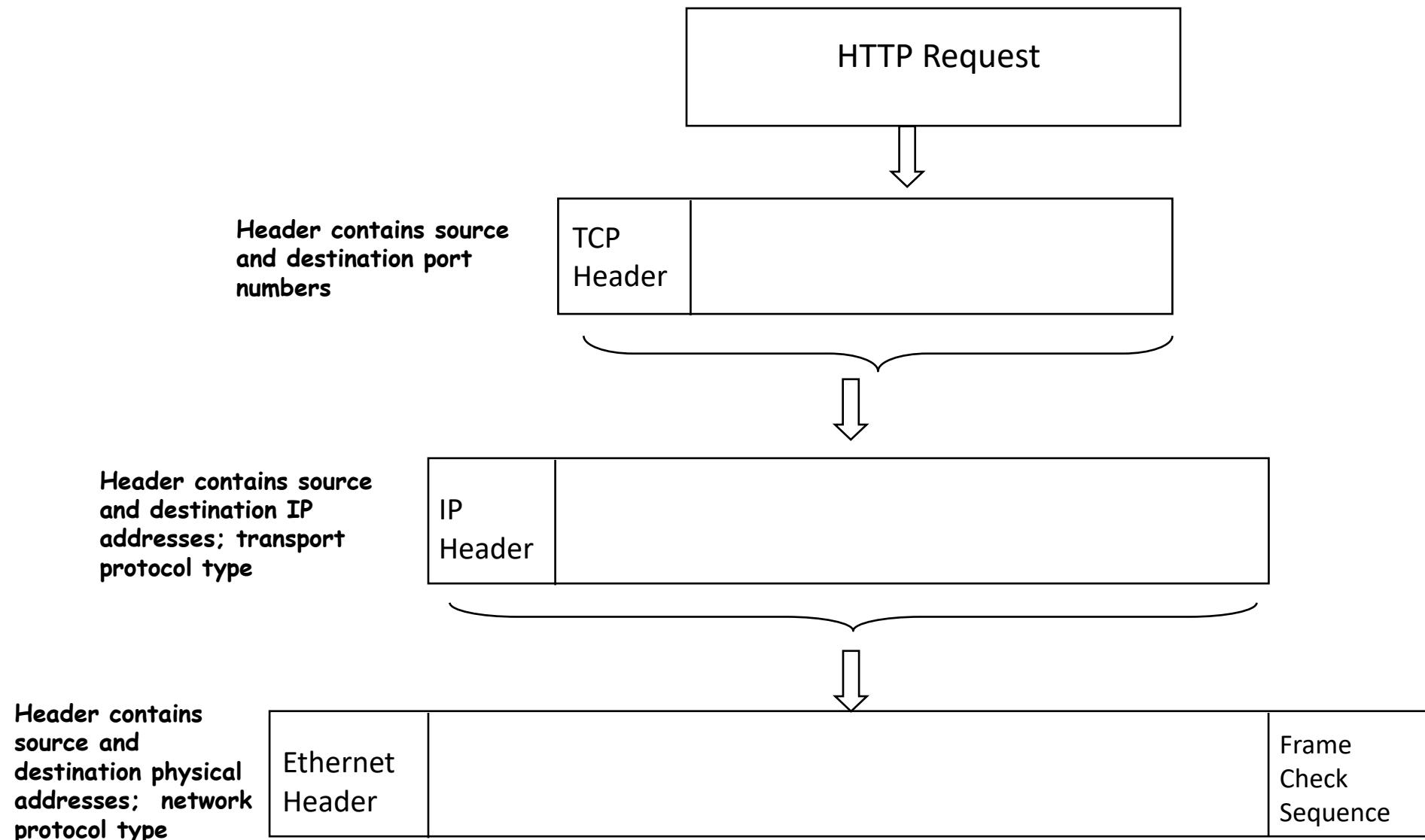
## Example: Web Servers



## Example: Web Servers



# Example: Web Servers



# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# IoT Architectural View

---

COCSC20

# Basic Premises

**Devices**

send and receive data interacting with the

**Network**

where the data is transmitted, normalized, and filtered using

**Edge Computing**

before landing in

**Data storage / Databases**

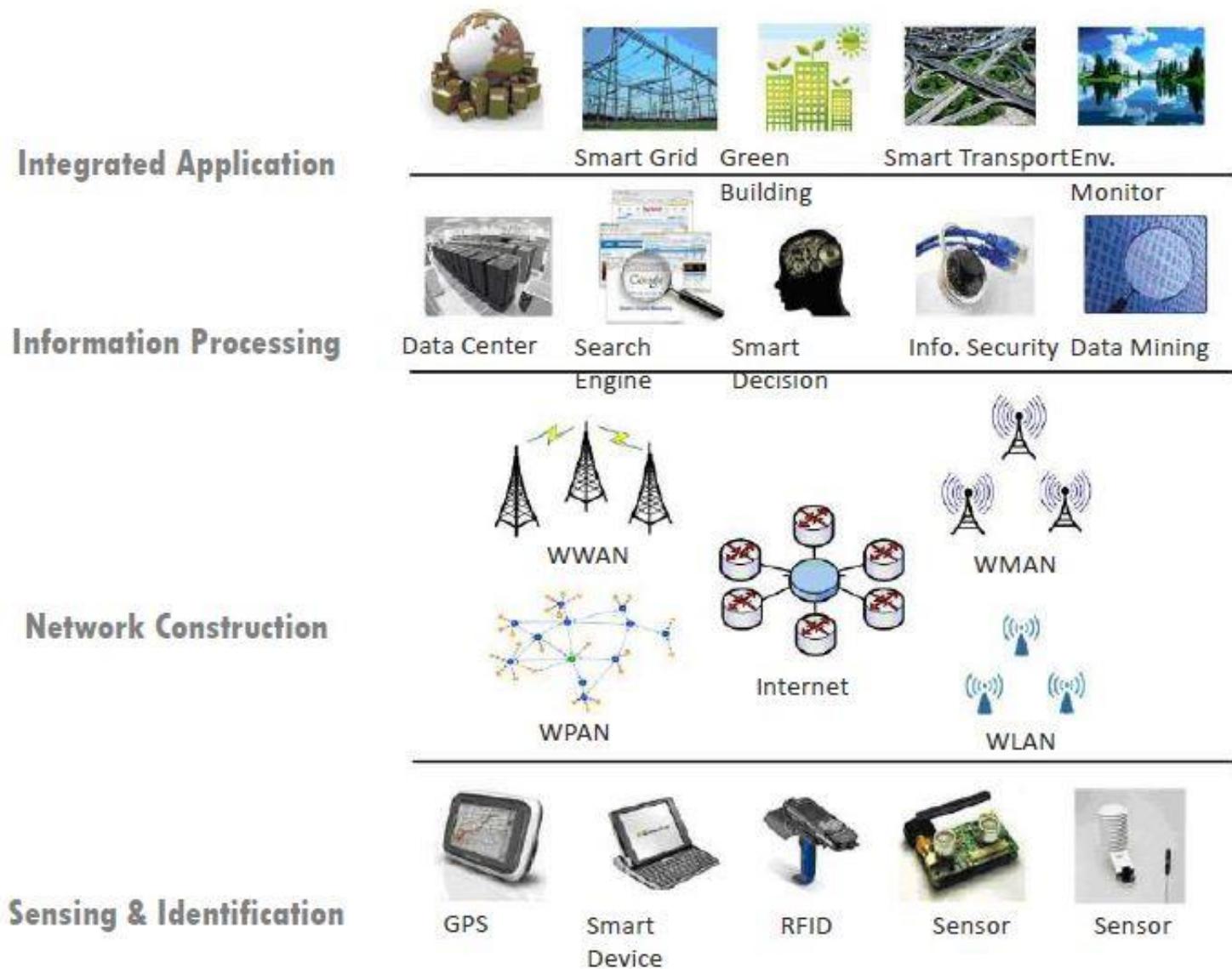
accessible by

**Applications**

which process it and provide it to people who will

**Act and Collaborate**

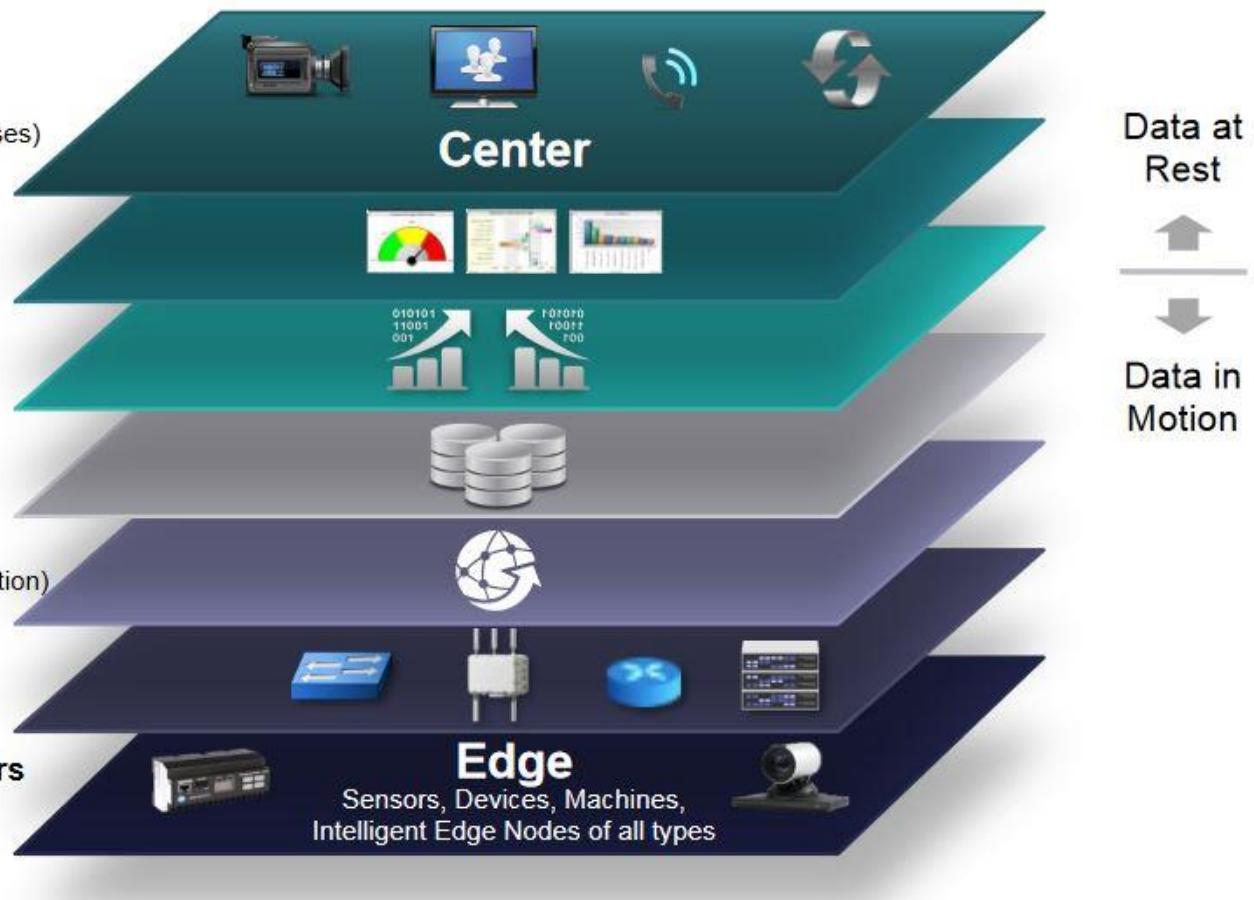
# IoT 4 Layers model



# Reference Model

## Levels

- 7 **Collaboration & Processes**  
(Involving People & Business Processes)
- 6 **Application**  
(Reporting, Analytics, Control)
- 5 **Data Abstraction**  
(Aggregation & Access)
- 4 **Data Accumulation**  
(Storage)
- 3 **Edge (Fog) Computing**  
(Data Element Analysis & Transformation)
- 2 **Connectivity**  
(Communication & Processing Units)
- 1 **Physical Devices & Controllers**  
(The "Things" in IoT)



# 1

## Physical Devices & Device Controllers (The “Things” in IoT)

IoT “devices” are capable of:

- Analog to digital conversion, as required
- Generating data
- Being queried / controlled over-the-net



### Edge

Sensors, Devices, Machines,  
Intelligent Edge Nodes of all types

## 2

## Connectivity

(Communication & Processing Units)

Level 2 functionality focuses  
on East-West communications

Connectivity includes:

- Communicating with and between the Level 1 devices
- Reliable delivery across the network(s)
- Implementation of various protocols
- Switching and routing
- Translation between protocols
- Security at the network level
- (Self Learning) Networking Analytics



3

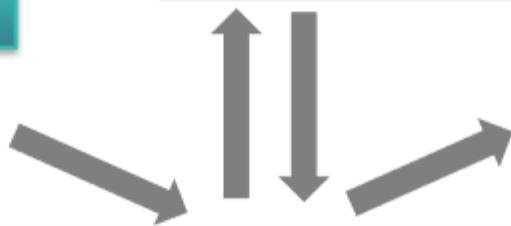
## Edge (Fog) Computing (Data Element Analysis & Transformation)

Level 3 functionality focuses on North-South communications

Include;

- Data filtering, cleanup, aggregation
- Packet content inspection
- Combination of network and data level analytics
- Thresholding
- Event generation

Data packets



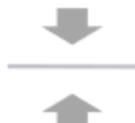
Information understandable to the higher levels

# 4

## Data Accumulation (Storage)

- Event filtering/sampling
- Event comparison
- Event joining for CEP
- Event based rule evaluation
- Event aggregation
- Northbound/southbound alerting
- Event persistence in storage

Query Based Data Consumption



Event Based Data Generation

Making network data usable by applications

1. Converts data-in-motion to data-at-rest
2. Converts format from network packets to database relational tables
3. Achieves transition from 'Event based' to 'Query based' computing
4. Dramatically reduces data through filtering and selective storing



or



# 5

## Data Abstraction (Aggregation & Access)

Abstracting the data interface for applications



## Information Integration

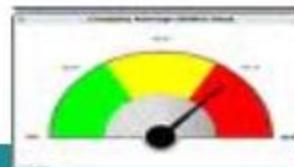
1. Creates schemas and views of data in the manner that applications want
2. Combines data from multiple sources, simplifying the application
3. Filtering, selecting, projecting, and reformatting the data to serve the client applications
4. Reconciles differences in data shape, format, semantics, access protocol, and security



# 6

## Application

(Reporting, Analytics, Control)



Control  
Applications



Vertical and  
Mobile  
Applications



Business  
Intelligence  
and Analytics

# 7

## Collaboration & Processes

(Involving people and business processes)



Center

# How Many Layers in OSI model?

- A. Four
- B. Five
- C. Six
- D. Seven
- E. None of the above.

# TCP/IP stands for?

---

Transmission Control Protocol/Internet Networking Protocol have

- A. Four
- B. Five
- C. Six
- D. Seven
- E. None of the above.

## OSI MODEL

Application Layer

Presentation Layer

Session Layer

Transport Layer

Network Layer

Data Link Layer

Physical Layer

## TCP/IP MODEL

Application Layer

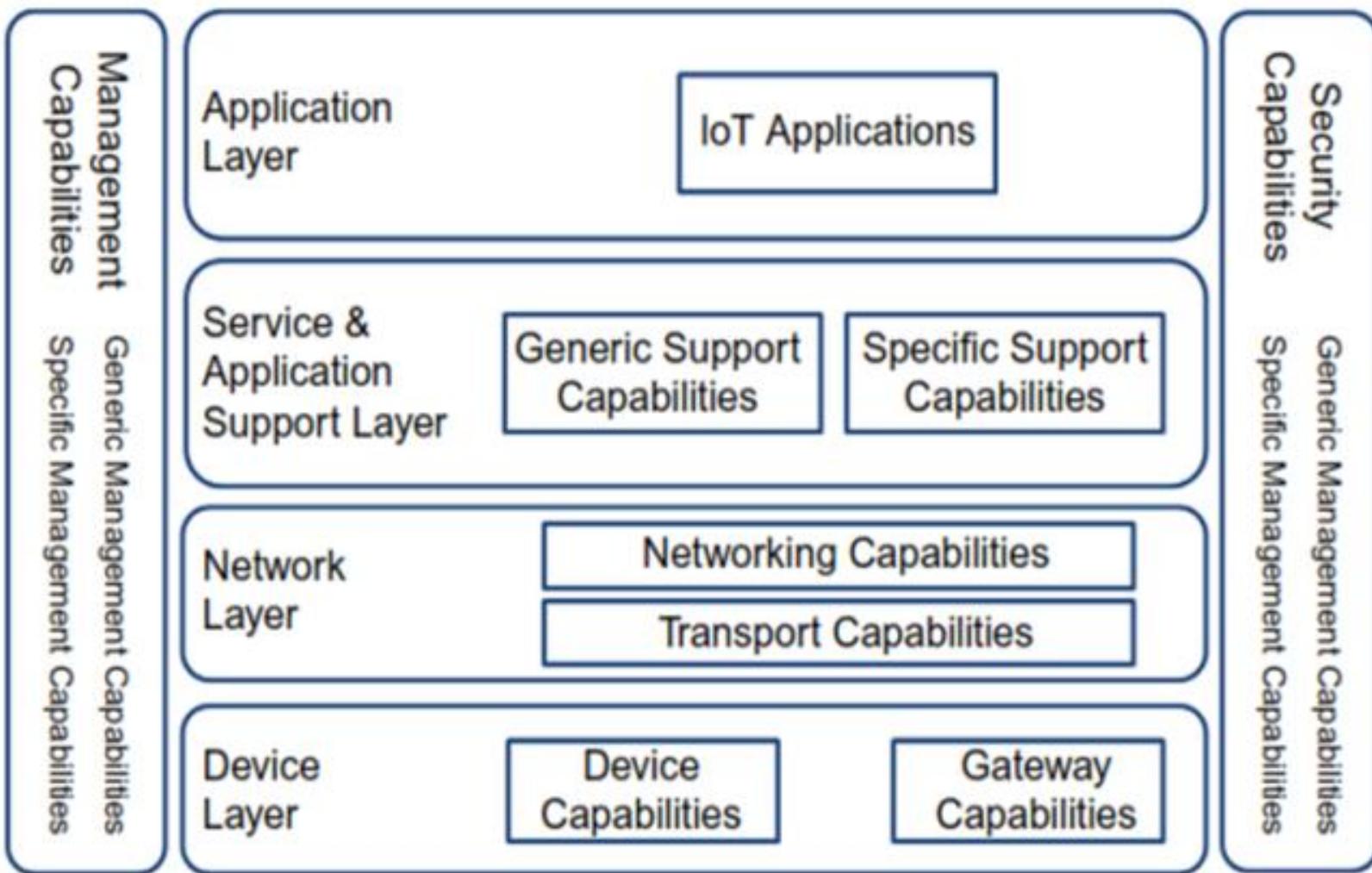
Transport Layer

Internet Layer

Network Access Layer

	<b>IoT Stack</b>	<b>Web Stack</b>
<b>TCP/IP Model</b>	IoT Applications Device Management	Web Applications
<b>Data Format</b>	Binary, JSON, CBOR	HTML, XML, JSON
<b>Application Layer</b>	CoAP, MQTT, XMPP, AMQP	HTTP, DHCP, DNS, TLS/SSL
<b>Transport Layer</b>	UDP, DTLS	TCP, UDP
<b>Internet Layer</b>	IPv6/IP Routing 6LoWPAN	IPv6, IPv4, IPSec
<b>Network/Link Layer</b>	IEEE 802.15.4 MAC IEEE 802.15.4 PHY / Physical Radio	Ethernet (IEEE 802.3), DSL, ISDN, Wireless LAN (IEEE 802.11), Wi-Fi

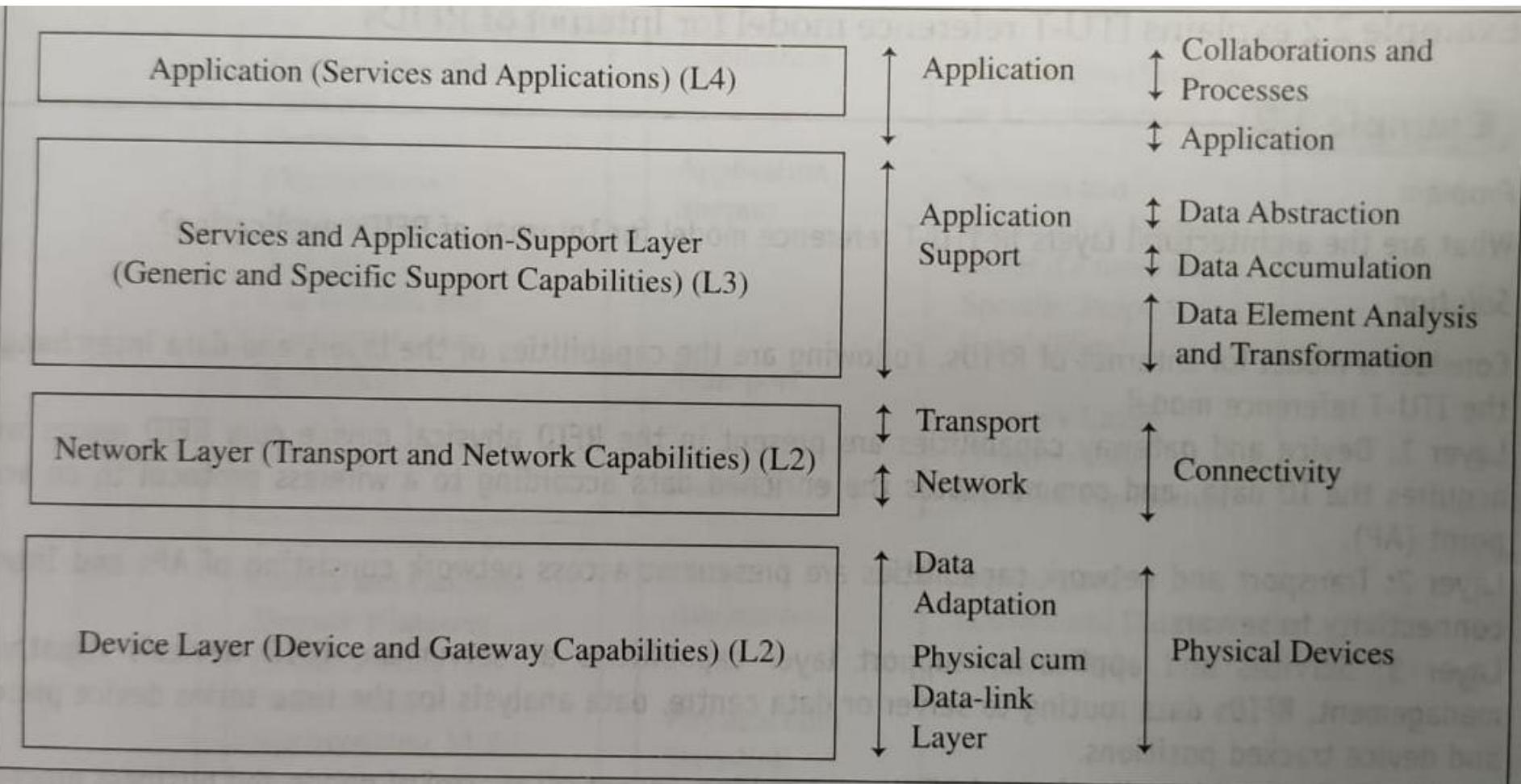
# ITU-T IoT Reference Model



ICMP stands for

- A. Internet Connect Message Protocol
- B. Internet Control Message Protocol
- C. International Connect Message Protocol
- D. International Control Message Protocol

# Comparison



# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# Mobile Adhoc Networks

COCSC20

# Wi-Fi

- Wi-Fi:
  - name is NOT an abbreviation
- **Wireless Local Area Network (WLAN)** technology.
- WLAN and Wi-Fi often used synonymous.
- Typically **in 2.4 and 5 GHz bands**.
- Based on **IEEE 802.11** family of standards.

# IEEE

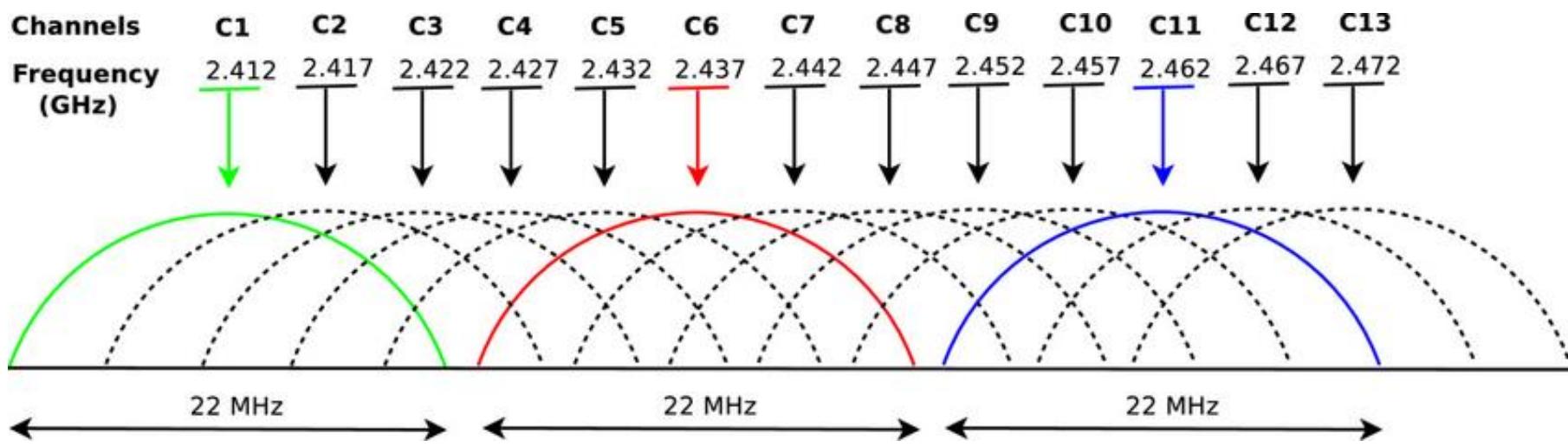
- IEEE (Institute of Electrical and Electronics Engineers) established the 802.11 Group in 1990. Specifications for standard ratified in 1997.
- Initial speeds were 1 and 2 Mbps.
- IEEE modified the standard in 1999 to include:
  - 802.11b
  - 802.11a
  - 802.11g
  - 802.11n
  - **802.11ac**

# WLAN (Wi-Fi)

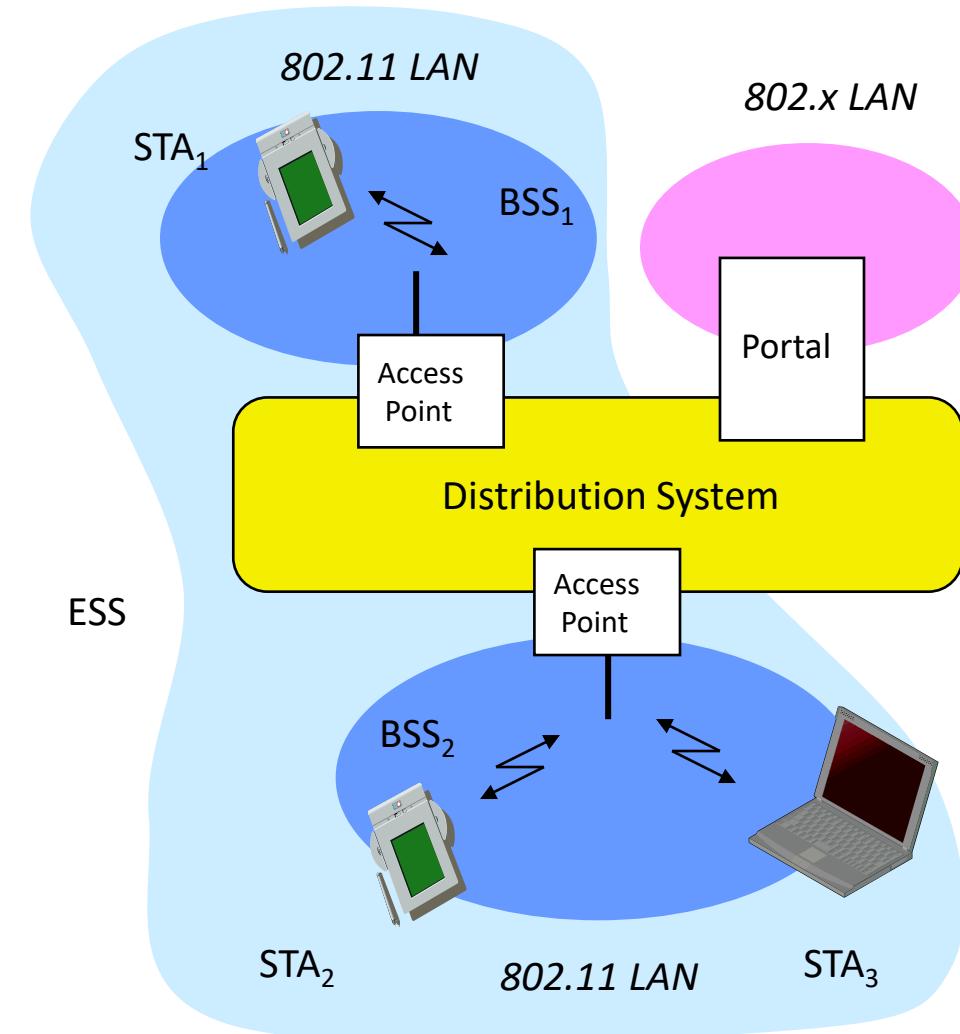
## 802.11 Wireless Standards

IEEE Standard	802.11a	802.11b	802.11g	802.11n	802.11ac
Year Adopted	1999	1999	2003	2009	2014
Frequency	5 GHz	2.4 GHz	2.4 GHz	2.4/5 GHz	5 GHz
Max. Data Rate	54 Mbps	11 Mbps	54 Mbps	600 Mbps	1 Gbps
Typical Range Indoors*	100 ft.	100 ft.	125 ft.	225 ft.	90 ft.
Typical Range Outdoors*	400 ft.	450 ft.	450 ft.	825 ft.	1,000 ft.

# Wi-Fi Channels

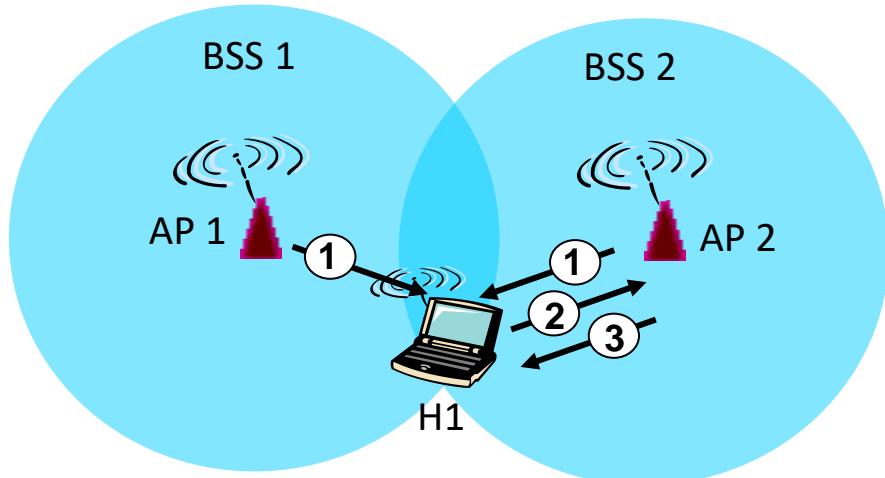


# 802.11 - Architecture of an Infrastructure Network



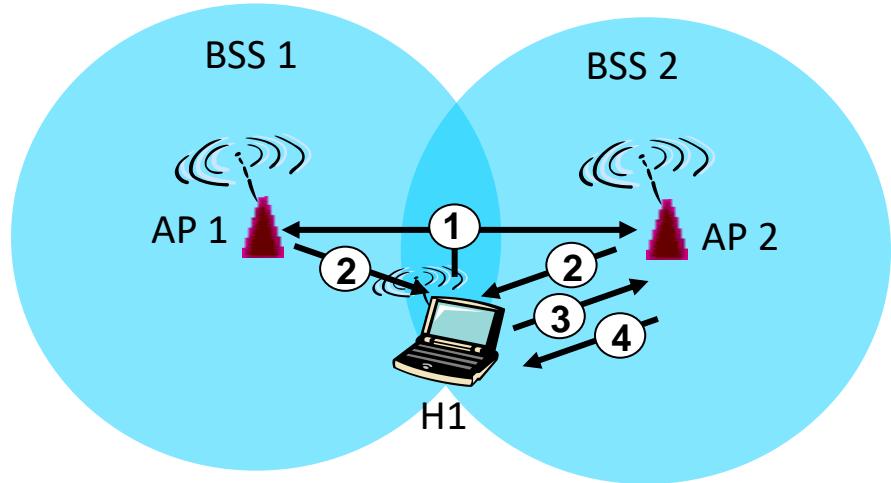
- **Station (STA)**
  - terminal with access mechanisms to the wireless medium and radio contact to the access point
- **Basic Service Set (BSS)**
  - group of stations using the same radio frequency
- **Access Point**
  - station integrated into the wireless LAN and the distribution system
- **Portal**
  - bridge to other (wired) networks
- **Distribution System**
  - interconnection network to form one logical network (ESS: Extended Service Set) based on several BSS

# Wi-Fi (802.11) Scanning



## Passive Scanning

- (1) Beacons sent from APs
- (2) Association Request sent from H1 to selected AP
- (3) Association Response sent from AP to H1



## Active Scanning

- (1) Probe Request (broadcast) sent from H1
- (2) Probe Response sent from APs
- (3) Association Request sent from H1 to selected AP
- (4) Association Response sent from AP to H1

# Wi-Fi Alliance Mission Statement

- Non-profit organization
- Certify the interoperability of products and services based on IEEE 802.11 technology
- Grow the global market for **Wi-Fi® CERTIFIED** products and services across all market segments, platforms, and applications
- Rigorous interoperability testing requirements

# Certificate & Logo

Wi-Fi® Interoperability Certificate

Certification ID: 24567832AP

**Wi-Fi CERTIFIED abgn**

This certificate represents the capabilities and features that have passed the interoperability testing governed by the Wi-Fi Alliance.

Detailed descriptions of these features can be found at [www.wi-fi.org/certificate](http://www.wi-fi.org/certificate)

**Certification Date:** February 14, 2004

**Category:** Access Point

**Company:** Name of Company

**Product:** Wireless LAN Access Point/Router Model#EX1010

**Model/SKU #:** EX1010

This product has passed Wi-Fi certification testing for the following standards:

IEEE Standard	Security	Quality of Service	Public Access
802.11a 802.11b 802.11g 802.11n	WPA - Personal WPA - Enterprise WPA2 - Personal (802.11i) WPA2 - Enterprise (802.11i)	WME (802.11e EDCA profile) WSM (802.11e HCCA profile)	
<b>Regulatory</b> 802.11d 802.11h	<b>Suplicant</b> EAP-TLS EAP-TTLS/MSCHAPv2 EAP-TTLS/PAP PEAPv0/EAP-MSCHAPv2 PEAPv1/EAP-GTC PEAPv1/EAP-MD5 EAP-SIM		
	<b>Authentication Server</b> EAP-TLS EAP-TTLS/MSCHAPv2 EAP-TTLS/PAP PEAPv0/EAP-MSCHAPv2 PEAPv1/EAP-GTC PEAPv1/EAP-MD5		

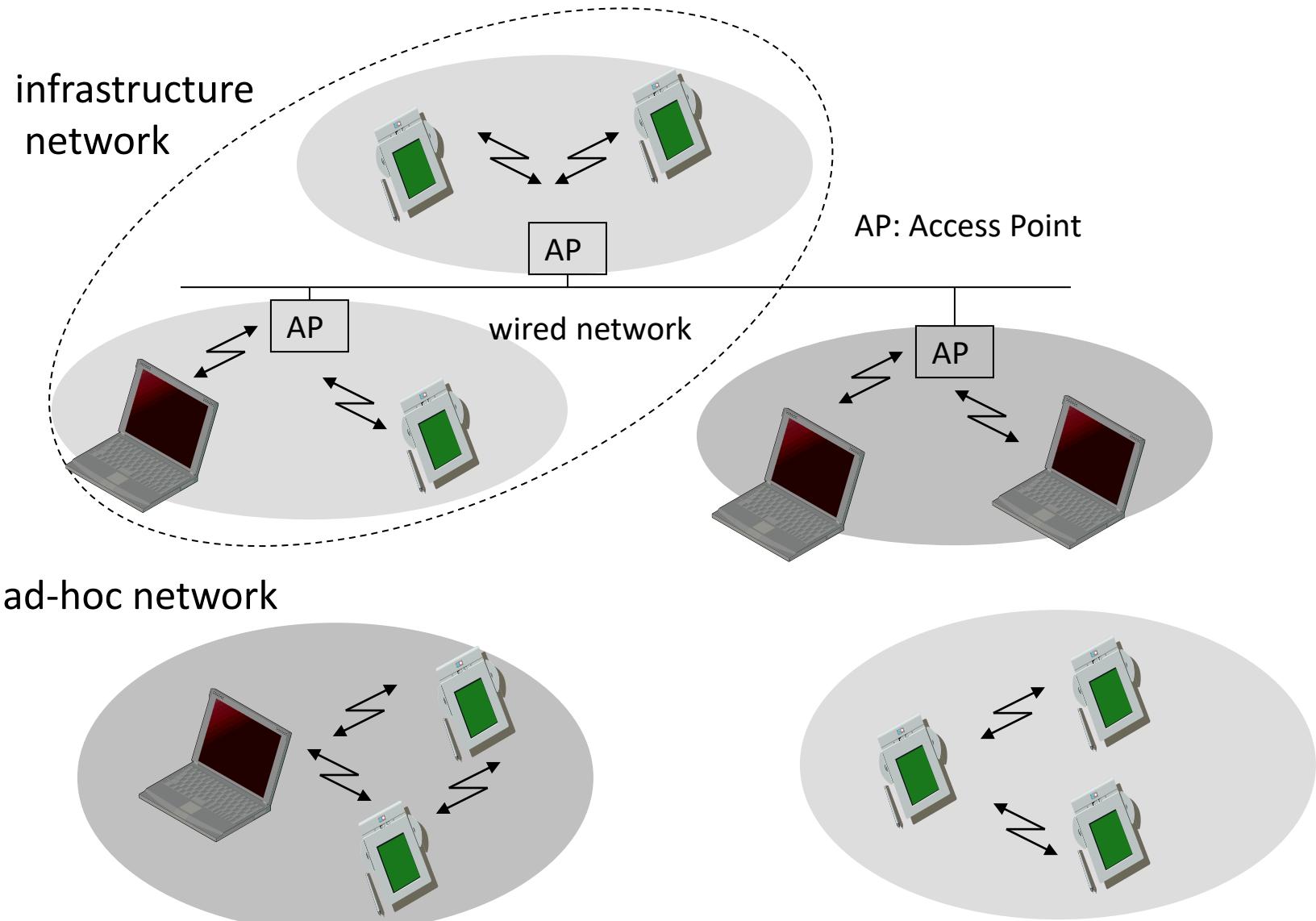
For more information: [www.wi-fi.org/certified\\_products](http://www.wi-fi.org/certified_products)

Certificate inside packaging (optional)



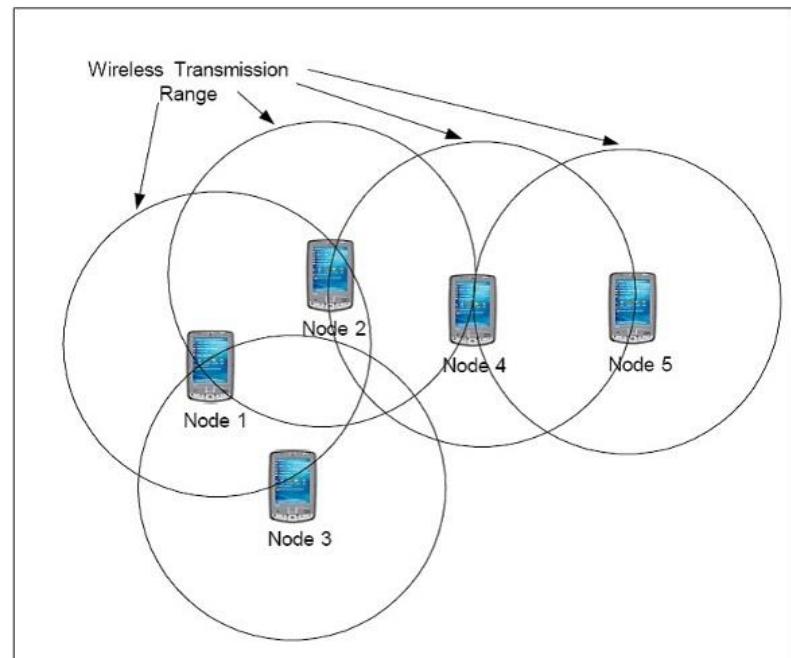
- Logo on product packaging (mandatory)
- Helps retailers and consumers

# Infrastructure vs. Ad-Hoc Networks



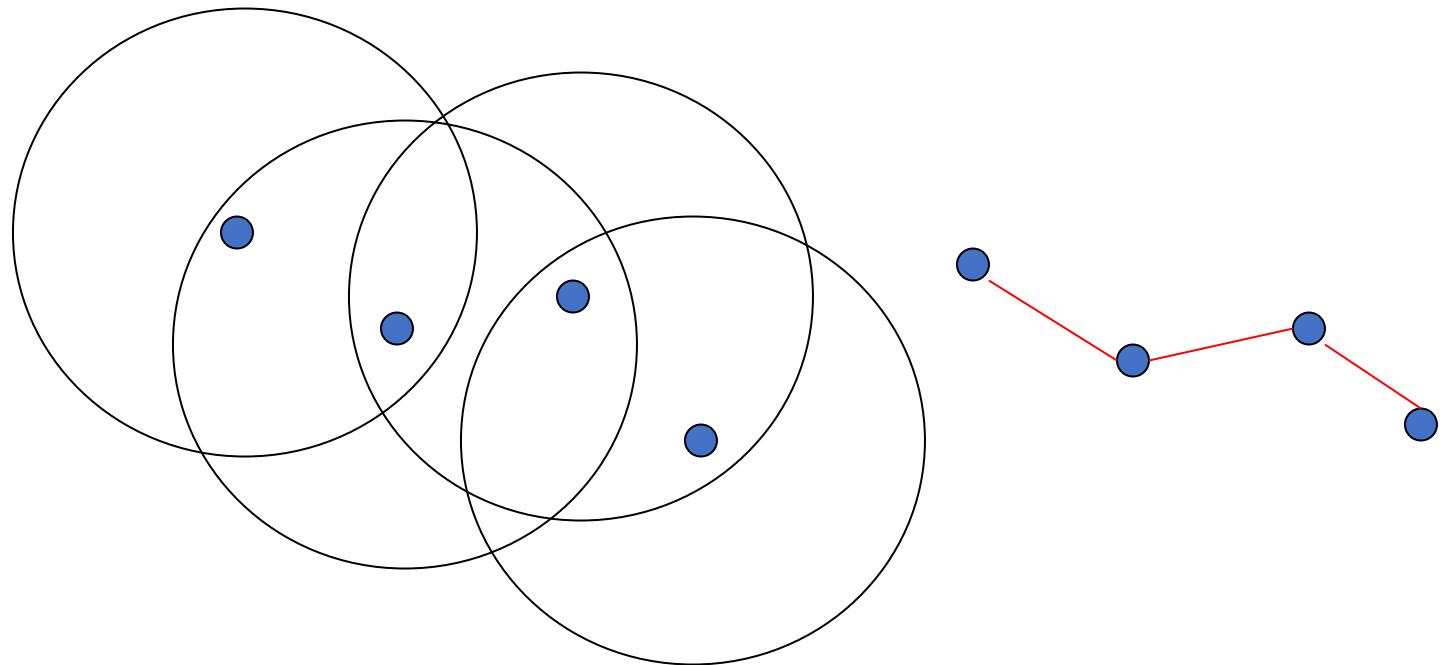
# Infrastructure-Less (Ad-Hoc)

- Ad-hoc means '*for this purpose*'
- No need for infrastructure (like routers, cell towers, etc.)
- MANET: **Mobile Ad-Hoc Network**



# Routing

- Packets may need to traverse multiple links to reach destination
- Mobility causes route changes

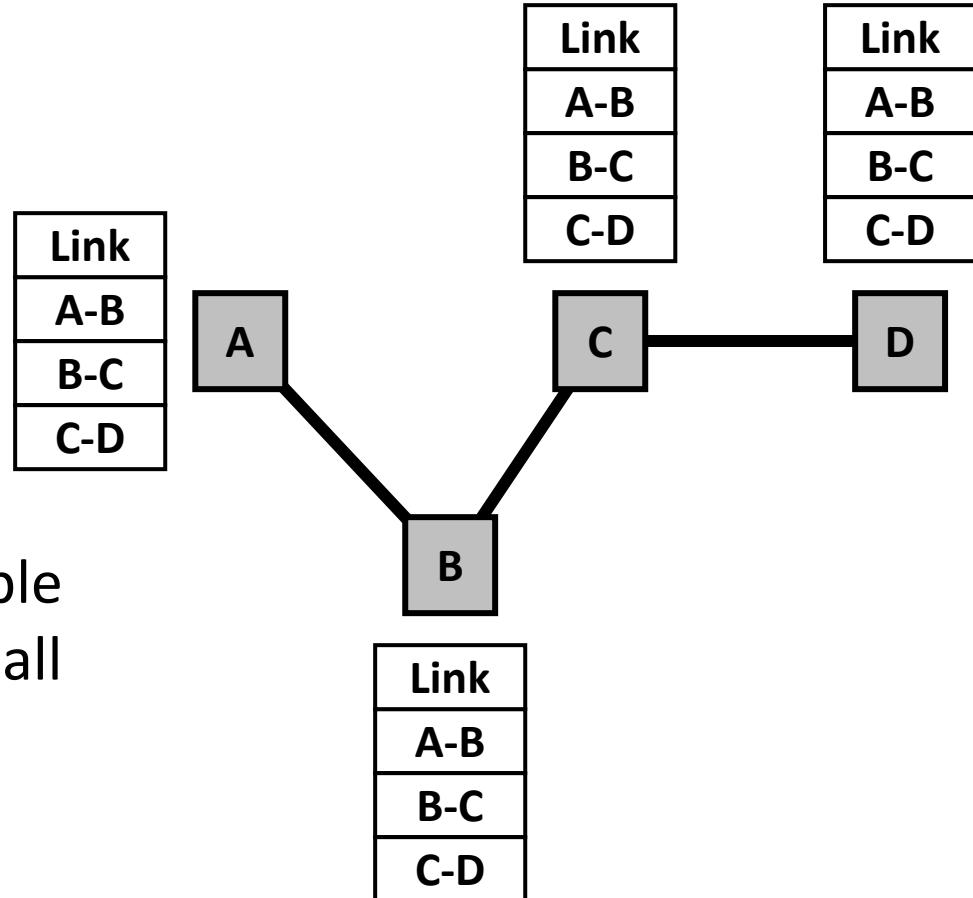


# Ad-Hoc Routing Protocol

- An ad-hoc routing protocol is a convention that controls how nodes decide which way to route packets between computing devices in a mobile ad-hoc network
- Foundation in most protocols: **neighbor discovery**
  - Nodes send periodic announcements as broadcast packets (beacon messages, alive messages, ...)
  - Can embed “neighbor table” into such messages; allows nodes to learn “2-hop neighborhood”
- Popular types of routing protocols:
  - **Proactive**
  - **Reactive**
  - **Geographic**

# Proactive: “Link-State” Algorithms

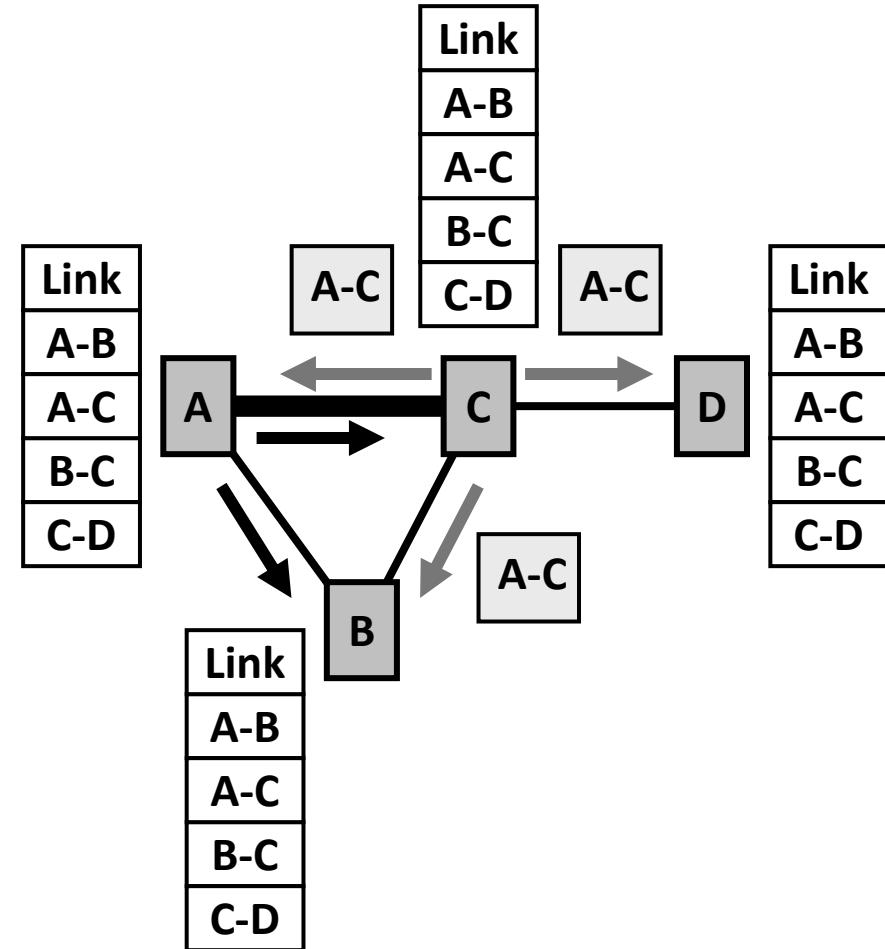
- Each node shares its link information so that all nodes can build a map of the full network topology



- Assuming the topology is stable for a sufficiently long period, all nodes will have the same topology information

# Proactive: “Link-State” Algorithms

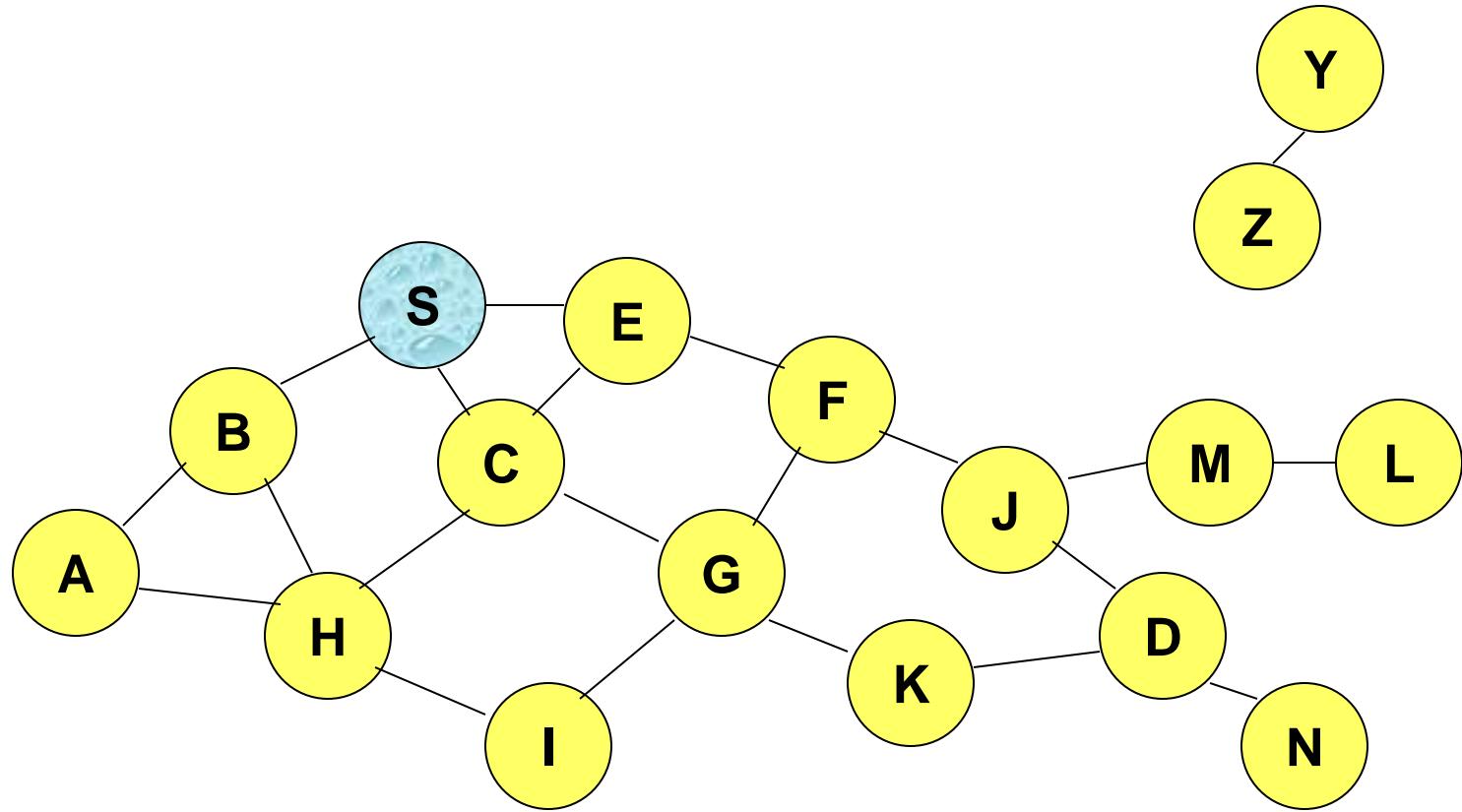
- Link information is updated when a link changes state (goes up or down)
  - by sending small “hello” packets to neighbors
- Nodes A and C propagate the existence of link A-C to their neighbors and, eventually, to the entire network



## Reactive: DSR

- **Dynamic Source Routing**
- Search for route when needed only
  - Search using **Route Request (RREQ)** broadcasts
  - Response using **Route Reply (RREP)** message
- Every message along route contains entire path to help intermediate nodes to decide what to do with message

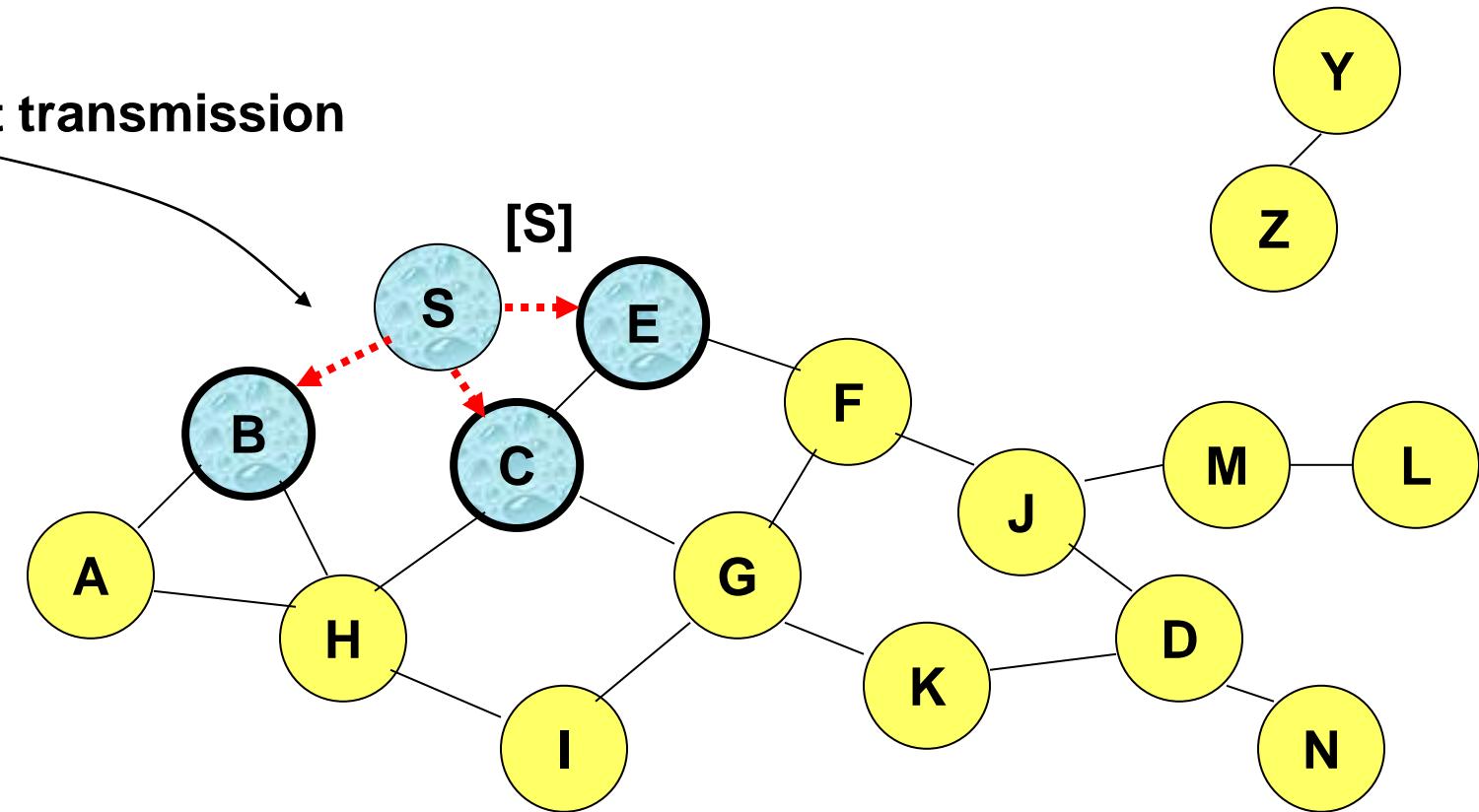
# Route Discovery in DSR



Represents a node that has received RREQ for D from S

# Route Discovery in DSR

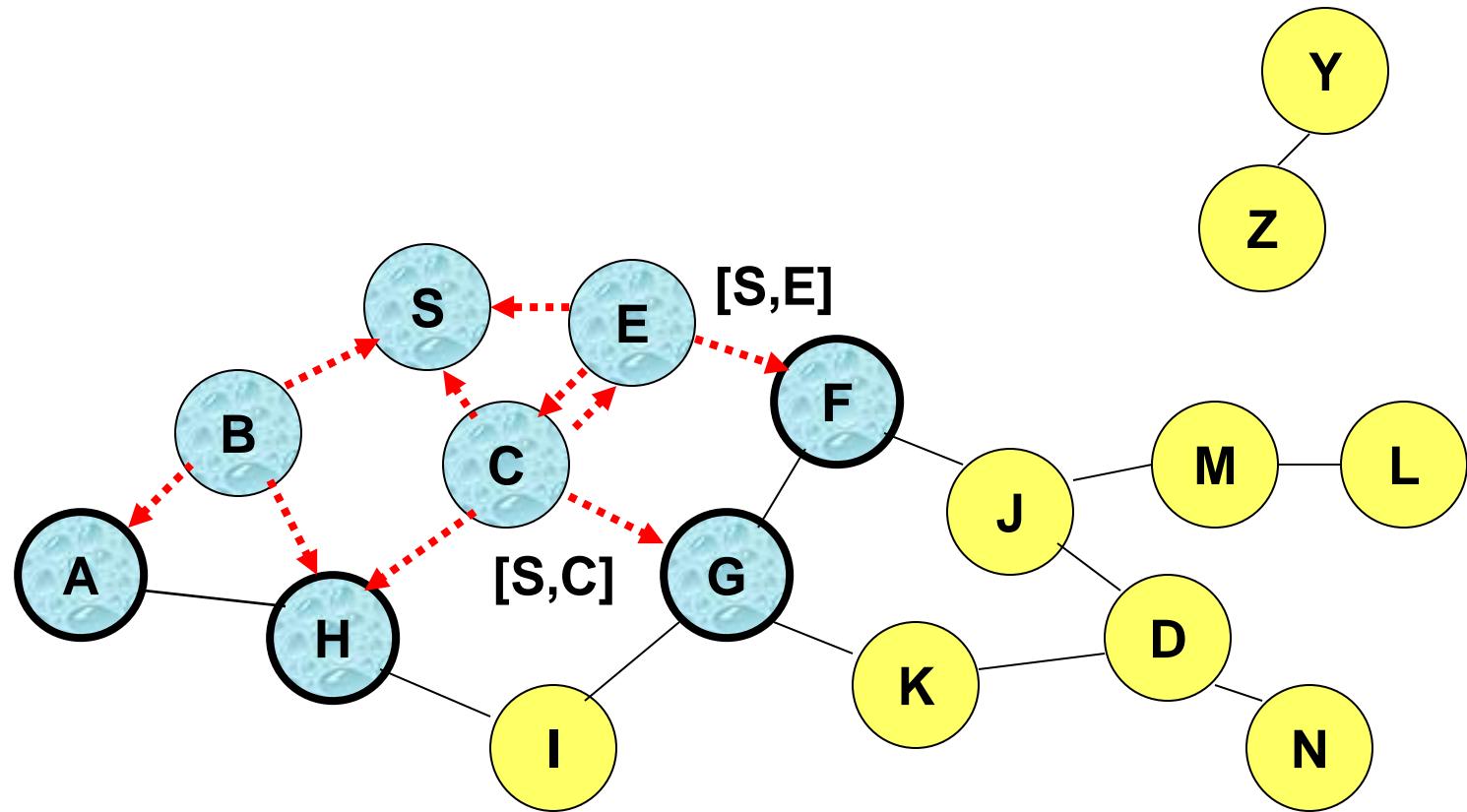
Broadcast transmission



-----> Represents transmission of RREQ

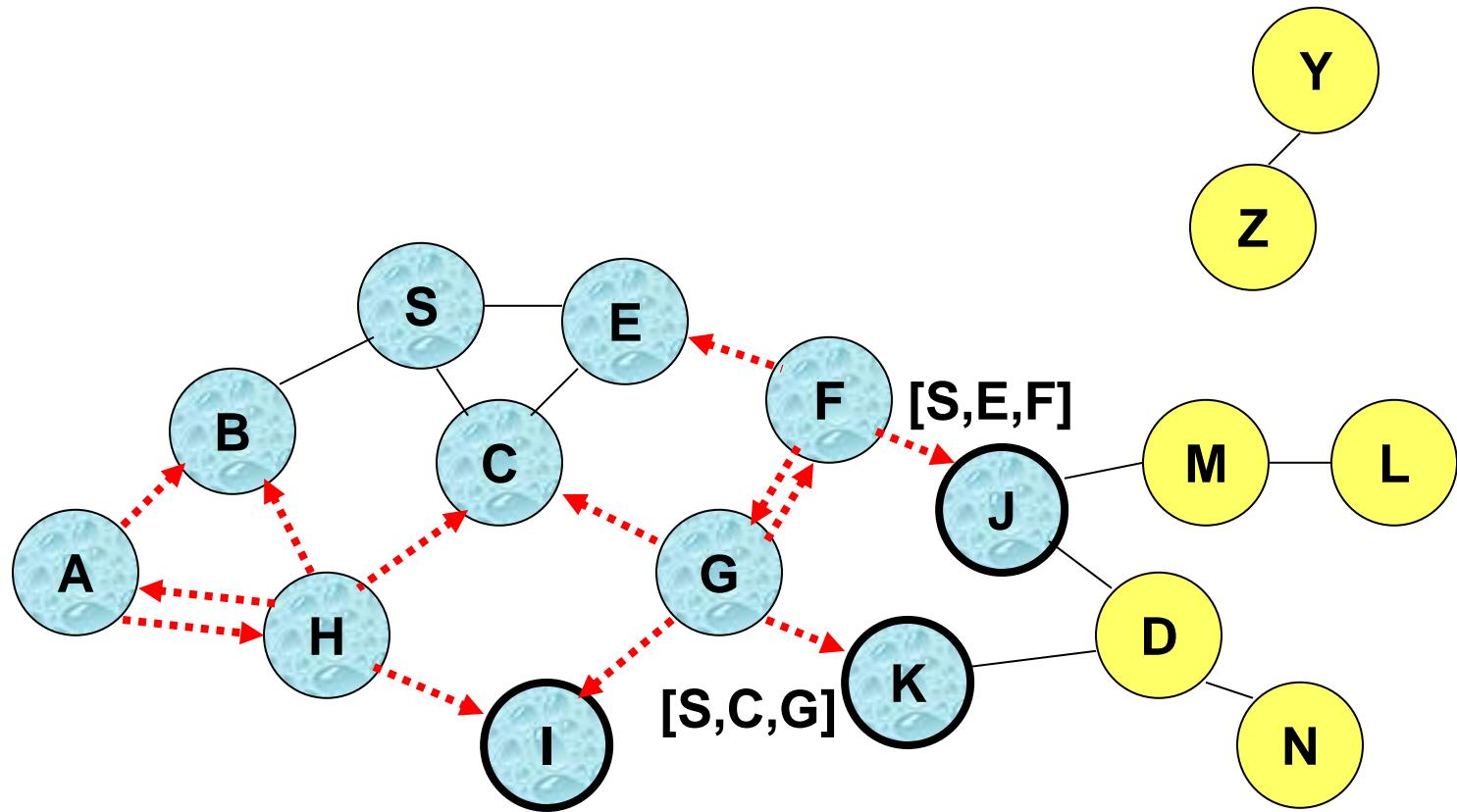
[X,Y] Represents list of identifiers appended to RREQ

# Route Discovery in DSR



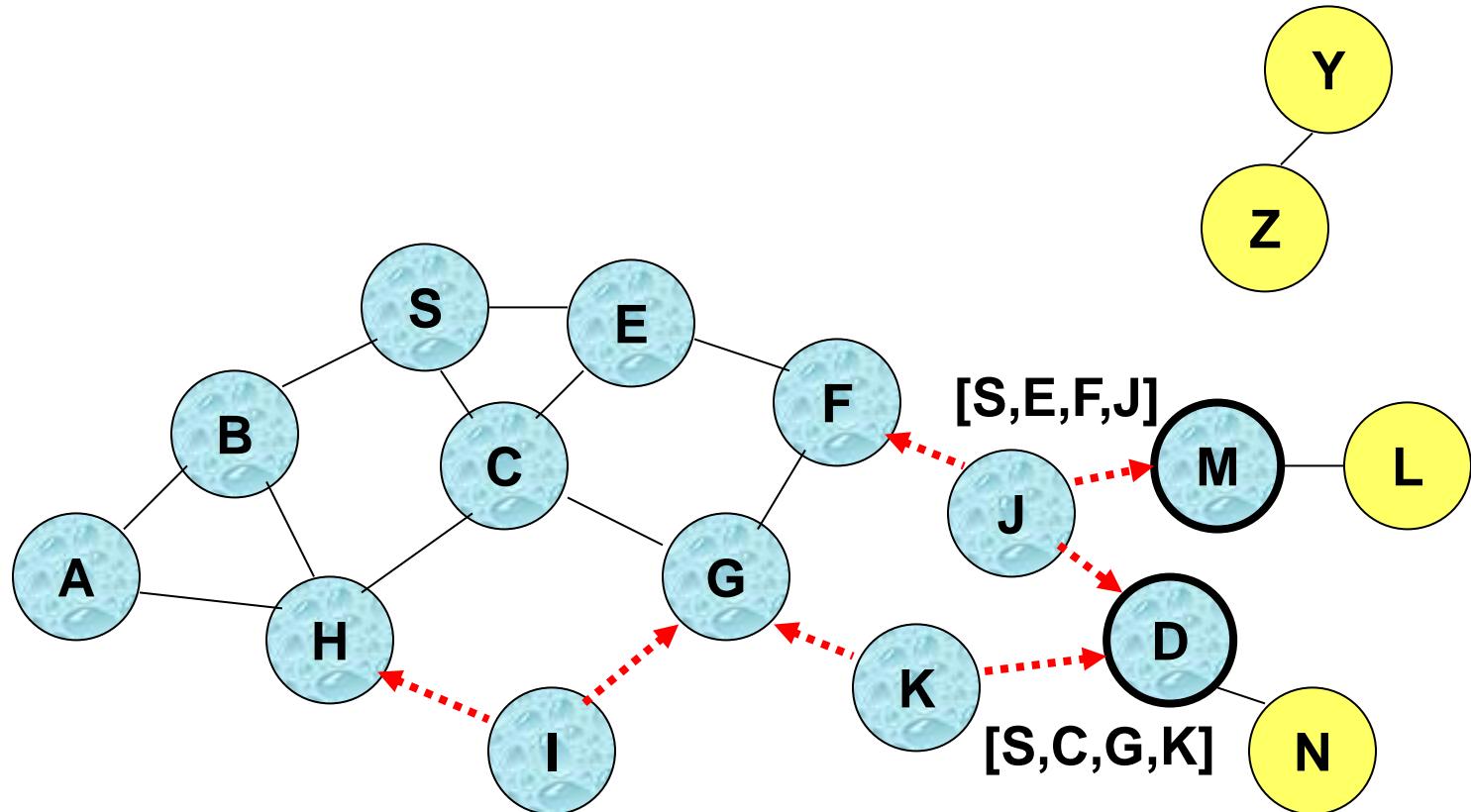
- Node H receives packet RREQ from two neighbors:  
**potential for collision**

# Route Discovery in DSR



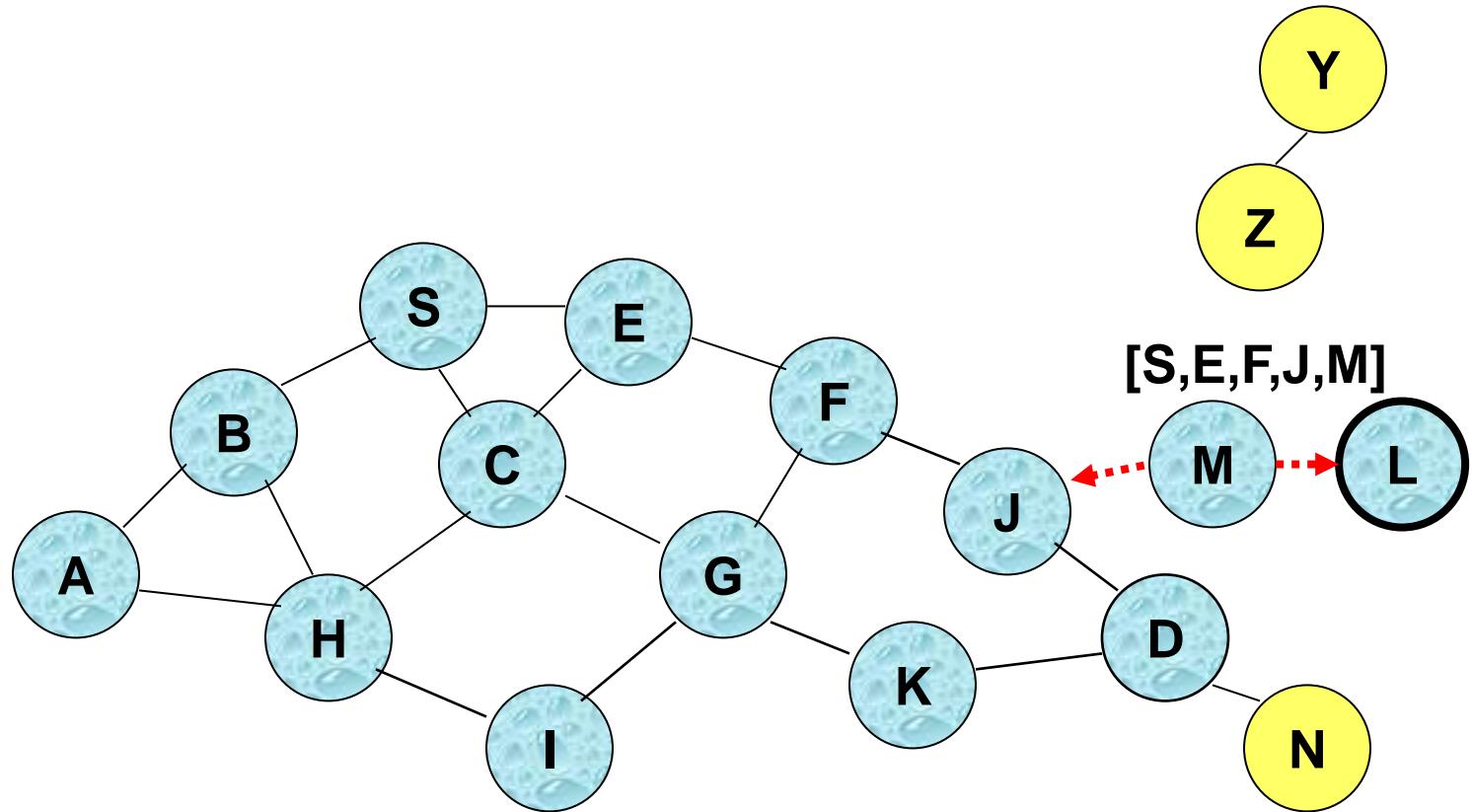
- Node C receives RREQ from G and H, but does not forward it again, because node C has **already forwarded RREQ once**

# Route Discovery in DSR



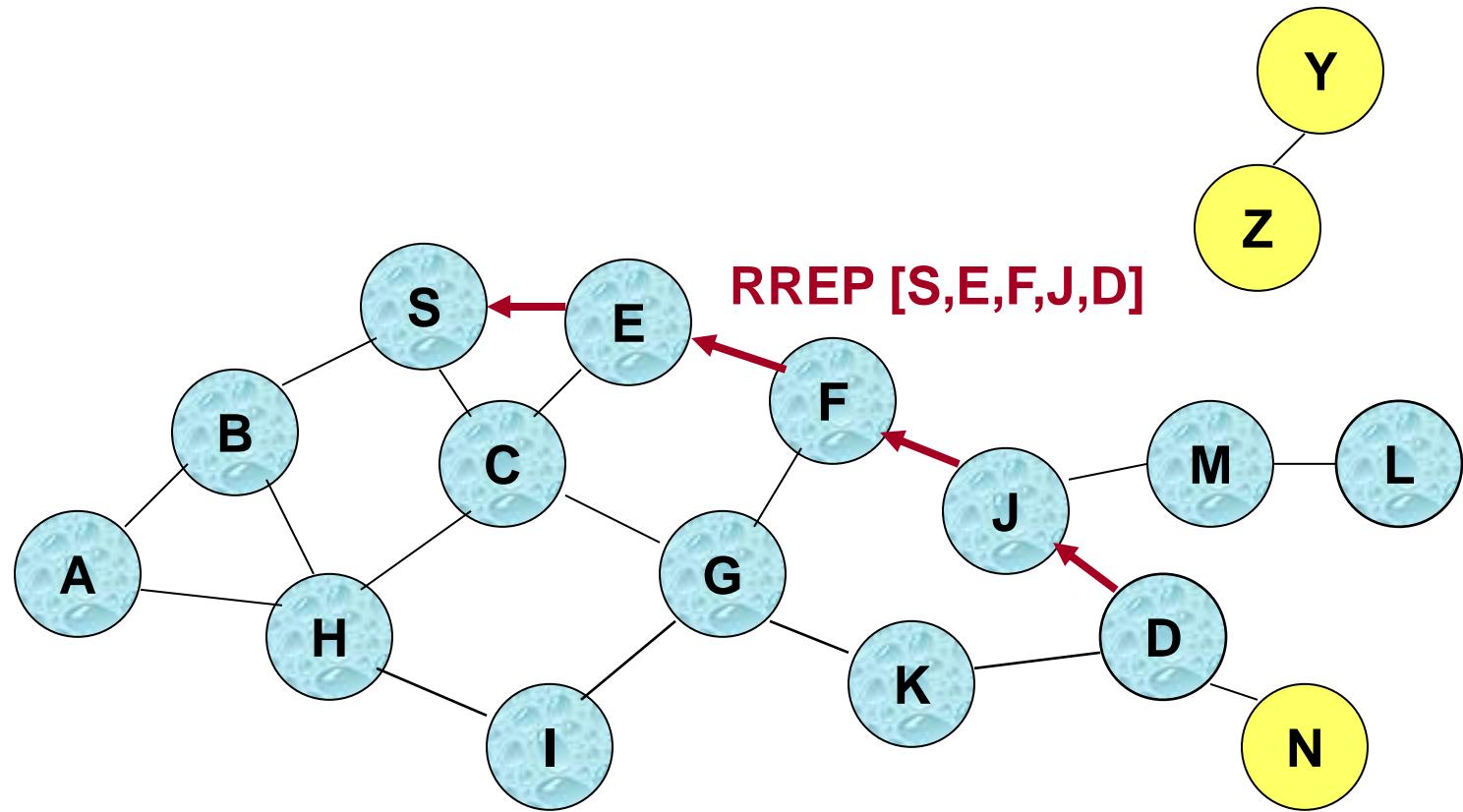
- Nodes J and K both broadcast RREQ to node D
- Since nodes J and K are **hidden** from each other, their **transmissions may collide**

# Route Discovery in DSR

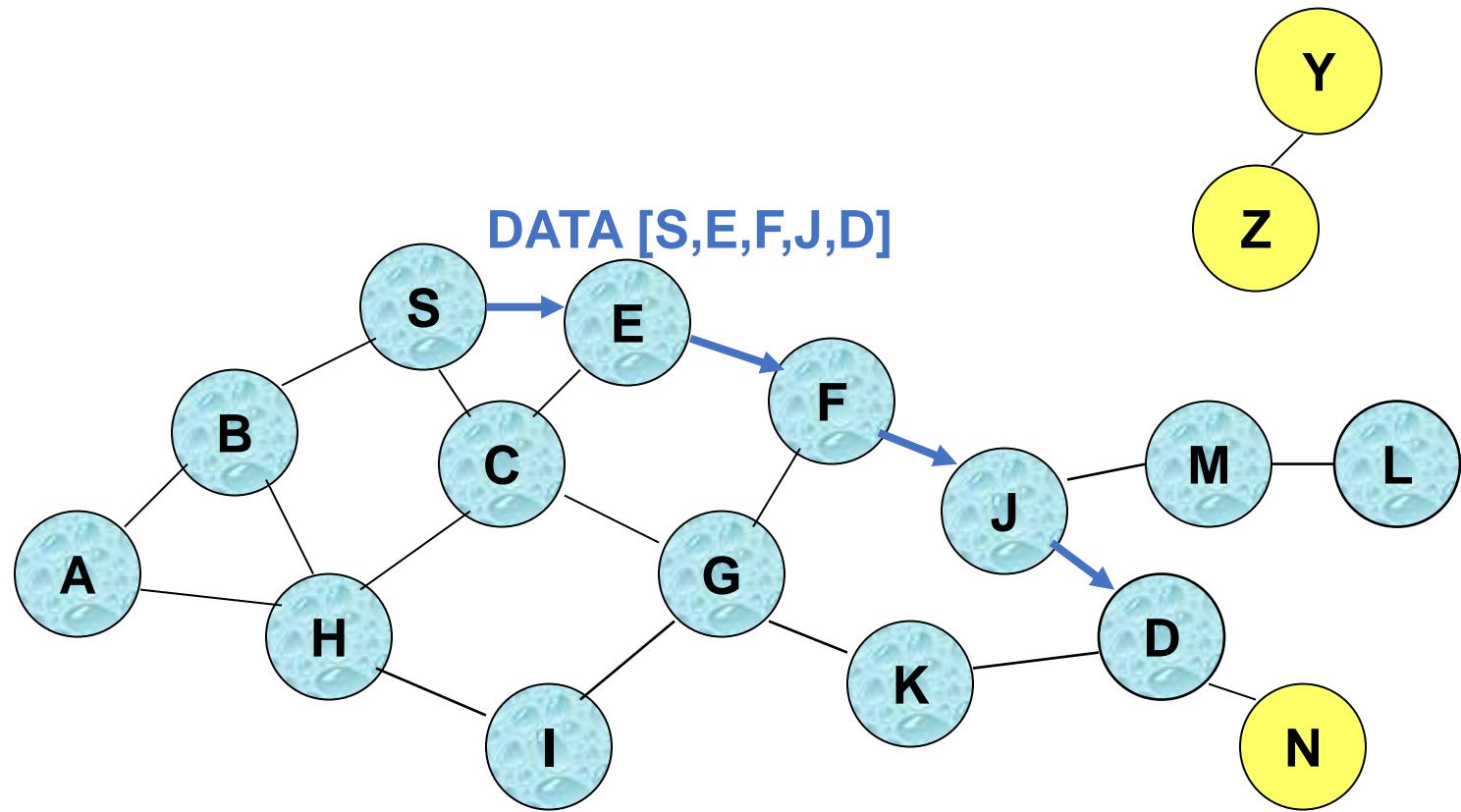


- Node D **does not forward RREQ**, because node D is the **intended target** of the route discovery

# Route Reply in DSR



# Data Delivery in DSR



**Packet header size grows with route length**

# Proactive vs Reactive

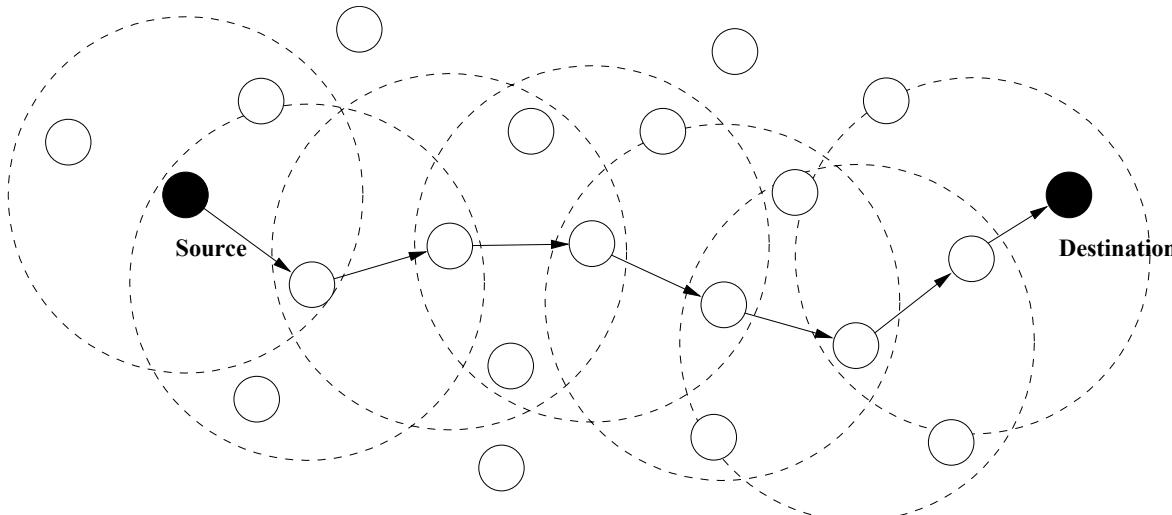
- Reactive:
  - Only establish/maintain routes between nodes needed them (in contrast: tables store ALL routes)
  - Store entire route in each message; message size grows with route length
  - Route requests cause “flooding”
- Proactive:
  - Route information always available; no need to search for route (but route information can be outdated)
  - Continuous exchange of route change updates

# Geographic Routing

- Nodes use location information to make routing decisions
  - sender must know the locations of itself, the destination, and its neighbors
  - location information can be queried or obtained from a **location broker**
  - location information can come from GPS (Global Positioning System) or some other form of positioning technology.

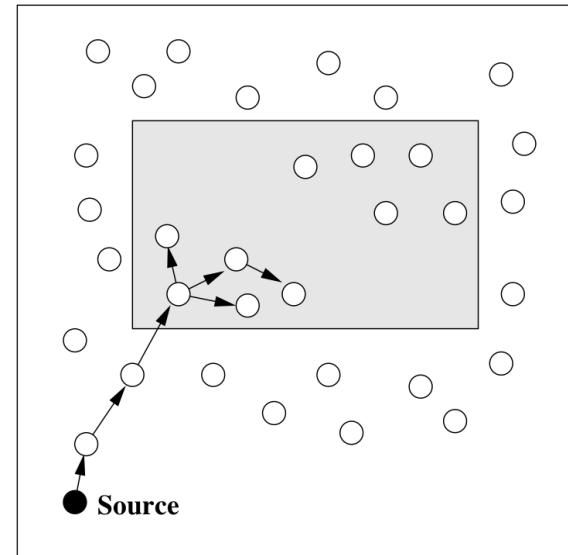
# Unicast Location-Based Routing

- One single destination
- Each forwarding node makes localized decision based on the location of the destination and the node's neighbors (**greedy forwarding**)
- Challenge: packet may arrive at a node without neighbors that could bring packet closer to the destination (**voids or holes**)



# Geocasting

- Packet is sent to all or some nodes within specific geographic region
- Example: query sent to all sensors within geographic area of interest
- Routing challenge:
  - propagate a packet near the target region (similar to unicast routing)
  - distribute packet within the target region (similar to flooding)



# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

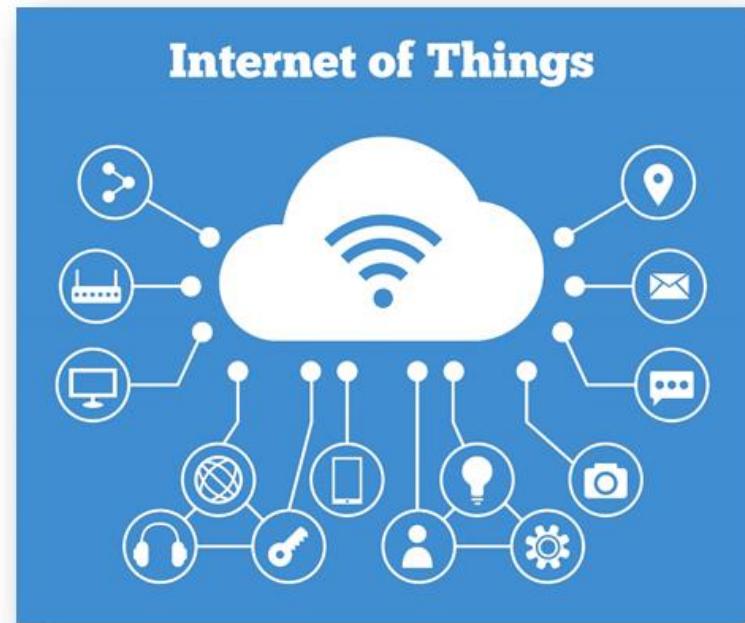
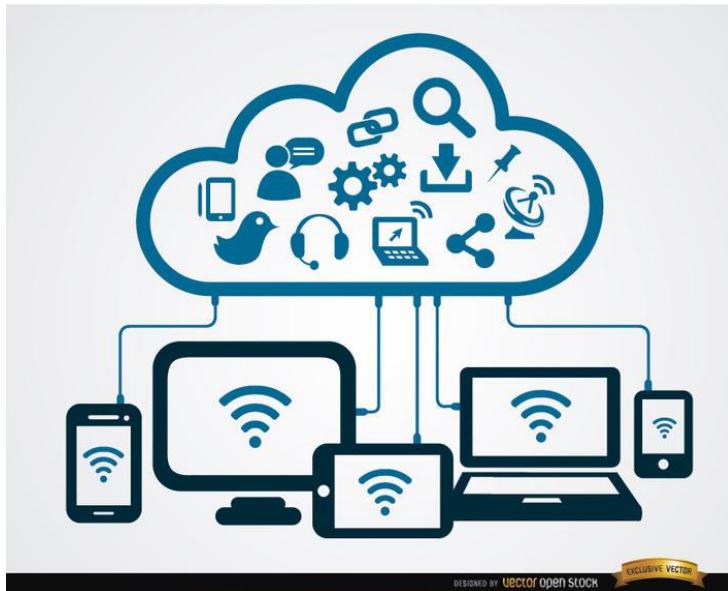
LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# HTTP & REST

COCSC20

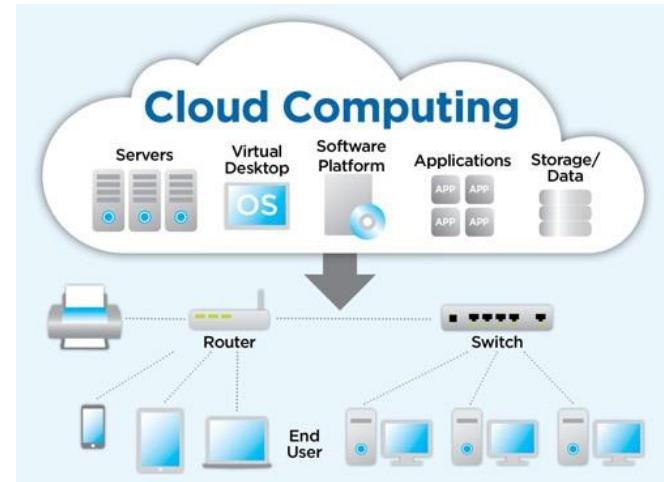
# What is Cloud Computing?



# What is Cloud Computing?

- Delivery of computing **services**

- servers
- storage
- analytics
- databases
- networking
- and much more...



- Another definition: network-based computing taking place over the Internet, while hiding complexity of underlying infrastructure using simple APIs.

# What is Cloud Computing?

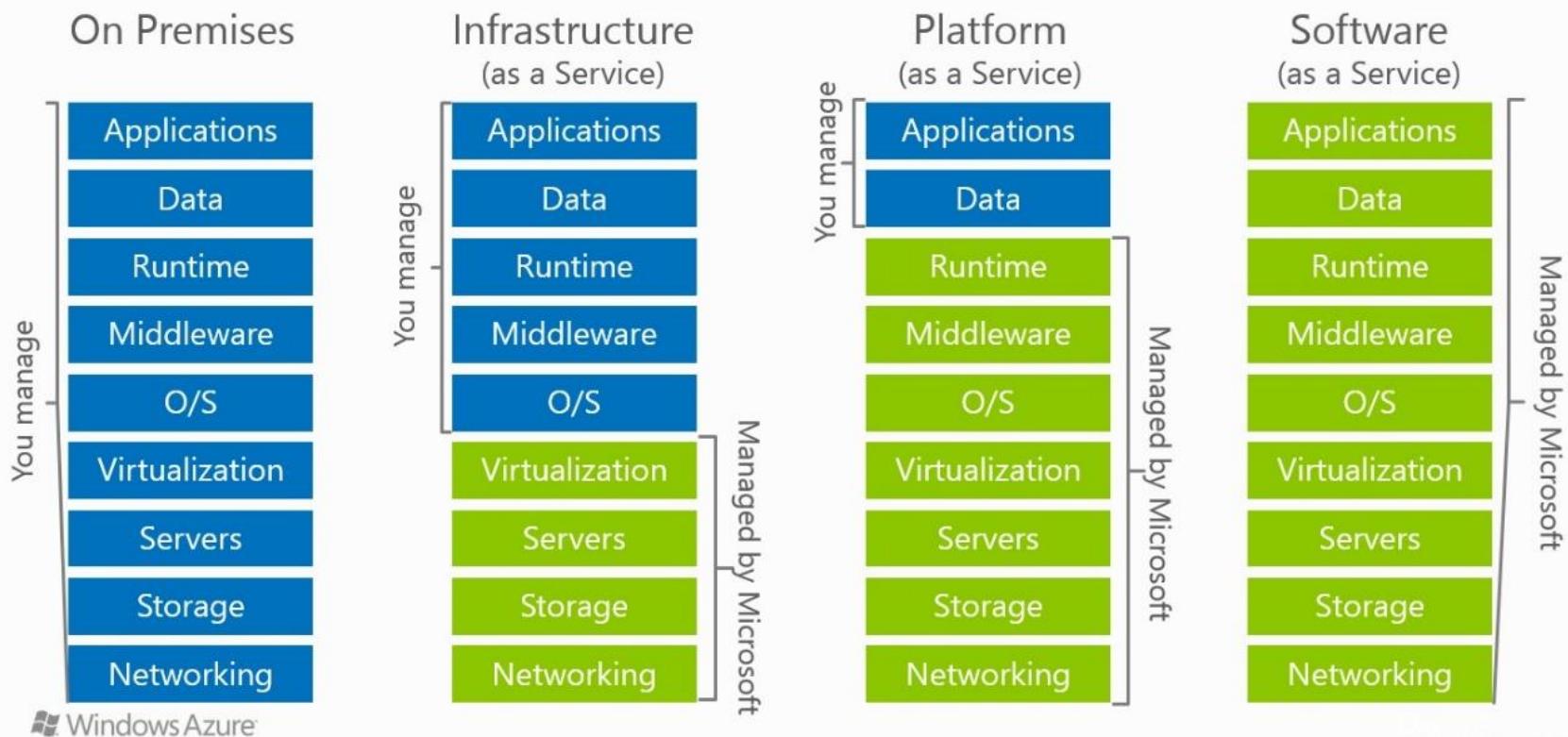
- Collection/group of **integrated and networked hardware, software, and Internet infrastructure** (called a platform)
- Platforms provide **on demand** services that are **always on** and **accessible anytime and anywhere**

# What is Cloud Computing?

- Advantages:
  - New applications
  - Anytime/anywhere access
  - Homogeneity
  - Virtualization
  - Resilient
  - Cost
  - Sharing, collaboration
  - Management/maintenance
  - Security
  - ...

# Cloud Models: IaaS, PaaS, SaaS

## Cloud Models



# Definitions

- **Virtualization:** creation of a virtual resource such as a server, desktop, operating system, file, storage, or network
- **Middleware:** software that acts as a bridge between an operating system or database and applications, especially on a network
- **Runtime:** software designed to support the execution of computer programs

# IaaS, PaaS, SaaS

## Software as a Service (SaaS)

Enduser application is delivered as a service. Platform and infrastructure is abstracted, and can be deployed and managed with less effort.

## Platform as a Service (PaaS)

Application platform onto which custom applications and services can be deployed. Can be built and deployed more inexpensively, although services need to be supported and managed.

## Infrastructure as a Service (IaaS)

Physical infrastructure is abstracted to provide computing, storage, and networking as a service, avoiding the expense and need for dedicated systems.

Simple example:

- IaaS: barebones computer
- PaaS: computer + OS (incl. development environment)
- SaaS: complete solution including application(s)

# IaaS, PaaS, SaaS

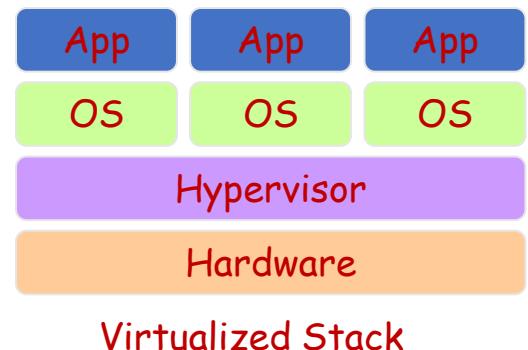
- IaaS: Amazon Web Services (AWS), Microsoft Azure, Google Compute Engine
- PaaS: Google App Engine, Heroku, OpenShift, AWS Elastic Beanstalk
- SaaS: Google Apps, Dropbox, Cisco Webex, Salesforce, Concur, GoToMeeting

# Basic Cloud Characteristics

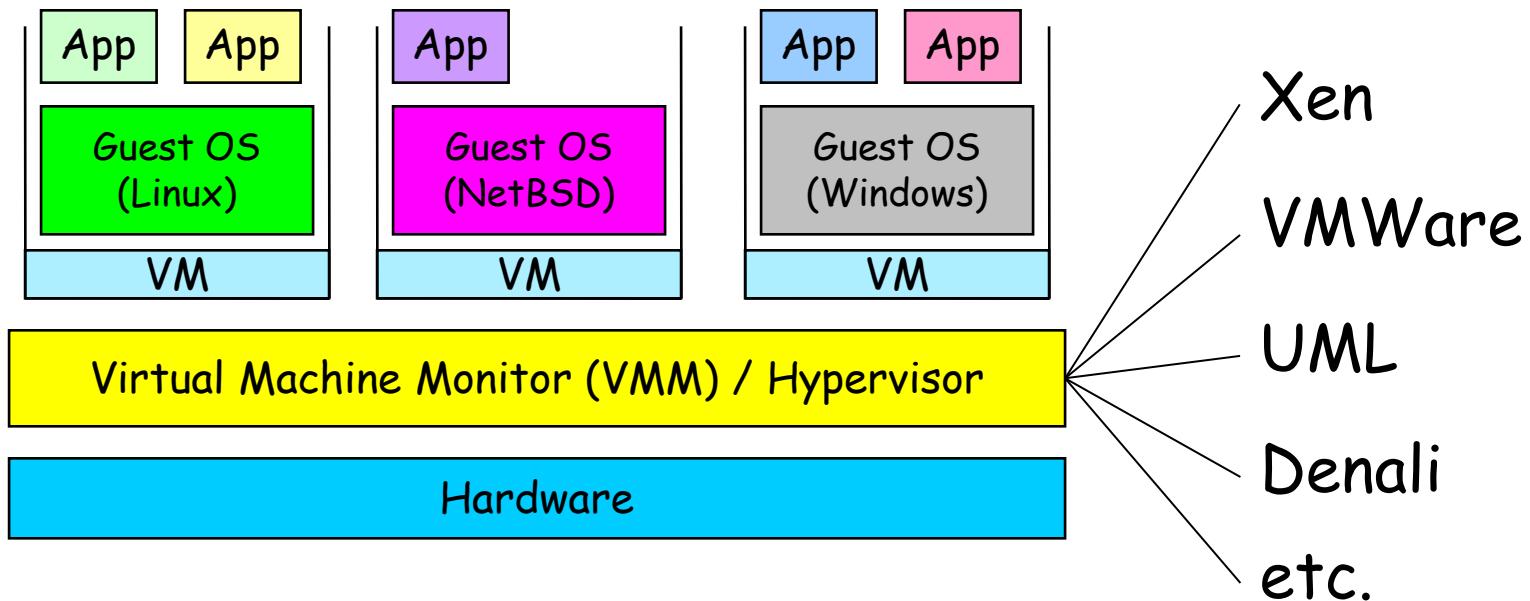
- “No-need-to-know”: interact with underlying infrastructure via API
- Flexibility and elasticity: scale systems up and down (allocate/release resources) based on needs
- Pay as much as used and needed (actual usage vs. service levels)
- Anytime anywhere access

# Virtualization

- Virtual workspaces:
  - An abstraction of an execution environment that can be made dynamically available to authorized clients by using well-defined protocols
  - Resource quota (e.g., CPU, memory share)
  - Software configuration (e.g., OS, provided services)
- Implemented on Virtual Machines (VMs):
  - Abstraction of a physical host machine
  - Hypervisor intercepts and emulates instructions from VMs, and allows management of VMs
  - VMWare, Xen, etc.



# Virtual Machines



# Cloud Example: S3

- Amazon Simple Storage Service (S3)
- Unlimited storage
- Pay for what you use

	S3 Standard	S3 Standard – Infrequent Access	AWS Glacier
<strong>STORAGE</strong>			
First 50 TB/ month	\$0.023 / GB	\$0.0125 / GB	\$0.004 / GB
Next 450 TB/ month	\$0.022 / GB	\$0.0125 / GB	\$0.004 / GB
Over 500 TB/ month	\$0.021 / GB	\$0.0125 / GB	\$0.004 / GB
<strong>REQUESTS</strong>			
PUT, COPY, POST, or LIST	\$0.005 / 1,000 requests	\$0.01 / 1,000 requests	
GET and all other requests	\$0.004 / 10,000 requests	\$0.01 / 10,000 requests	
Delete requests	Free	Free	Free, but with limits and potential surcharges
Lifecycle Transition Requests into S3 Standard IA		\$0.01 / 1,000 requests	
Glacier archive and restore requests			\$0.05 / 1,000 requests, see <a href="#">Glacier pricing</a> for more details on retrieval fees

# Cloud Example: EC2

- Amazon Elastic Compute Cloud (EC2)
  - Virtual computing environments (“instances”)
  - Pre-configured templates for instances
  - Launch as many virtual servers as needed (“elastic”)
  - Xen and KVM hypervisor

# Do You Use The Cloud?



# Cloud for IoT

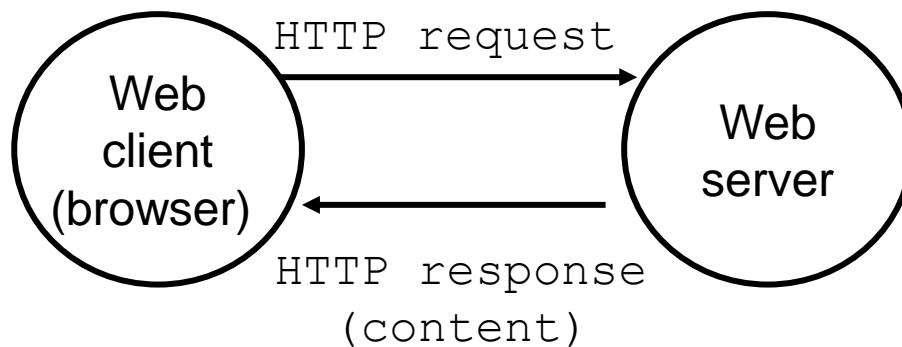


Microsoft Azure  
IoT Platform

IBM **Watson IoT**™

# HyperText Transfer Protocol (HTTP)

- Clients and servers communicate using the **HyperText Transfer Protocol (HTTP)**
  - Client and server establish TCP connection
  - Client requests content
  - Server responds with requested content
  - Client and server close connection (usually)



# Web Content

- Web servers return **content** to clients
    - a sequence of bytes with an associated MIME (Multipurpose Internet Mail Extensions) type
  - Example MIME types
    - text/html HTML document
    - text/plain Unformatted text
    - application/postscript Postscript document
    - image/gif Binary image encoded in GIF format
    - image/jpeg Binary image encoded in JPEG format

# Static & Dynamic Content

- The content returned in HTTP responses can be either **static** or **dynamic**
  - Static content: content stored in files and retrieved in response to an HTTP request
    - Examples: HTML files, images, audio clips
  - Dynamic content: content produced on-the-fly in response to an HTTP request
    - Example: content produced by a program executed by the server on behalf of the client
- Bottom line: all web content is associated with a **file** that is managed by the server

# URLs

- Each file managed by a server has a unique name called a URL (Universal Resource Locator)
- URLs for static content:
  - `http://www.cse.nd.edu:80/index.html`
  - `http://www.cse.nd.edu/index.html`
  - `http://www.cse.nd.edu`
    - Identifies a file called `index.html`, managed by a web server at `www.cse.nd.edu` that is listening on port 80
- URLs for dynamic content:
  - `http://www.cse.nd.edu:8000/cgi-bin/adder?15000&213`
    - Identifies an executable file called `adder`, managed by a web server at `www.cse.nd.edu` that is listening on port 8000, that should be called with two argument strings: `15000` and `213`

# Anatomy of an HTTP Transaction

```
<unix> telnet www.aol.com 80          Client: open connection to server
Trying 205.188.146.23...                Telnet prints 3 lines to the terminal
Connected to aol.com.
Escape character is '^]'.
GET / HTTP/1.1                          Client: request line
host: www.aol.com                        Client: required HTTP/1.1 HOST header
                                         Client: empty line terminates headers .
HTTP/1.0 200 OK                         Server: response line
                                         Server: followed by five response headers
MIME-Version: 1.0
Date: Mon, 08 Jan 2001 04:59:42 GMT
Server: NaviServer/2.0 AOLserver/2.3.3
Content-Type: text/html                  Server: expect HTML in the response body
Content-Length: 42092                    Server: expect 42,092 bytes in the resp body
                                         Server: empty line ("\\r\\n") terminates hdrs
<html>
...
</html>                                    Server: first HTML line in response body
                                         Server: 766 lines of HTML not shown.
                                         Server: last HTML line in response body
Connection closed by foreign host.      Server: closes connection
unix>                                    Client: closes connection and terminates
```

# HTTP Requests

- HTTP request is a *request line*, followed by zero or more *request headers*
- Request line:
  - <method> <uri> <version>
  - <version> is HTTP version of request (HTTP/1.0 or HTTP/1.1)
  - <uri> is typically URL for proxies, URL suffix for servers
  - <method> is either GET, POST, OPTIONS, HEAD, PUT, DELETE, or TRACE

# HTTP Requests

- **HTTP methods:**
  - GET: Retrieve static or dynamic content
    - Arguments for dynamic content are in URI
    - Workhorse method (99% of requests)
  - POST: Retrieve dynamic content
    - Arguments for dynamic content are in the request body
  - OPTIONS: Get server or file attributes
  - HEAD: Like GET but no data in response body
  - PUT: Write a file to the server
  - DELETE: Delete a file on the server
  - TRACE: Echo request in response body
    - Useful for debugging

# HTTP Responses

- HTTP response is a *response line* followed by zero or more *response headers*
- Response line:

```
<version> <status code> <status msg>
```

- <version> is HTTP version of the response
- <status code> is numeric status
- <status msg> is corresponding English text
  - 200 OK Request was handled without error
  - 403 Forbidden Server lacks permission to access file
  - 404 Not found Server couldn't find the file

- Response headers: <header name>: <header data>
  - Provide additional information about response
  - Content-Type: MIME type of content in response body
  - Content-Length: Length of content in response body

# REST

- Representational State Transfer (REST) is a design pattern.
- A style of software architecture for distributed hypermedia systems such as the World Wide Web
- A collection of network architecture principles which outline how **resources** are defined and addressed.
- It is a certain approach to creating Web Services.
- To understand the REST design pattern, let's look at an example (learn by example).

# Example:

## Airline Reservation Service

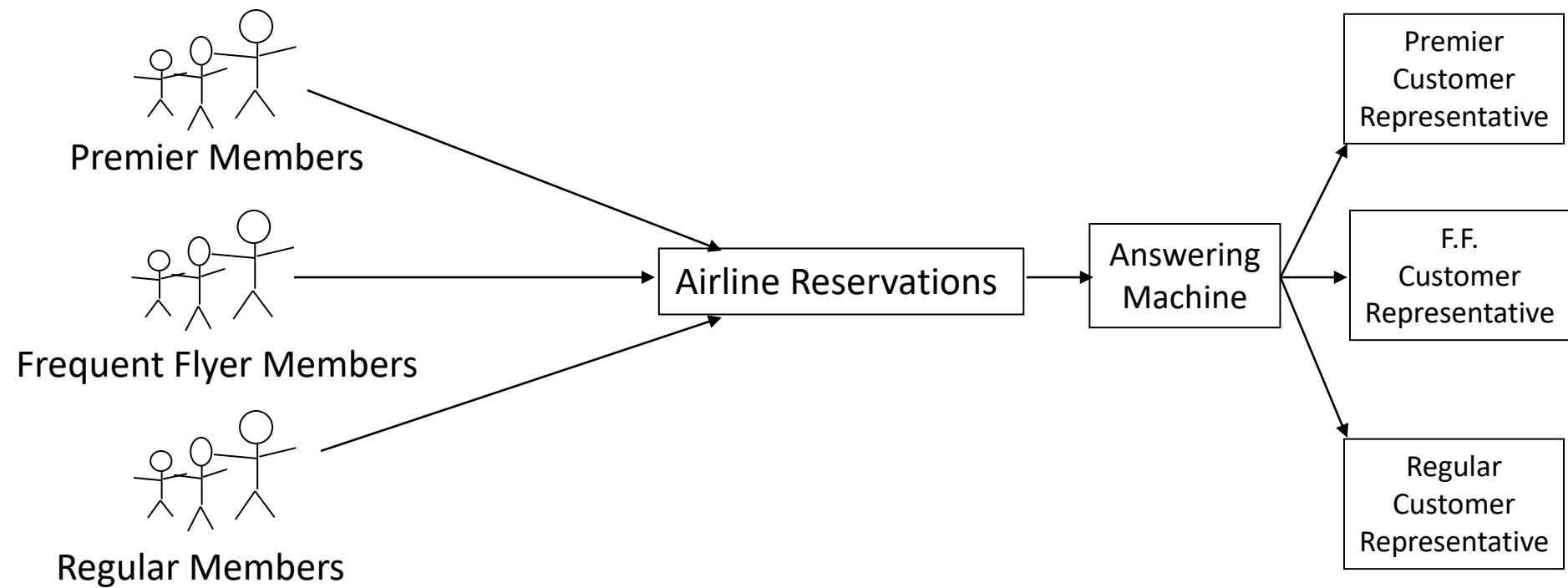
- Suppose that an airline wants to create a telephone reservation system for customers to call in and make flight reservations.
- The airline wants to ensure that its premier members get immediate service, its frequent flyer members get expedited service, and all others get regular service.
- There are two main approaches to implementing the reservation service...

# Approach 1

"Press 1 for Premier, Press 2 for..."

The airline provides a single telephone number.

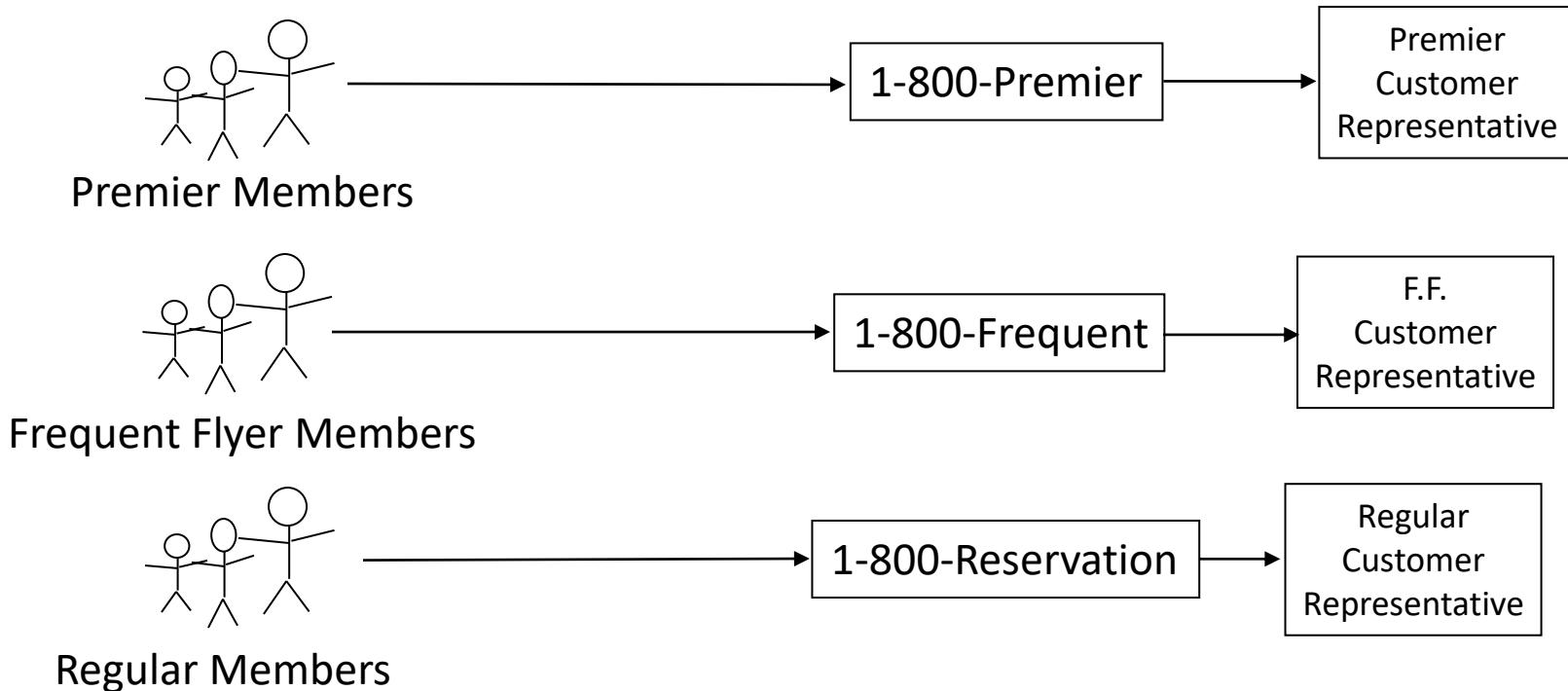
Upon entry into the system a customer encounters an automated message, "Press 1 if you are a premier member, press 2 if you are a frequent flyer, press 3 for all others."



# Approach 2

## Telephone Numbers are Cheap! Use Them!

The airline provides several telephone numbers - one number for premier members, a different number for frequent flyers, and still another for regular customers.



# Discussion

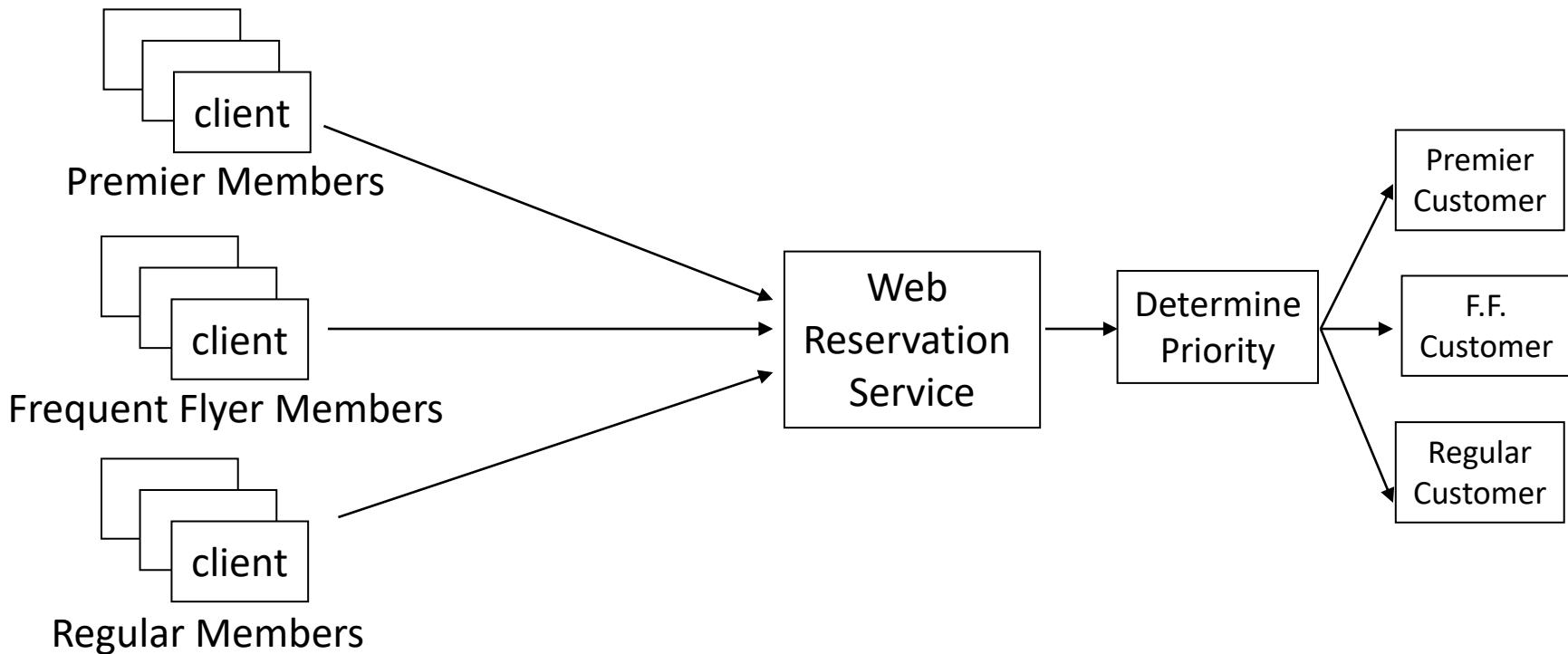
- In Approach 1 the answering machine introduces an extra delay, which is particularly annoying to premier members. (Doesn't everyone hate those answering systems)
- With Approach 2 there is no intermediate step. Premier members get instant pickup from a customer service representative. Others may have to wait for an operator.

# Web-Based Reservation Service

- Suppose now the airline ([kings-air.com](http://kings-air.com)) wants to provide a Web reservation service for customers to make flight reservations through the Web.
- Just as with the telephone service, the airline wants to ensure that its premier members get immediate service, its frequent flyer members get expedited service, all others get regular service.
- There are two main approaches to implementing the Web reservation service. The approaches are analogous to the telephone service...

# Approach 1 One-Stop Shopping

The airline provides a single URL. The Web service is responsible for examining incoming client requests to determine their priority and process them accordingly.



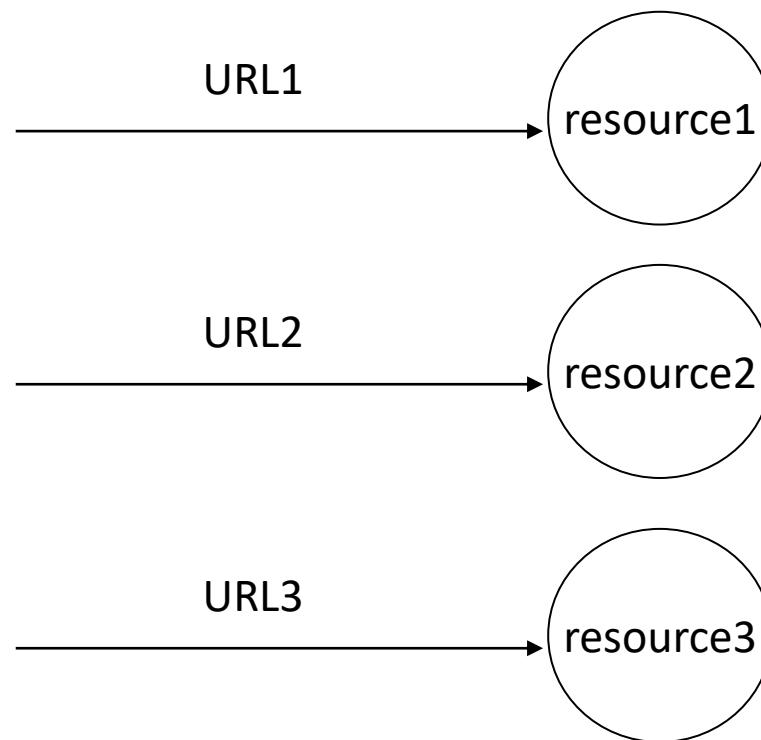
# Approach 1 Disadvantages

- There is currently no industry accepted practice (rules) for expressing priorities, so rules would need to be made. The clients must learn the rule, and the Web service application must be written to understand the rule.
- This approach is based upon the incorrect assumption that a URL is "expensive" and that their use must be rationed.
- The Web service is a central point of failure. It is a bottleneck. Load balancing is a challenge.
- It violates Tim Berners-Lee Web Design, Axiom 0 (see next slide).

# Web Design, Axiom 0

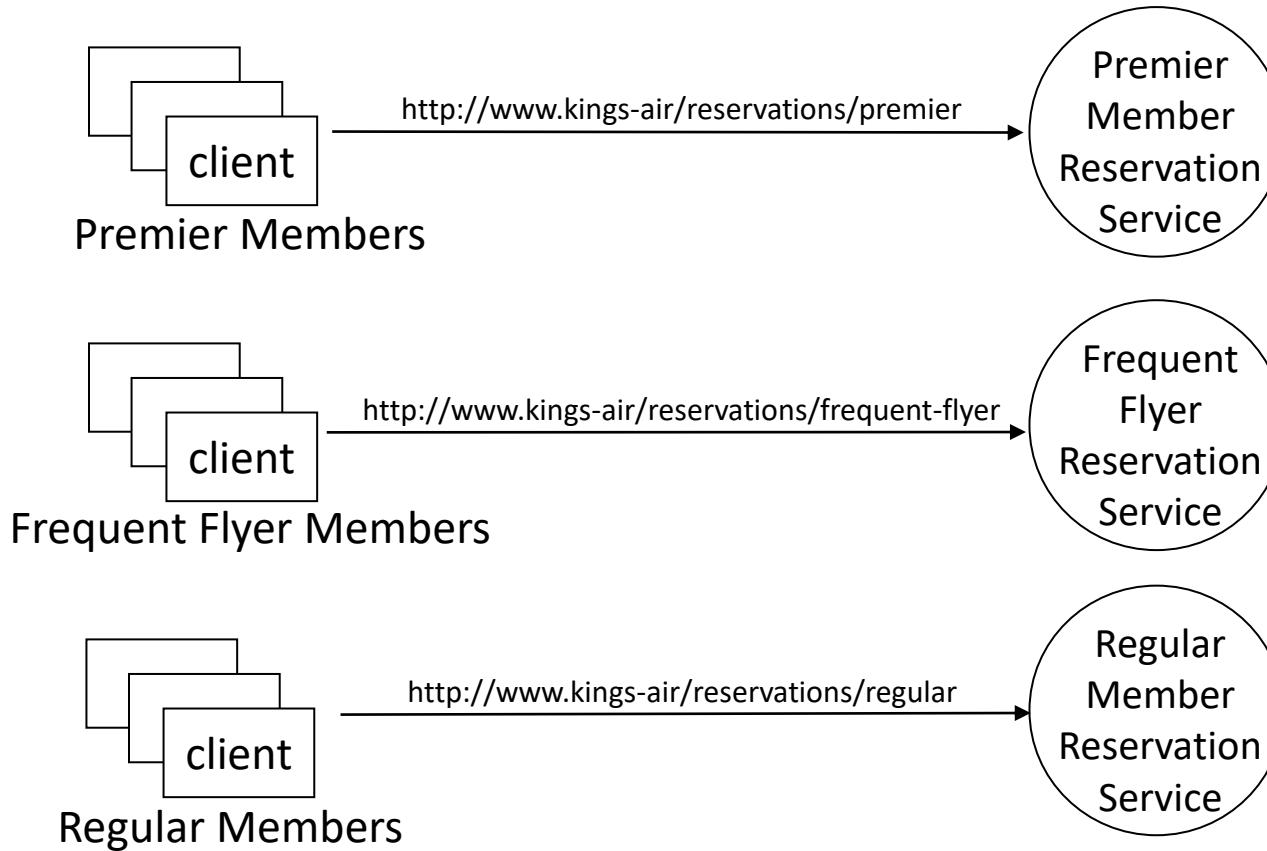
(Tim Berners-Lee, director of W3C)

- Axiom 0: all resources on the Web must be uniquely identified with a URI.



# Approach 2: URLs are Cheap! Use Them!

The airline provides several URLs - one URL for premier members, a different URL for frequent flyers, and still another for regular customers.



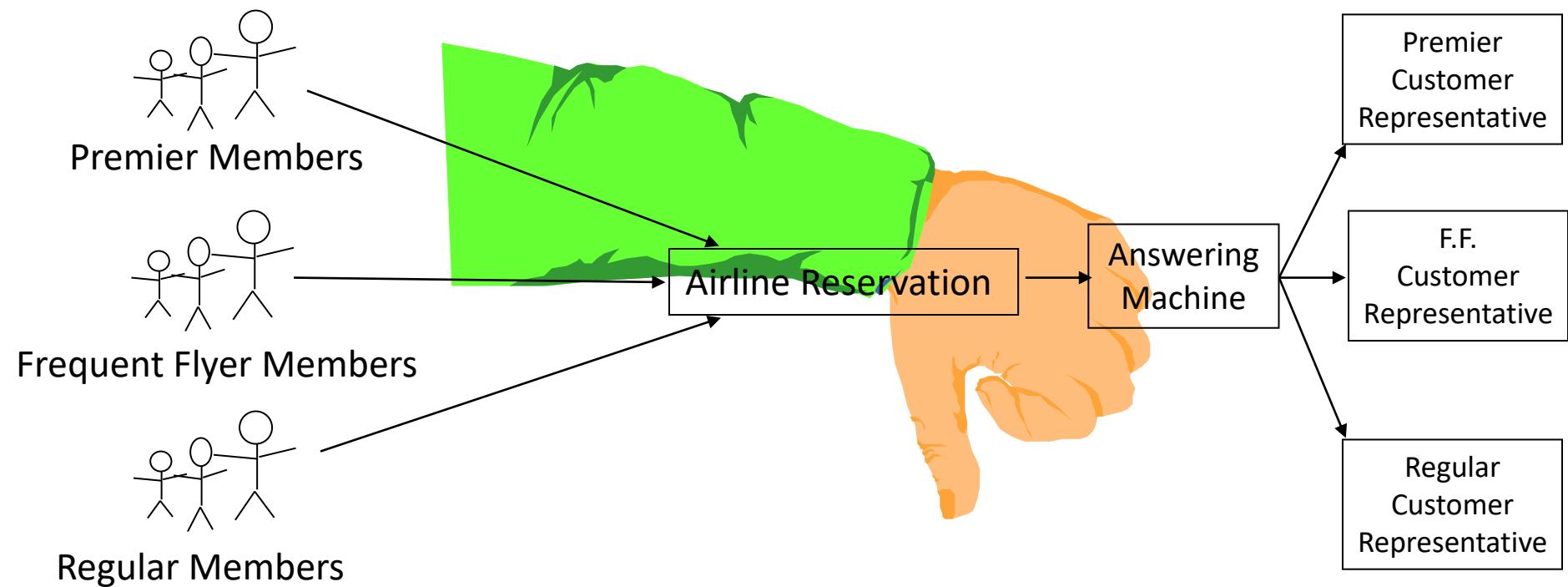
# Approach 2 Advantages

- The different URLs are discoverable by search engines and UDDI registries.
- It's easy to understand what each service does simply by examining the URL, *i.e.*, it exploits the Principle of Least Surprise.
- There is no need to introduce rules. Priorities are elevated to the level of a URL. "What you see is what you get."
- It's easy to implement high priority - simply assign a fast machine at the premier member URL.
- There is no bottleneck. There is no central point of failure.
- Consistent with Axiom 0.

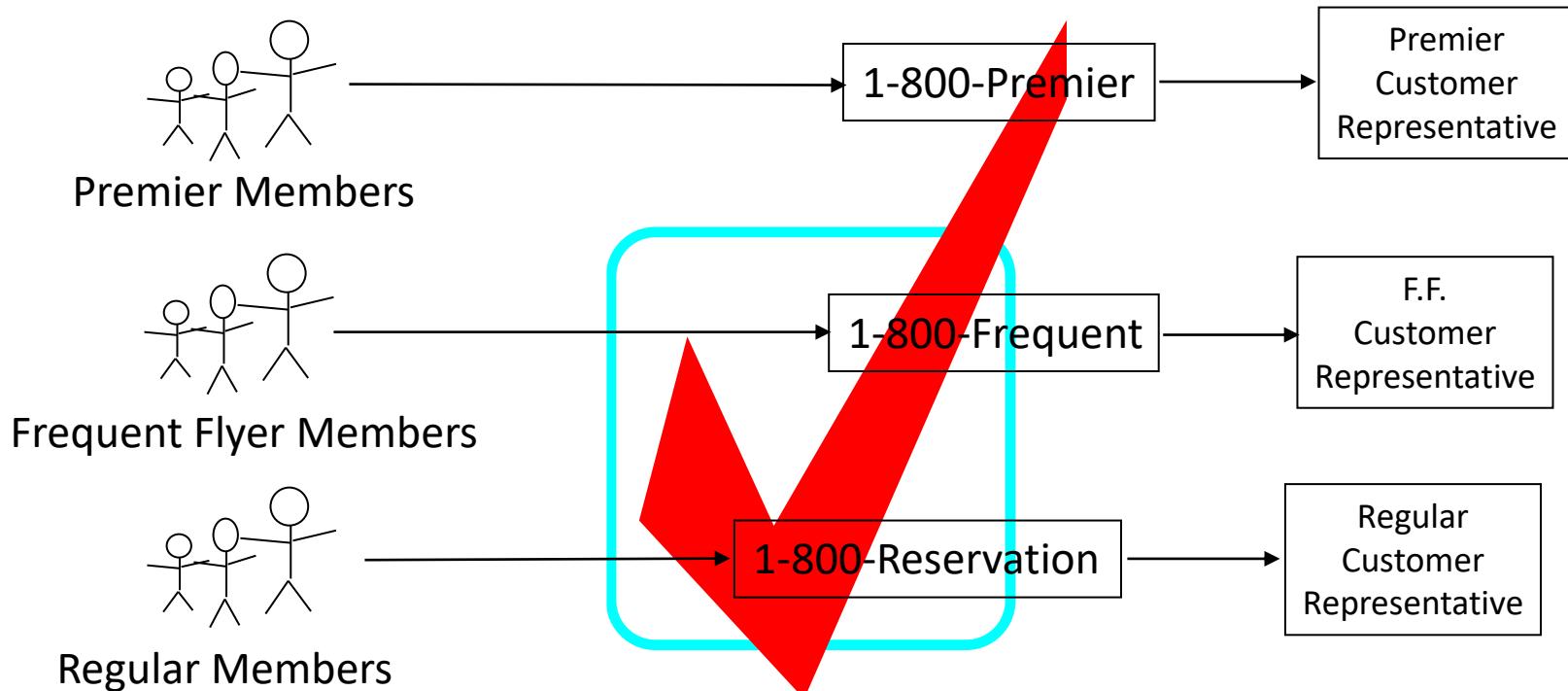
# Recap

- We have looked at a reservation service.
- We have seen a telephone-based version and a Web-based version of the reservation service.
- With each version we have seen two main approaches to implementing the service.
- Which approach is the REST design pattern and which isn't? See the following slides.

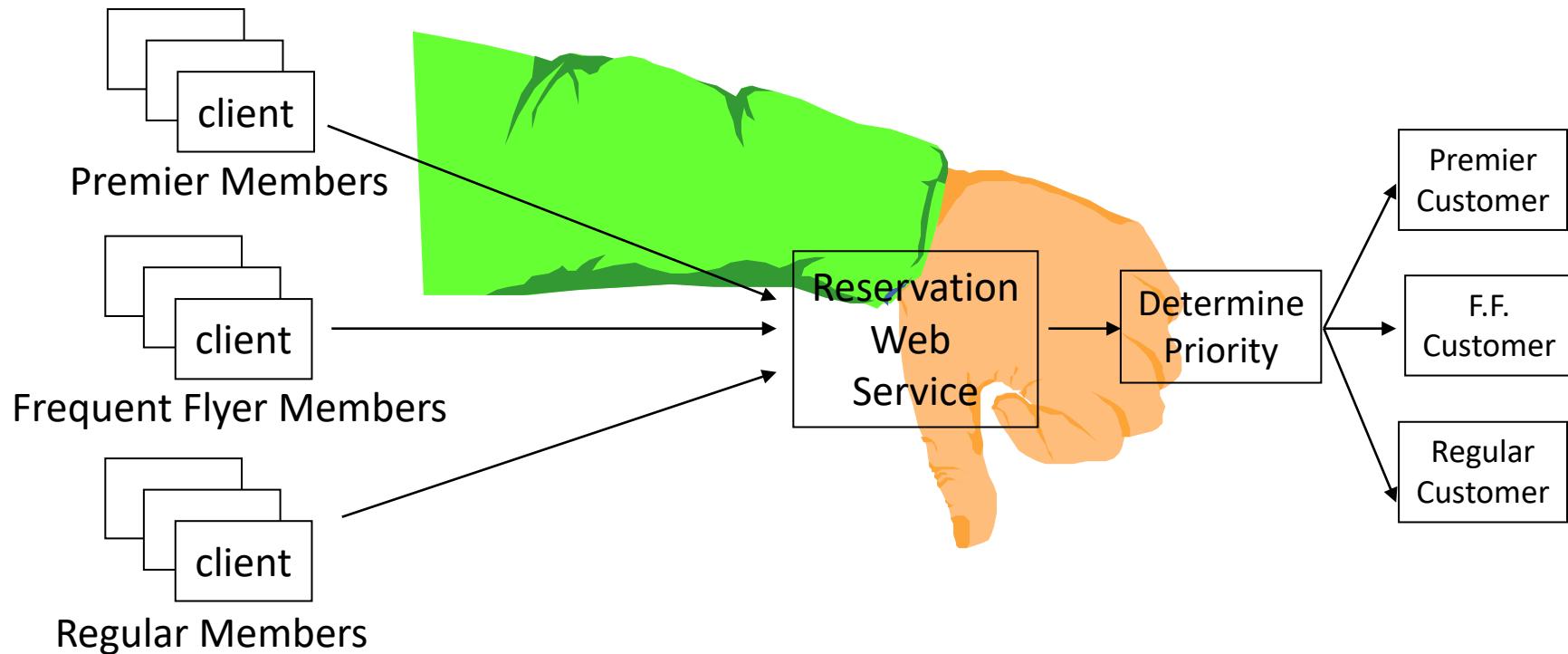
# This Ain't the REST Design Pattern



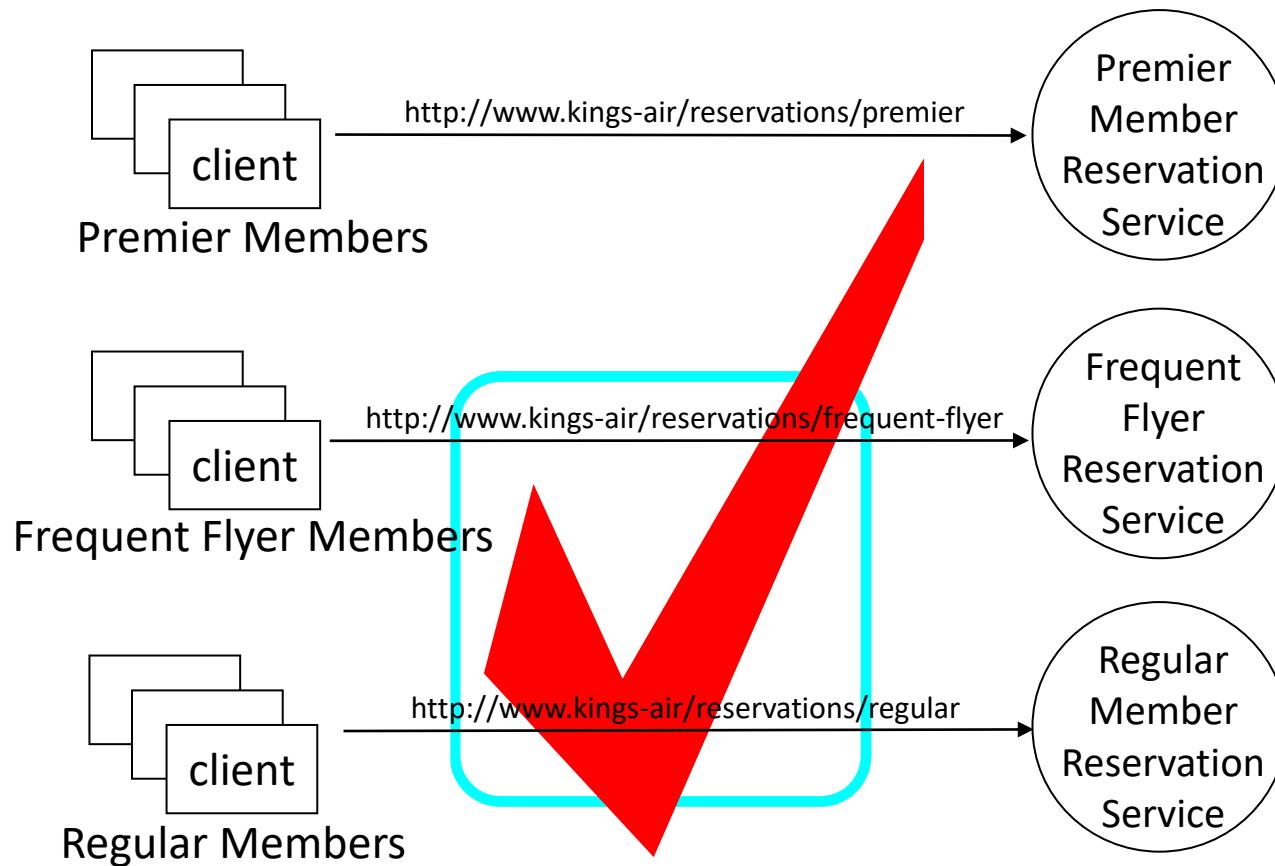
# This is the REST Design Pattern



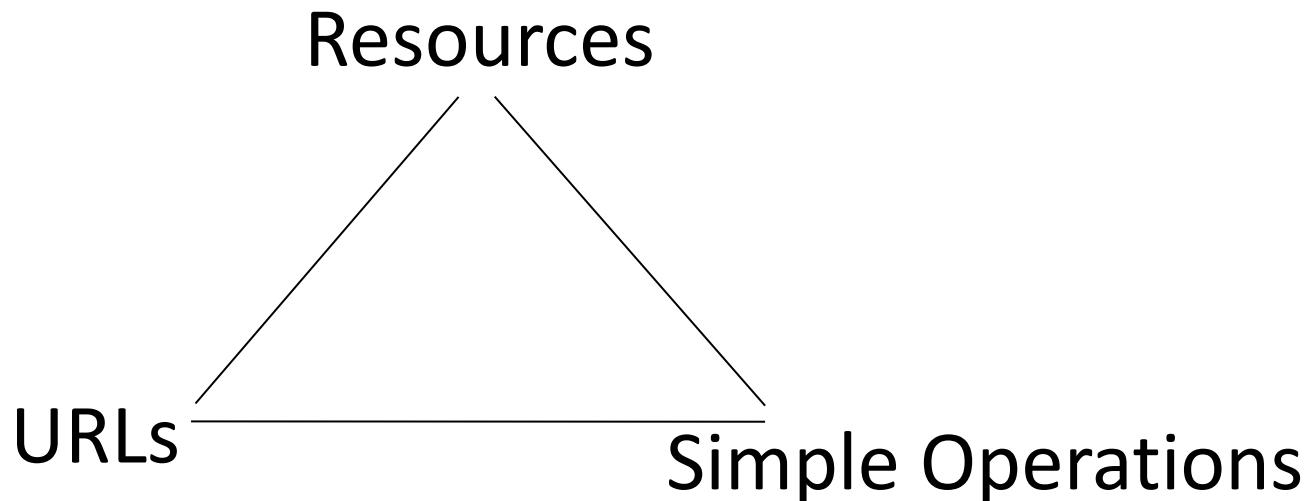
# This ain't the REST Design Pattern



# This is the REST Design Pattern



# The Three Fundamental Aspects of the REST Design Pattern



In this tutorial we discussed how Resources and URLs are fundamental to REST. In a follow up tutorial we will discuss how Simple Operations are also fundamental to REST.

# REST & HTTP

- The motivation for REST was to capture the characteristics of the Web which made the Web successful
  - URI Addressable resources
  - HTTP Protocol
  - Make a Request – Receive Response – Display Response
- Exploits the use of the HTTP protocol beyond HTTP POST and HTTP GET
  - HTTP PUT, HTTP DELETE

# REST

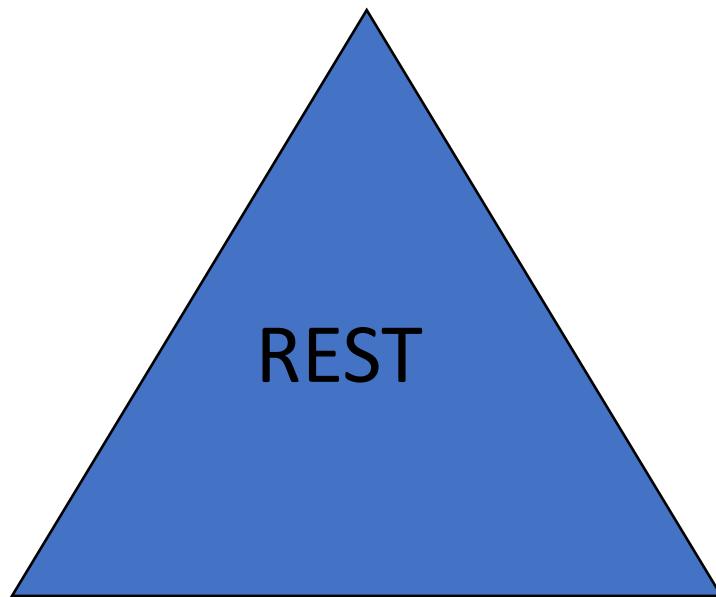
- REST is not a standard
  - is an architectural style
- But it uses several standards:
  - HTTP
  - URL
  - XML/HTML/GIF/JPEG/etc (Resource Representations)
  - text/xml, text/html, image/gif, image/jpeg, etc (Resource Types, MIME Types)

# REST Main Concepts

**Nouns (Resources)**

*unconstrained*

i.e., <http://example.com/employees/12345>



**Verbs**

*constrained*

i.e., GET

**Representations**

*constrained*

i.e., XML

# Resources

- The key abstraction of information in REST is a resource
- A resource is a conceptual mapping to a set of entities
  - Any information that can be named can be a resource: a document or image, a temporal service (e.g., “today's weather in Berlin”), a collection of other resources, a non-virtual object (e.g., a person), etc.
- Represented with a global identifier (URI in HTTP)
  - <http://www.boeing.com/aircraft/747>

# Naming Resources

- REST uses URI to identify resources
  - <http://localhost/books/>
  - <http://localhost/books/ISBN-0011>
  - <http://localhost/books/ISBN-0011/authors>
  - <http://localhost/classes>
  - <http://localhost/classes/cs2650>
  - <http://localhost/classes/cs2650/students>
- As you traverse the path from more generic to more specific, you are navigating the data

# Verbs

- Represent the actions to be performed on resources
- HTTP GET
- HTTP POST
- HTTP PUT
- HTTP DELETE

# HTTP GET

- How clients ask for the information they seek
- Issuing a GET request transfers the data from the server to the client in some representation
- GET <http://localhost/books>
  - Retrieve all books
- GET <http://localhost/books/ISBN-0011021>
  - Retrieve book identified with ISBN-0011021
- GET <http://localhost/books/ISBN-0011021/authors>
  - Retrieve authors for book identified with ISBN-0011021

# HTTP PUT & POST

- HTTP POST creates a resource
- HTTP PUT updates a resource
- POST <http://localhost/books/>
  - Content: {title, authors[], ...}
  - Creates a new book with given properties
- PUT <http://localhost/books/isbn-111>
  - Content: {isbn, title, authors[], ...}
  - Updates book identified by isbn-111 with submitted properties

# Representations

- How data is represented or returned to the client for presentation.
- Two main formats:
  - JavaScript Object Notation (JSON)
  - XML
- It is common to have multiple representations of the same data

# Representations

- XML

```
<COURSE>
  <ID>CS2650</ID>
  <NAME>Distributed Multimedia Software</NAME>
</COURSE>
```

- JSON

```
{course
  {id: CS2650}
  {name: Distributed Multimedia Software}
}
```

# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

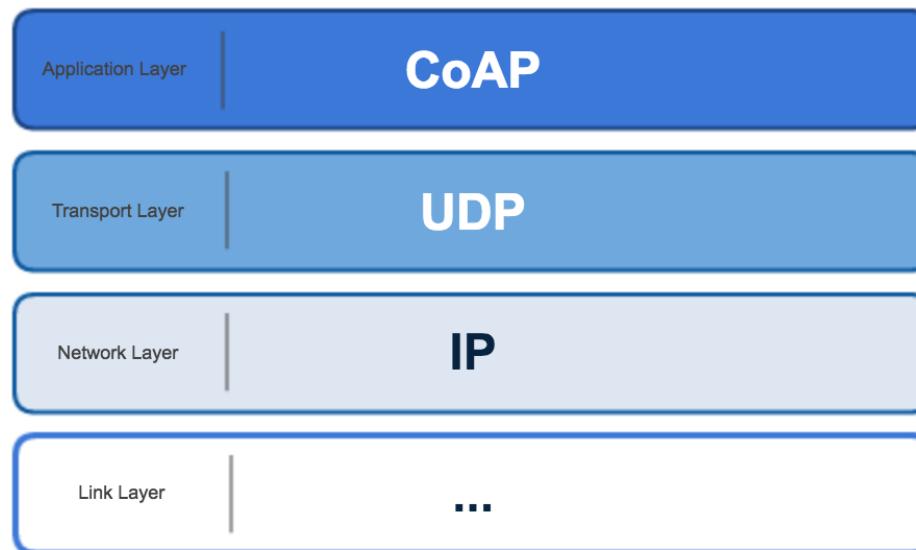
Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# CoAP

- Constrained Application Protocol
  - REST-based web transfer protocol
  - manipulates Web resources using the same methods as HTTP: GET, PUT, POST, and DELETE
  - subset of HTTP functionality re-designed for low power embedded devices such as sensors (for IoT and M2M)

# CoAP

- TCP overhead is too high and its flow control is not appropriate for short-lived transactions
- UDP has lower overhead and supports multicast



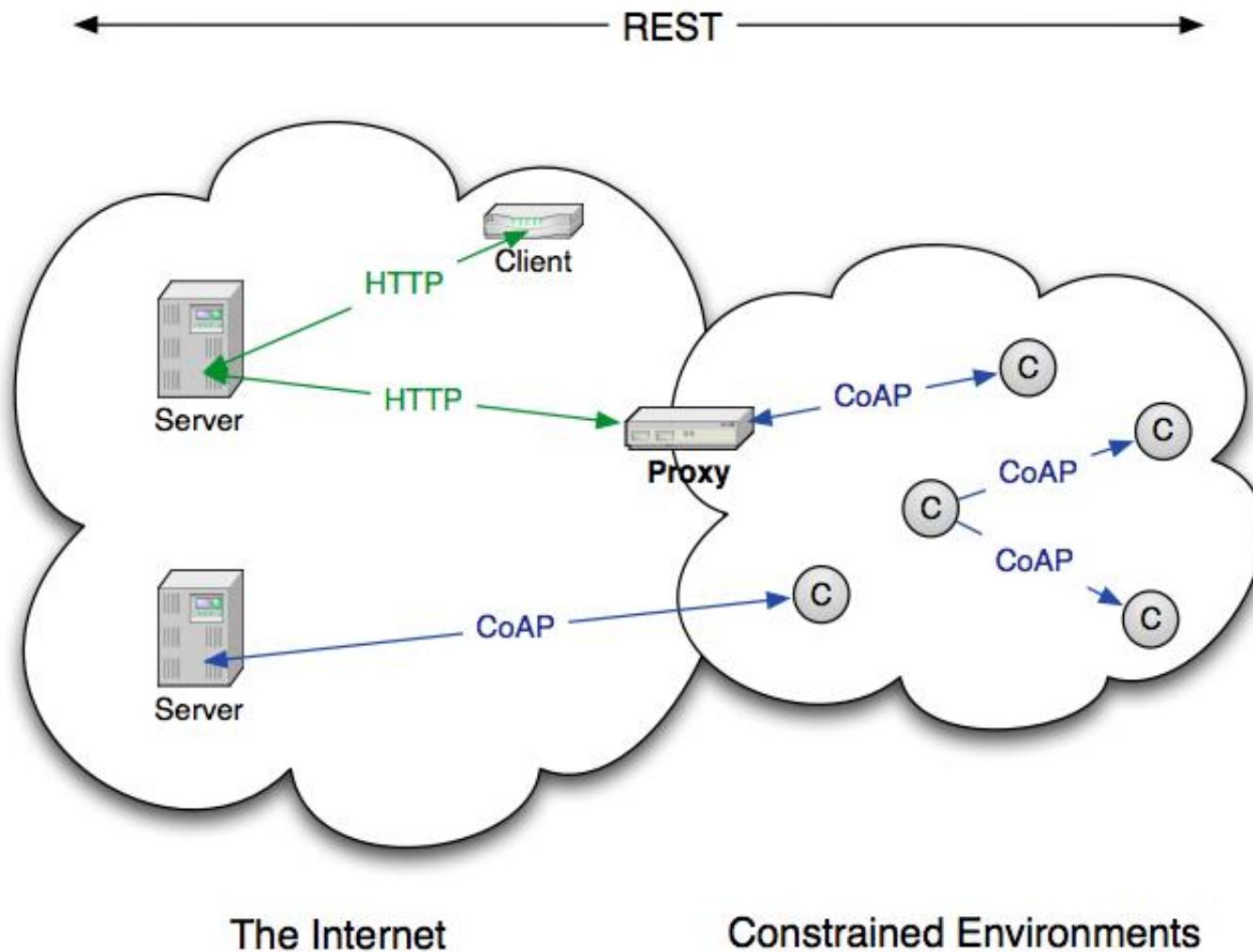
# CoAP

- Four message types:
  - **Confirmable** – requires an ACK
  - **Non-confirmable** – no ACK needed
  - **Acknowledgement** – ACKs a Confirmable
  - **Reset** - indicates a Confirmable message has been received but context is missing for processing

# CoAP

- CoAP provides reliability without using TCP as transport protocol
- CoAP enables asynchronous communication
  - e.g., when CoAP server receives a request which it cannot handle immediately, it first ACKs the reception of the message and sends back the response in an off-line fashion
- Also supports multicast and congestion control

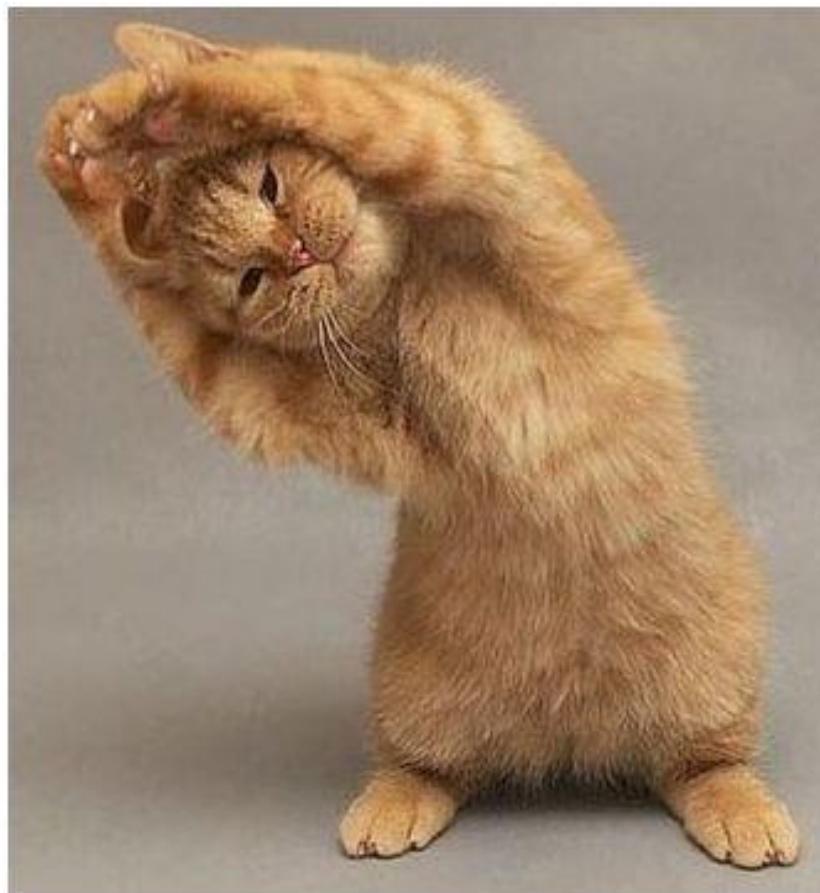
# CoAP



# What CoAP Is

- CoAP is
  - A RESTful protocol
  - Both synchronous and asynchronous
  - For constrained devices and networks
  - Specialized for M2M applications
  - Easy to proxy to/from HTTP

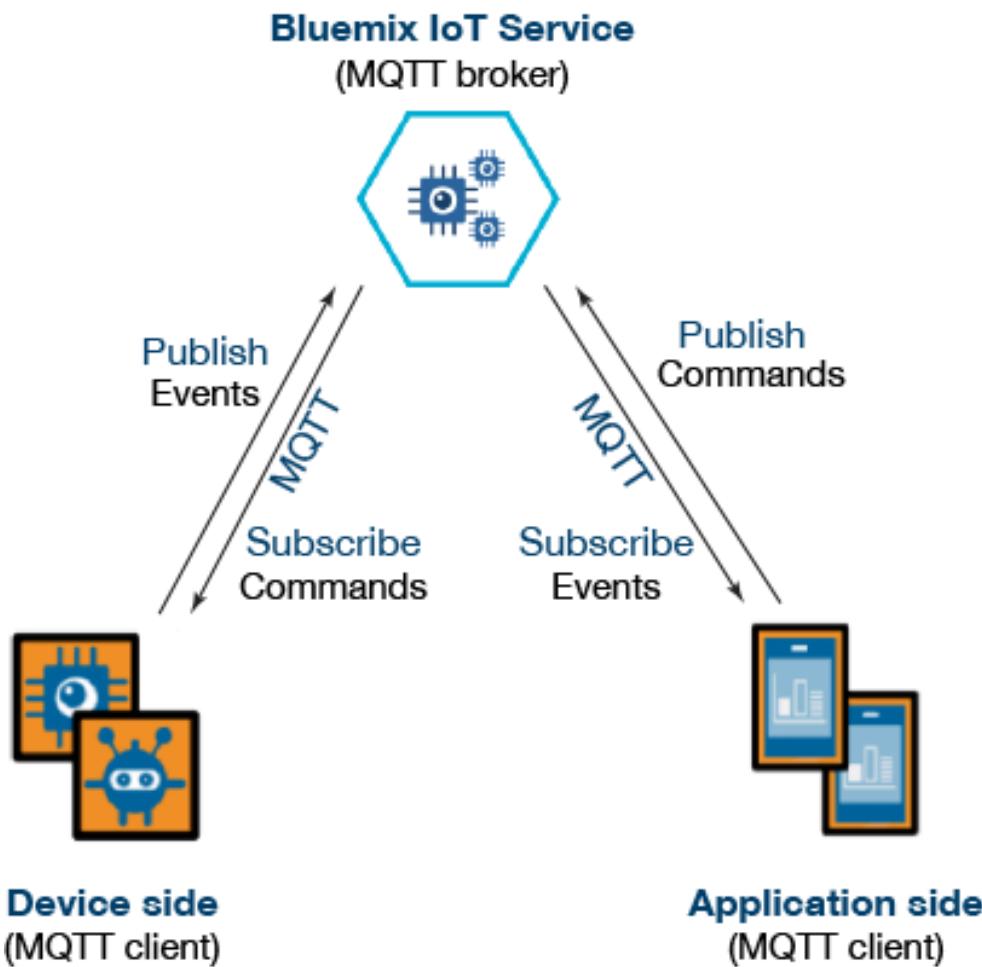
# BREAK



# MQTT

- Message Queuing Telemetry Transport
- In a nutshell, MQTT consists of three parts:
  - Broker
  - Subscribers
  - Publishers

# MQTT



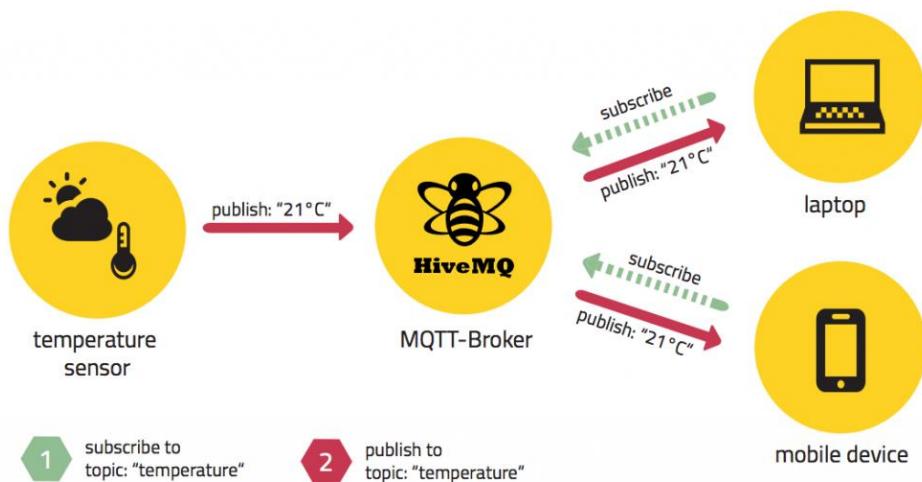
# MQTT

- MQTT was invented by Andy Stanford-Clark (IBM) and Arlen Nipper (Arcom, now Cirrus Link) back in 1999, where their use case was to create a protocol for minimal battery loss and minimal bandwidth connecting oil pipelines over satellite connections. They specified the following goals, which the future protocol should have:
  - Simple to implement
  - Provide a Quality of Service Data Delivery
  - Lightweight and Bandwidth Efficient
  - Data Agnostic
  - Continuous Session Awareness

# MQTT

- Built for proprietary embedded systems; now shifting to IoT
- You can send anything as a message; up to 256 MB
- Built for unreliable networks
- Enterprise scale implementations down to hobby projects
- Decouples readers and writers
- Message have a topic, quality of service, and retain status associated with them

# Publish/Subscribe Concept



## Decoupled in space and time:

The clients do not need each others IP address and port (space) and they do not need to be running at the same time (time).

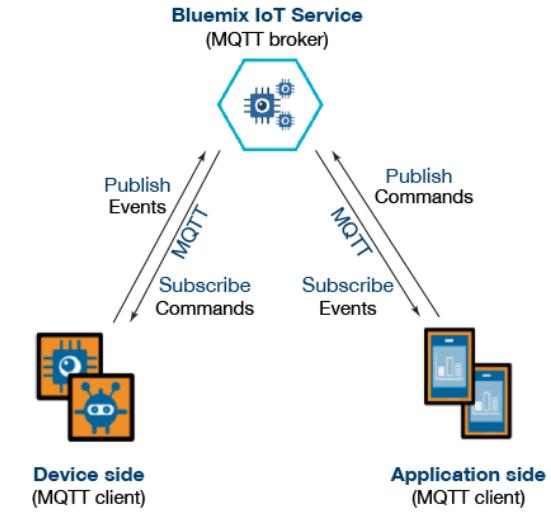
The broker's IP and port must be known by clients

Namespace hierarchy used for topic filtering

It may be the case that a published message is never consumed by any subscriber

# MQTT: Example

- Clients connect to a “Broker”
- Clients subscribe to topics e.g.,
  - `client.subscribe('toggleLight/1')`
  - `client.subscribe('toggleLight/2')`
  - `client.subscribe('toggleLight/3')`
- Clients can publish messages to topics:
  - `client.publish('toggleLight/1', 'toggle');`
  - `client.publish('toggleLight/2', 'toggle');`
- All clients receive all messages published to topics they subscribe to
- **Messages can be anything**
  - Text
  - Images
  - etc.

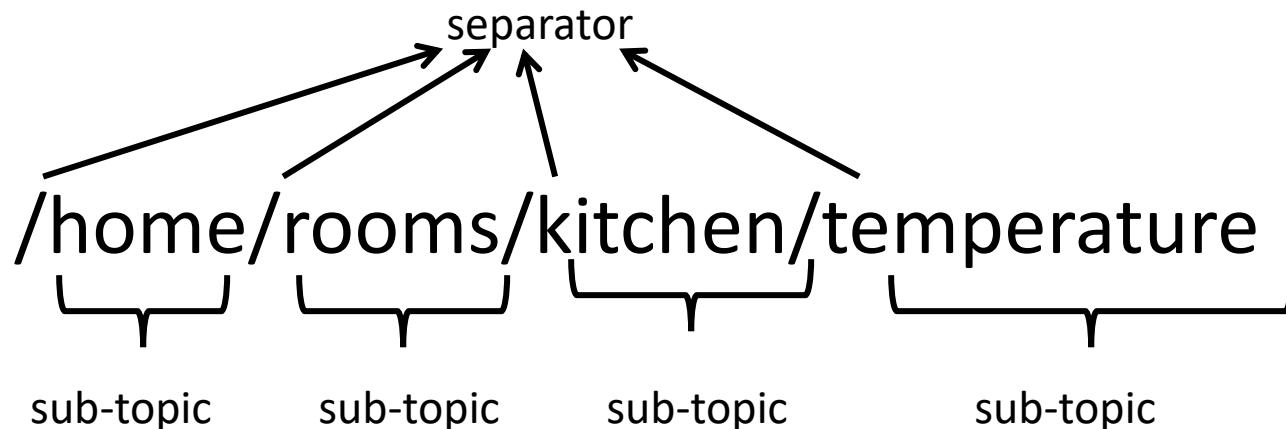


# Node.js Example

```
var mqtt = require('mqtt');
var client = mqtt.createClient('<><PortNumber>>', 'm11.cloudmqtt.com', {
    username: '<><UserName>>',
    password: '<><Password>>'
});
client.on('connect', function () { // When connected
    // subscribe to a topic
    client.subscribe('TEMPERATURE_READING', function () {
        // when a message arrives, do something with it
        client.on('message', function (topic, message, packet) {
            console.log("Received '" + message + "' on '" + topic + "'");
        });
    });
    // publish a message to a topic
    client.publish('SET_TEMPERATURE', '24', function () {
        console.log("Message is published");
    });
});
```

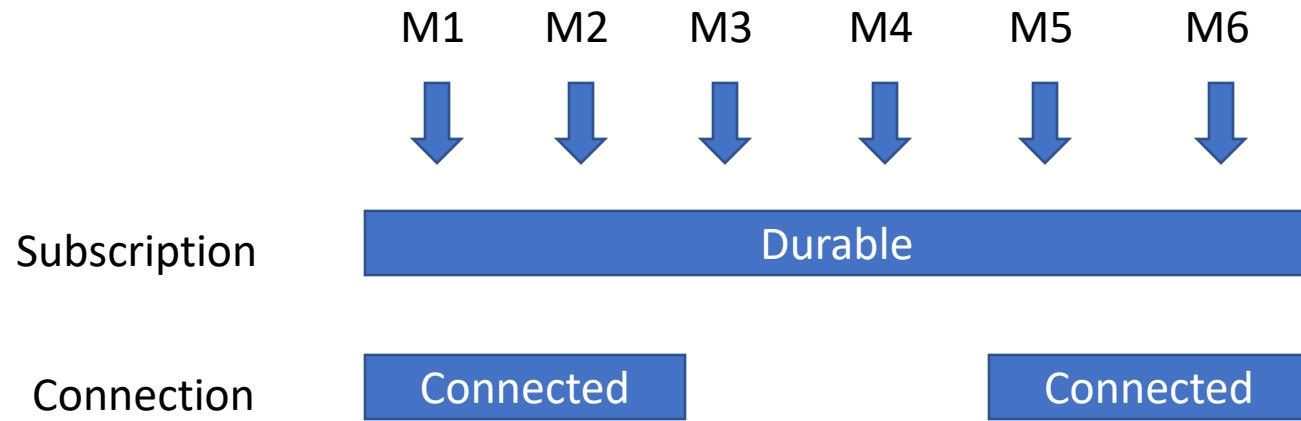
# Topics

- Each published data specifies a topic
- Each subscriber subscribed to that topic will receive it
- Topic format:



# Durable/Transient Subscriptions

- Subscriptions
  - Durable
    - If the subscriber disconnect messages are buffered at the broker and delivered upon reconnection
  - Non-durable
    - Connection lifetime gives subscription lifetime



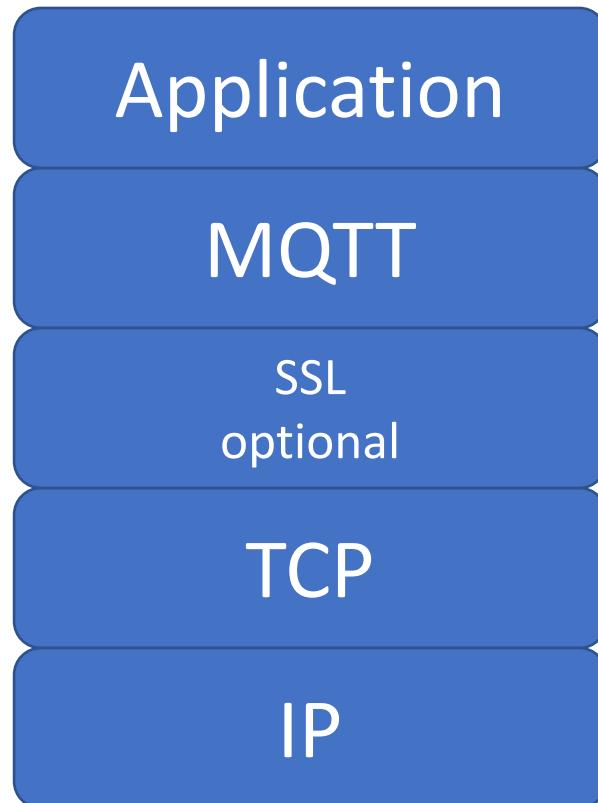
# State Retention

- Publications
  - Retained (“persistent” message)
    - The subscriber upon first connection receives the last good publication (i.e., does not have to wait for new publication)
  - One flag set both in the publish packet to the broker and in the published packet to the subscribers
    - Only the most recent persistent message is stored and distributed

# Session Aware

- Last Will and Testament (LWT) – topic published upon disconnecting a connection
- Any client can register a LWT
- Anybody subscribing to the LWT topic will know when a certain device (that registered a LWT) disconnected

# Protocol Stack



TCP/IP Port: 1883

When running over SSL, TCP/IP port 8883

SSL: Secure Socket Layer (encryption)

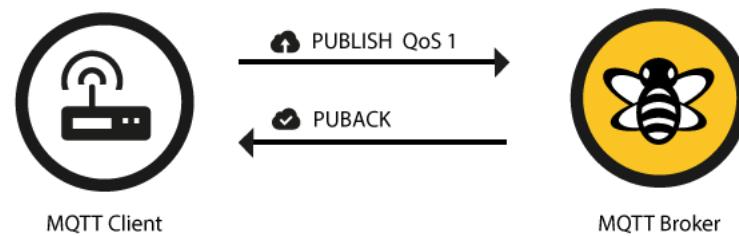
# Publishing “QoS” (Reliability)

- 0 – unreliable (aka “at most once”)
  - OK for continuous streams, least overhead (1 message)
  - “Fire and forget”
  - TCP will still provide reliability



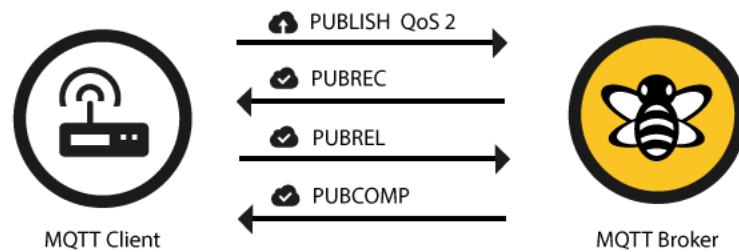
# Publishing “QoS” (Reliability)

- 1 – delivery “at least once” (duplicates possible)
  - Used for alarms – more overhead (2 messages)
  - Contains message ID (to match with ACKed message)



# Publishing “QoS” (Reliability)

- 2 – delivery “exactly once”
  - Utmost reliability is important – most overhead (4 messages) and slowest

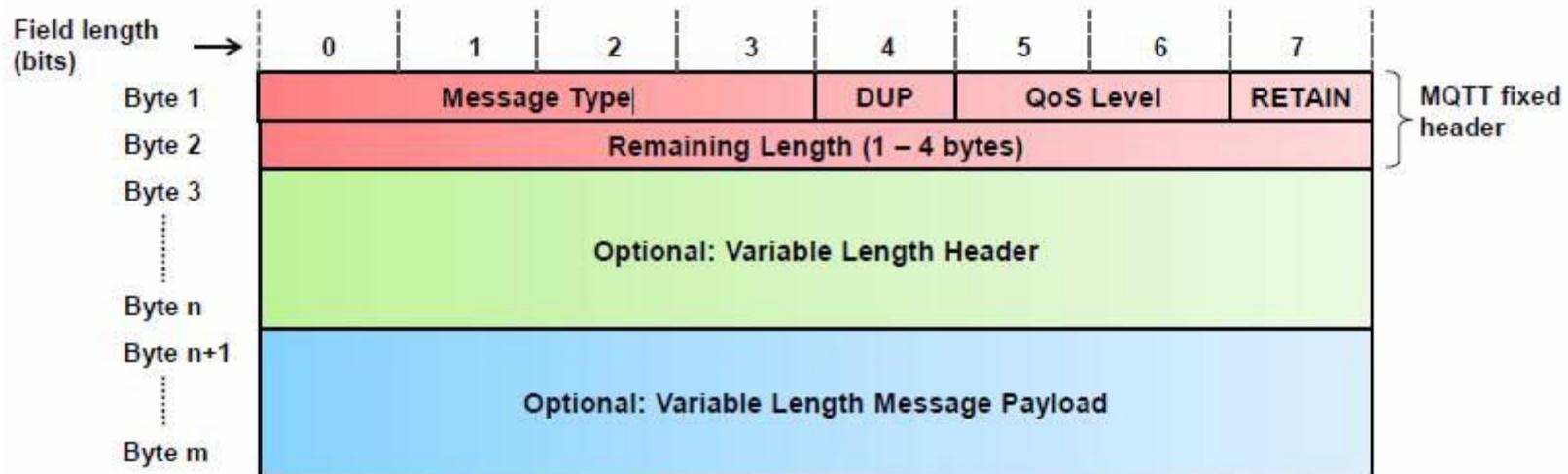


# Publishing “QoS” (Reliability)

- Reliability maintained even if the TCP connection breaks (intermittent connections)
- Separate QoS for publishing and for subscribing

# MQTT Message Format

Shortest Message is Two Bytes



# Message Types

Name	Value	Direction of flow	Description
Reserved	0	Forbidden	Reserved
CONNECT	1	Client to Server	Client request to connect to Server
CONNACK	2	Server to Client	Connect acknowledgment
PUBLISH	3	Client to Server or Server to Client	Publish message
		Server to Client	
PUBACK	4	Client to Server or Server to Client	Publish acknowledgment
PUBREC	5	Client to Server or Server to Client	Publish received (assured delivery part 1)
PUBREL	6	Client to Server or Server to Client	Publish release (assured delivery part 2)
PUBCOMP	7	Client to Server or Server to Client	Publish complete (assured delivery part 3)
SUBSCRIBE	8	Client to Server	Client subscribe request
SUBACK	9	Server to Client	Subscribe acknowledgment
UNSUBSCRIBE	10	Client to Server	Unsubscribe request
UNSUBACK	11	Server to Client	Unsubscribe acknowledgment
PINGREQ	12	Client to Server	PING request
PINGRESP	13	Server to Client	PING response
DISCONNECT	14	Client to Server	Client is disconnecting
Reserved	15	Forbidden	Reserved

# Message Types

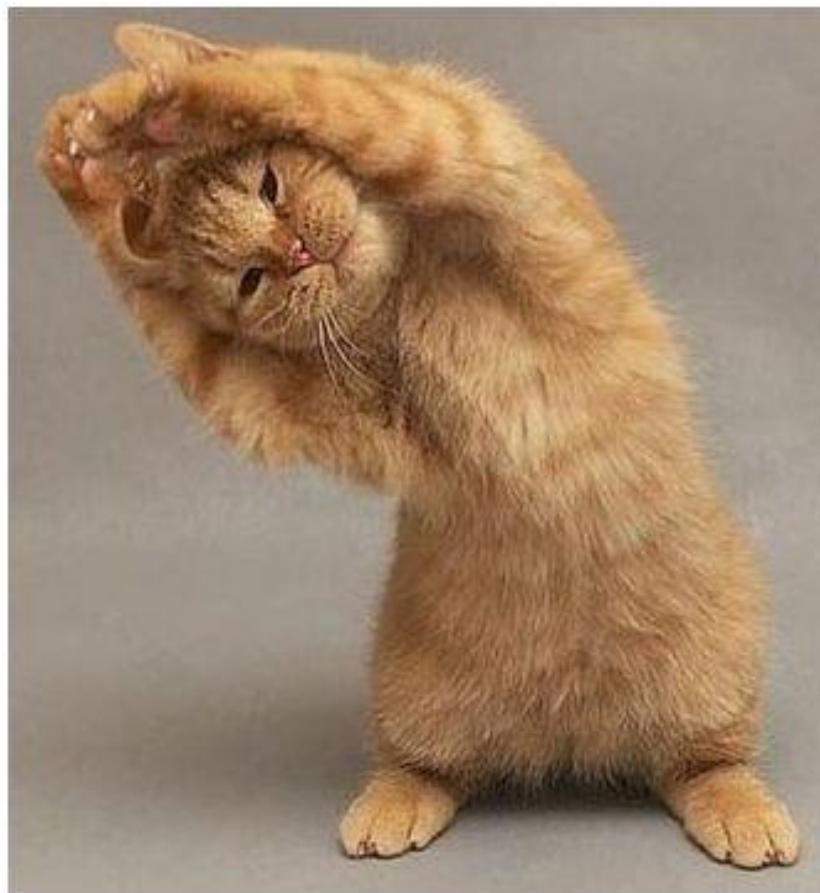
Message fixed header field	Description / Values	
<b>Message Type</b>	0: Reserved	8: SUBSCRIBE
	1: CONNECT	9: SUBACK
	2: CONNACK	10: UNSUBSCRIBE
	3: PUBLISH	11: UNSUBACK
	4: PUBACK	12: PINGREQ
	5: PUBREC	13: PINGRESP
	6: PUBREL	14: DISCONNECT
	7: PUBCOMP	15: Reserved
DUP	<p>Duplicate message flag. Indicates to the receiver that this message may have already been received.</p> <p>1: Client or server (broker) re-delivers a PUBLISH, PUBREL, SUBSCRIBE or UNSUBSCRIBE message (duplicate message).</p>	
QoS Level	<p>Indicates the level of delivery assurance of a PUBLISH message.</p> <p>0: At-most-once delivery, no guarantees, «Fire and Forget».</p> <p>1: At-least-once delivery, acknowledged delivery.</p> <p>2: Exactly-once delivery.</p> <p>Further details see <a href="#">MQTT QoS</a>.</p>	
RETAIN	<p>1: Instructs the server to retain the last received PUBLISH message and deliver it as a first message to new subscriptions.</p> <p>Further details see <a href="#">RETAIN (keep last message)</a>.</p>	
Remaining Length	<p>Indicates the number of remaining bytes in the message, i.e. the length of the (optional) variable length header and (optional) payload.</p> <p>Further details see <a href="#">Remaining length (RL)</a>.</p>	

# Comparison CoAP & MQTT

Both used in IoT

- CoAP:
  - one-to-one communication
  - UDP/IP
  - unreliable
  - lightweight and easy to implement
- MQTT:
  - many-to-many communication
  - TCP/IP
  - focus on message delivery; reliable
  - higher overheads (protocol data, processing costs)

# BREAK



# **RFID: Technology and Applications**

Internet-of-Things (IoT) COCSC20

## Effect on Manufacturing

- Need to ensure error-free, custom assembly
- Need inventory of components for the various customization options
- Critical Issues
  - Assembly process control
  - Inventory management
  - Supply chain integration
  - Customer insight
- One solution: RFID



# What is RFID?



Who Are You?

I am Product X



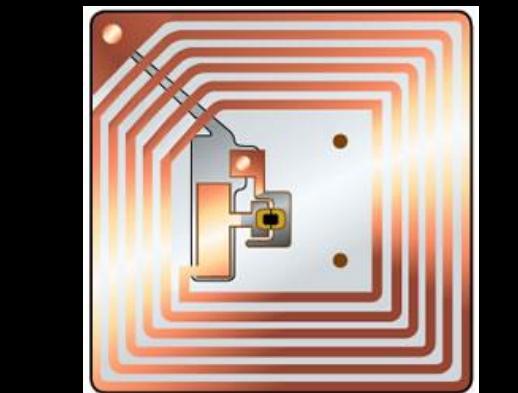
# What is RFID?

- RFID = Radio Frequency IDentification
- An ADC (Automated Data Collection) technology that:
  - Uses radio-frequency waves to transfer data between a reader and a movable item to identify, categorize, track
  - Is fast and does not require physical sight or contact between reader/scanner and the tagged item
  - Performs the operation using low cost components
  - Attempts to provide unique identification and backend integration that allows for wide range of applications
- Other ADC technologies: Bar codes, OCR

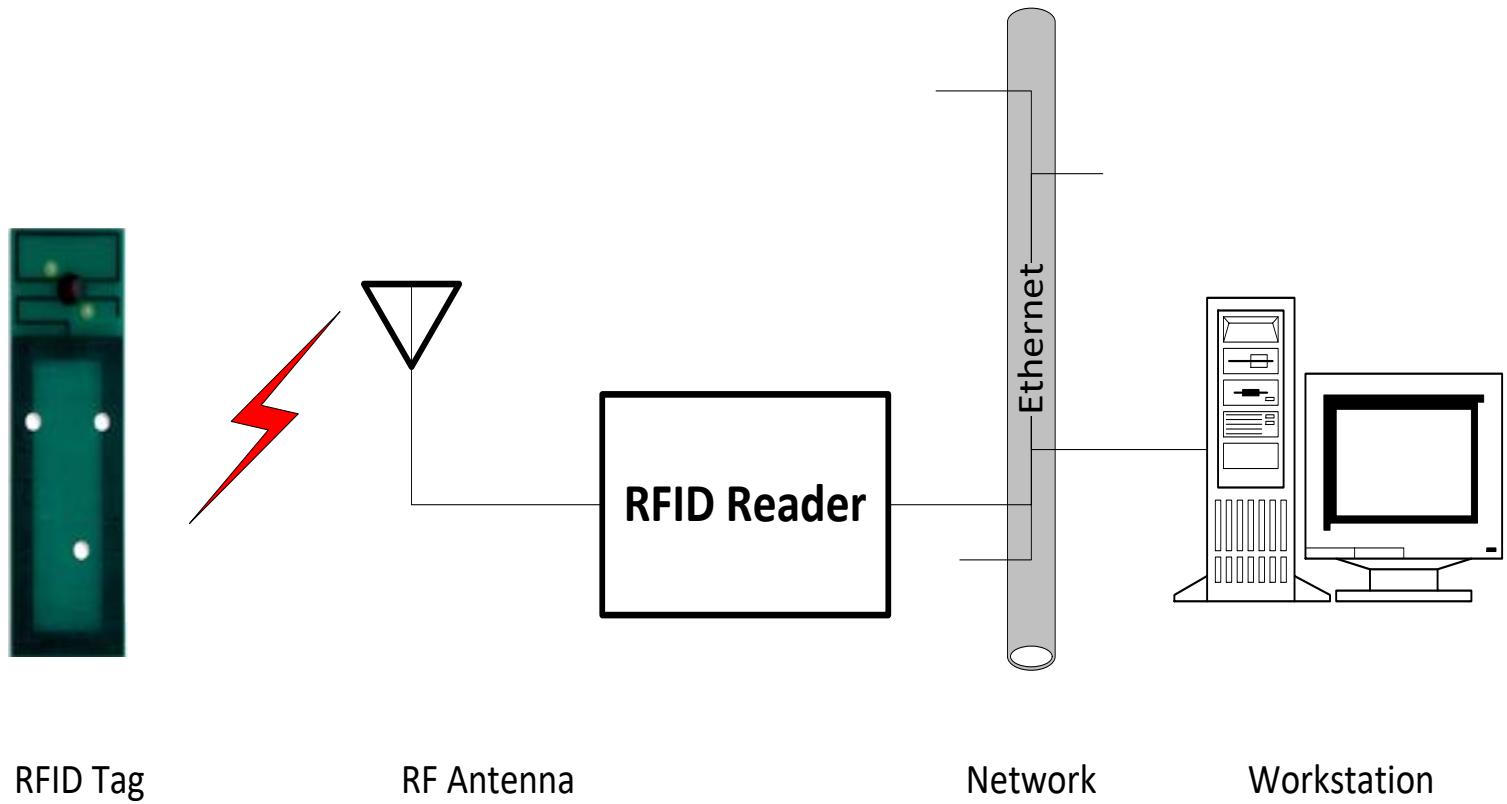
# RFID Systems

Main components:

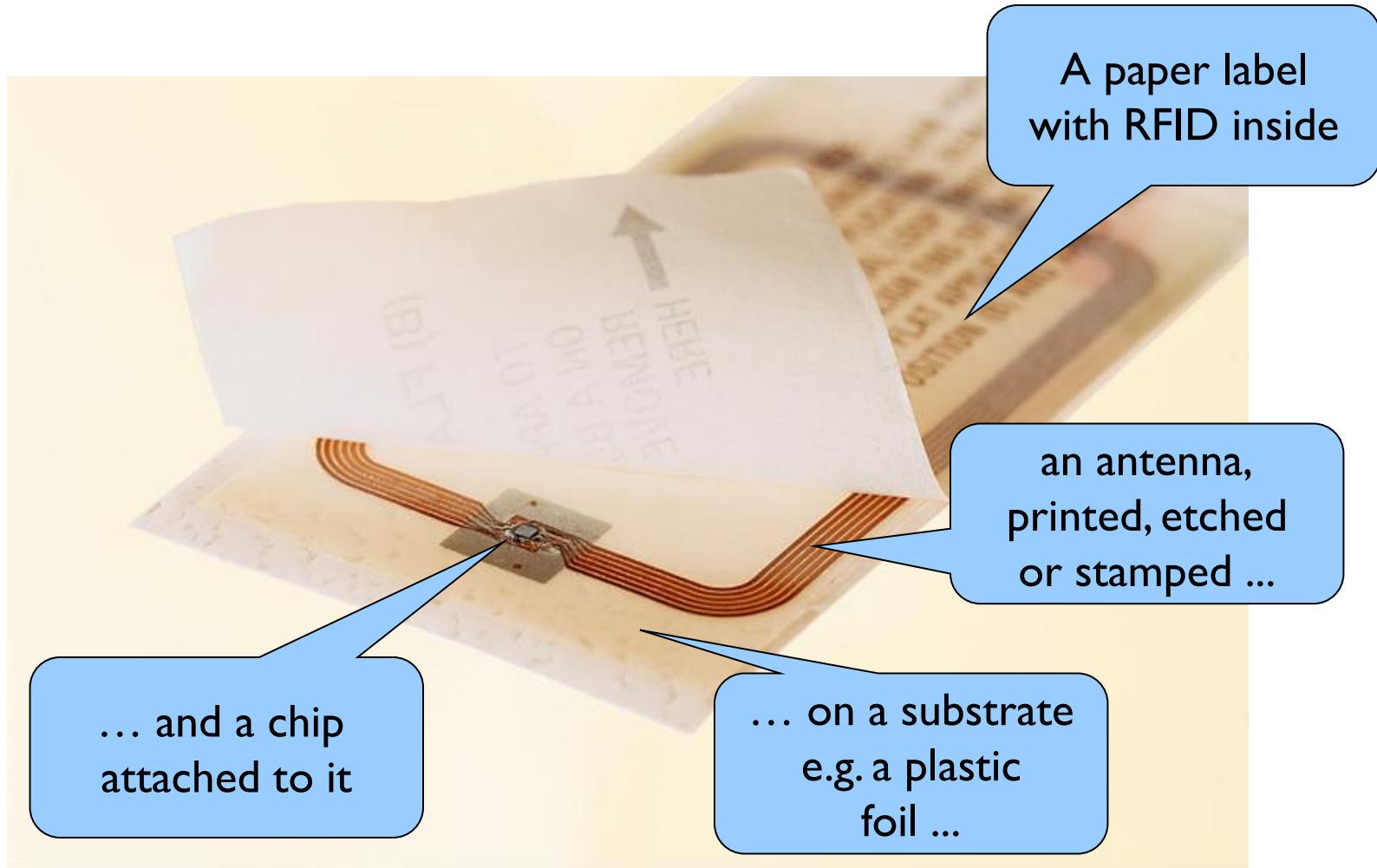
- **Tags (transponders)**
  - Microchip & antenna
- **Tag reader**
  - Decoder & antenna
  - RFID reader sends pulse of energy and waits for response
  - Can be on all the time or activate only in response to external event



# RFID System Connectivity



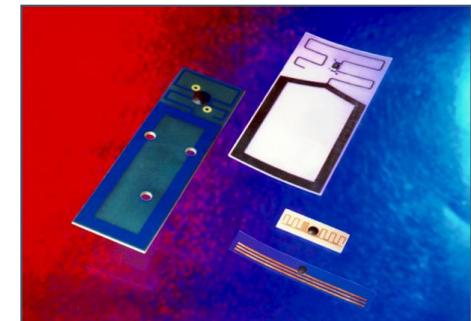
# RFID Tags: Smart Labels



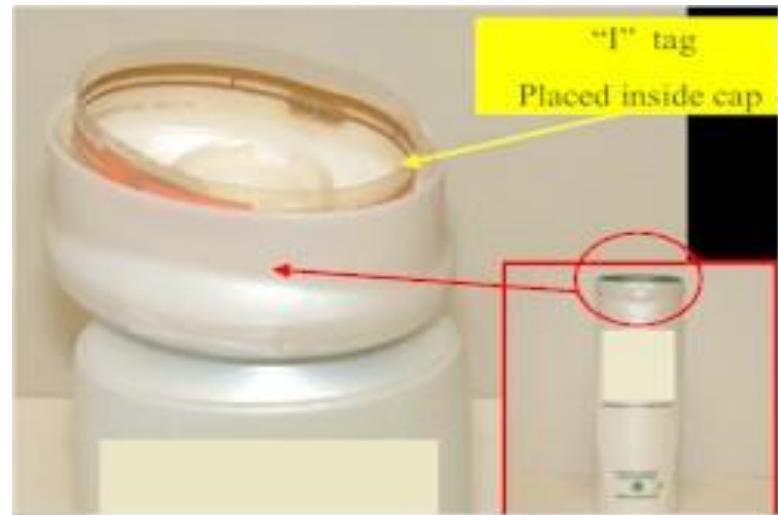
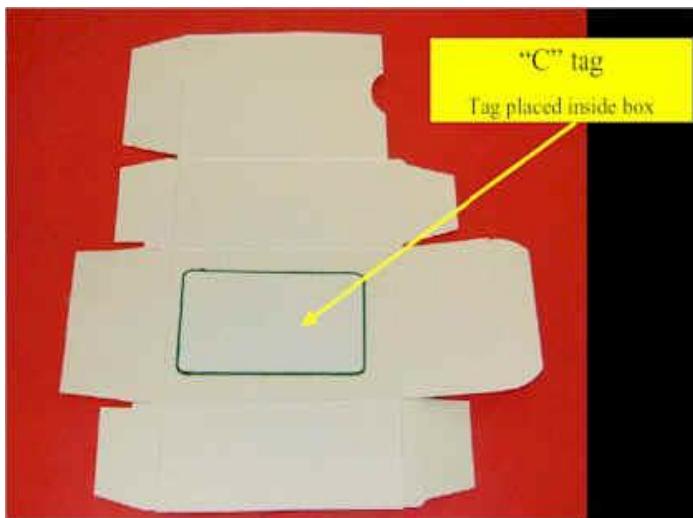
# Tags

Variations:

- **Memory**
  - Size (16 bits - 512 Kbytes)
  - Read-Only, Read/Write, or WORM
- **Arbitration (Anti-collision)**
  - Ability to read/write one or many tags at a time
- **Frequency**
  - 125KHz - 5.8 GHz
- **Price**
  - \$0.10 to \$250
- **Physical Dimensions**
  - Thumbnail to Brick sizes



# Some RFID Tags

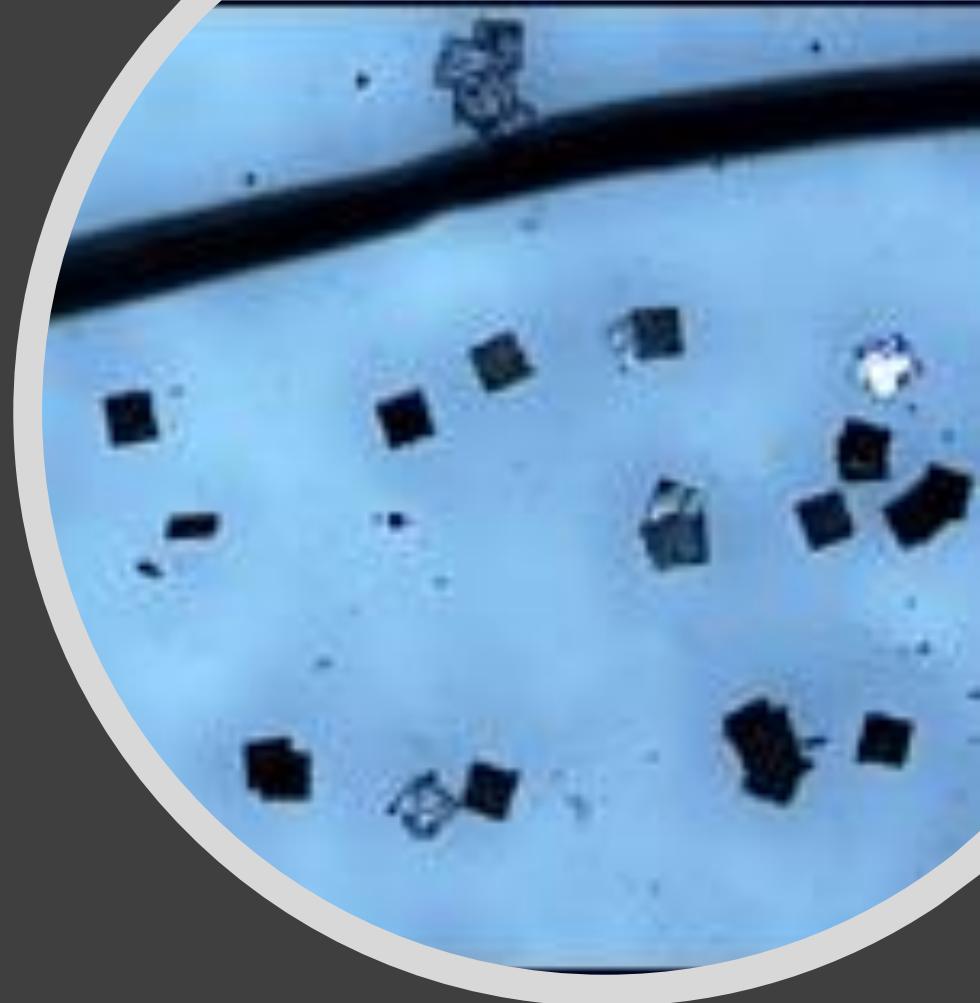




“Mission Impossible” |

## Tiny Tags

- 2007 Hitachi produced RFID device measuring  $0.05 \times 0.05$  mm, and thin enough to be embedded in a sheet of paper. The data contained on them can be extracted from as far away as a few hundred meters.

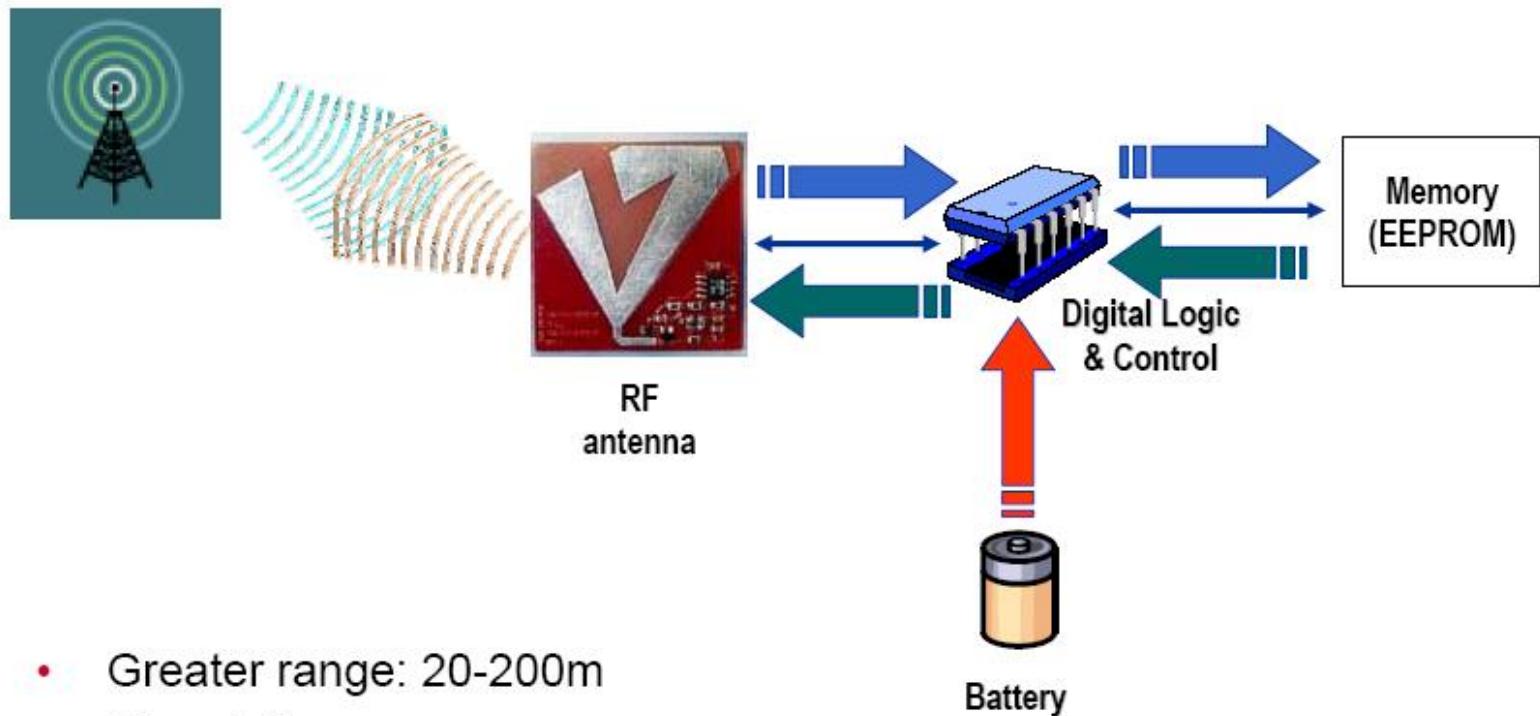


# Active v/s Passive

- Tags can be attached to almost anything:
  - Items, cases or pallets of products, high value goods
  - Vehicles, assets, livestock or personnel

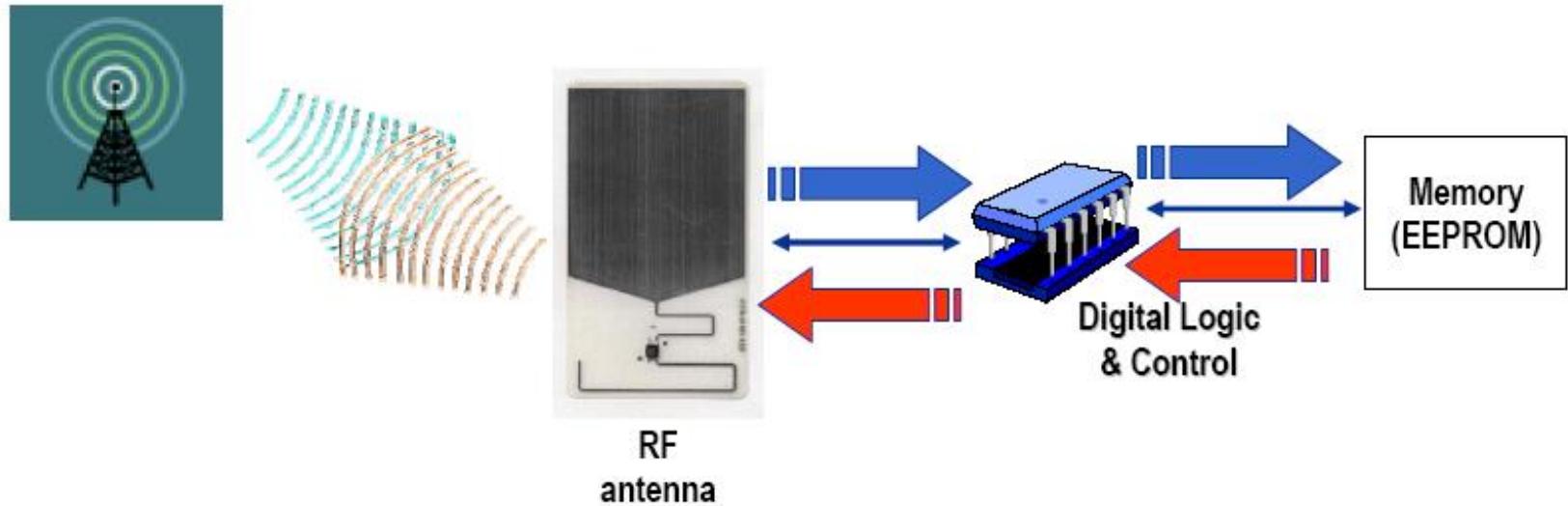
	<b>Active RFID</b>	<b>Passive RFID</b>
<b>Tag Power Source</b>	<b>Internal to tag</b>	<b>Energy transferred using RF from reader</b>
Tag Battery	Yes	No
Required signal strength	Very Low	Very High
Range	Up to 100m	Up to 3-5m, usually less
Multi-tag reading	1000's of tags recognized – up to 100mph	Few hundred within 3m of reader, about 3 sec per read => at most 3 mph.
Data Storage	Up to 512 KB	16 bits – 1 KB
Cost	3000 to 10000 INR	Upto 1000 INR
Tag Memory	Typically can be re-written by RF Interrogators	Usually Write-Once-Read-Many or Read-Only tags

# Active Tag



- Greater range: 20-200m
- 10 yr. Life
- Limited sensor capabilities
- “Self-powered” uses interrogator RF beam for wake-up and communication

# Passive Tag



- Limited range: <10m (frequency dependent)
- Communication & power from interrogator RF beam

# RFID Tag Memory

## Read-only tags

- Tag ID is assigned at the factory during manufacturing
  - Can never be changed
  - No additional data can be assigned to the tag

## Write once, read many (WORM) tags

- Data written once, e.g., during packing or manufacturing
  - Tag is locked once data is written
  - Similar to a compact disc or DVD

## Read/Write

- Tag data can be changed over time
  - Part or all of the data section can be locked

# RFID Readers

- Reader functions:
  - Remotely power tags
  - Establish a bidirectional data link
  - Inventory tags, filter results
  - Communicate with networked server(s)
  - Can read 100-300 tags per second
- Readers (interrogators) can be at a fixed point such as
  - Entrance/exit
  - Point of sale
- Readers can also be mobile/hand-held.



# Some RFID Readers



# RFID Advantages over Bar-Codes

- No line of sight required for reading
- Multiple items can be read with a single scan
- Each tag can carry a lot of data (read/write)
- Individual items identified and not just the category
- Passive tags have a virtually unlimited lifetime
- Active tags can be read from great distances
- Can be combined with barcode technology

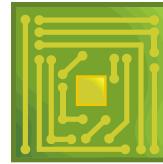
# “Smart labels”: EPC (Electronic Product Code) tags



Line-of-sight

Specifies object type

EPC tag



Radio contact

Uniquely specifies object

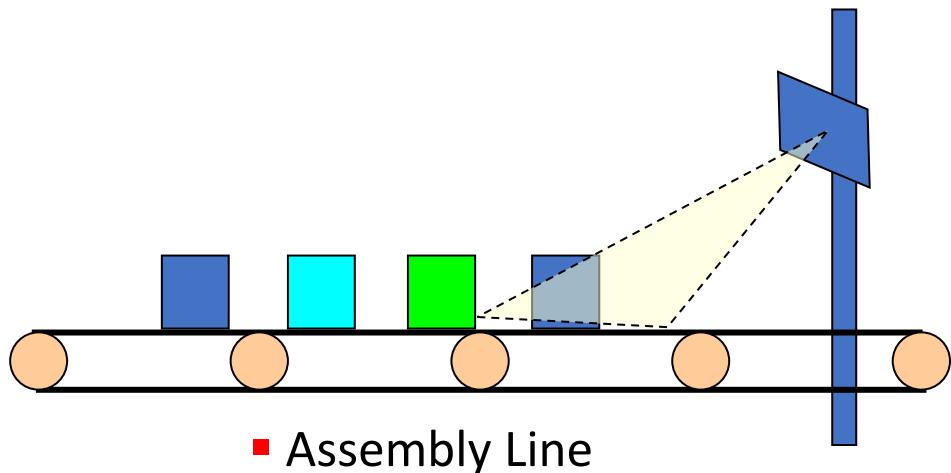
*Fast, automated  
scanning*

*Provides pointer  
to database entry  
for every object,  
i.e., unique,  
detailed history*

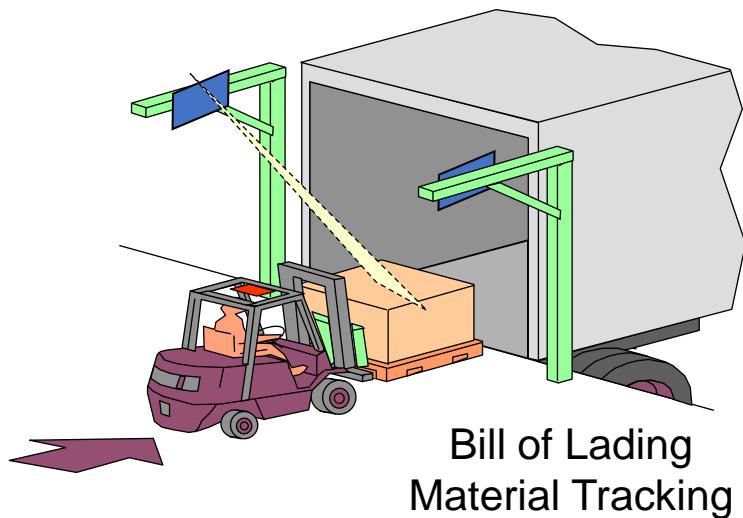
# RFID Vs Barcode

Barcode	RFID
Optical technology	Radio technology
Barcode uses UPC (Universal Product Code)	RFID uses EPC (electronic product code)
Bar codes are larger than the smallest tag	Tags range in size from a postage stamp to a book
Barcodes have unlimited shelf life but are subject to degradation with handling.	Tags have no moving parts and have multi-year lifespan
Bar Codes can be easily duplicated and reattached to products and are, therefore, easily counterfeited	Tags have a unique identity code embedded on the microchip. Its difficult to duplicate tags so products can not be counterfeited

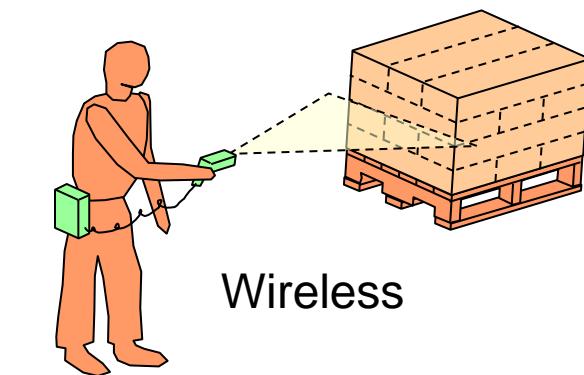
# RFID Application Points



■ Assembly Line



Bill of Lading  
Material Tracking



Wireless

■ Handheld Applications

■ Shipping Portals

# RFID Applications

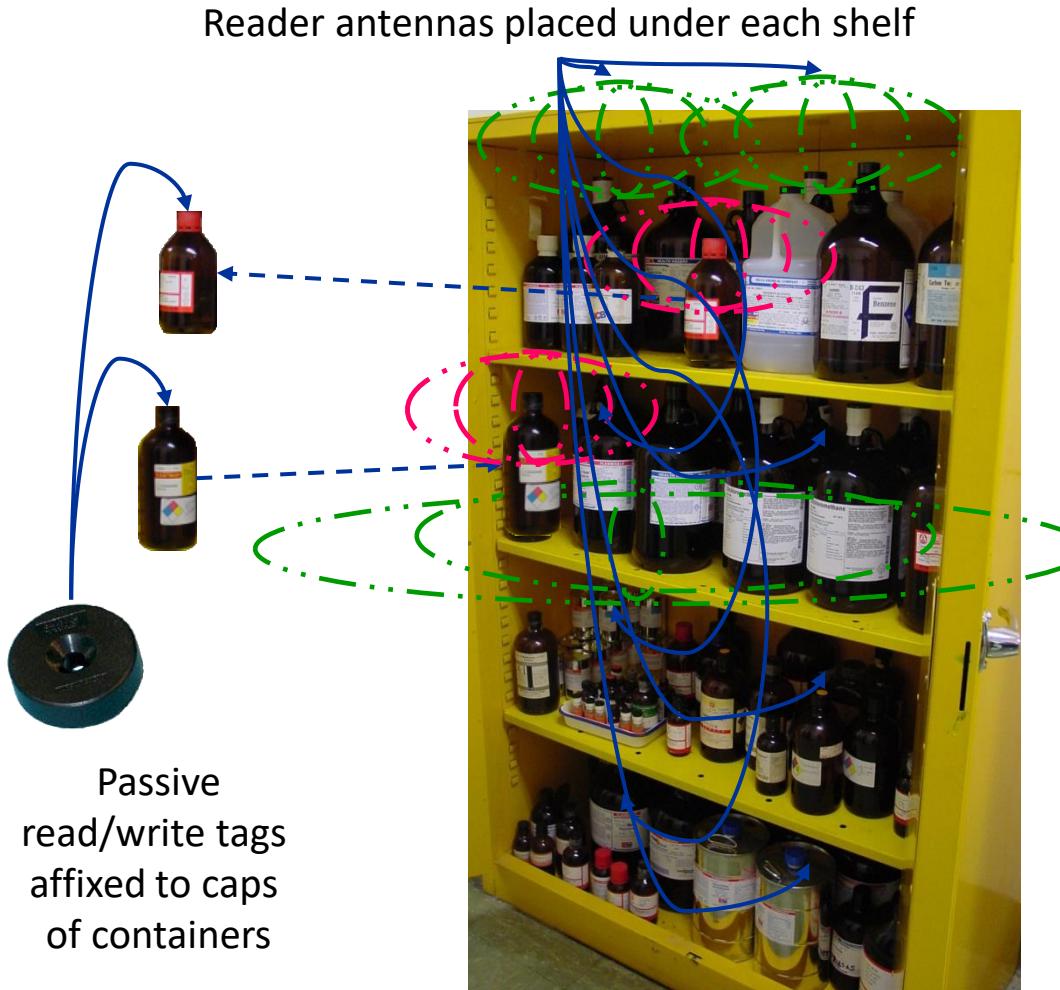
- Manufacturing and Processing
  - Inventory and production process monitoring
  - Warehouse order fulfillment
- Supply Chain Management
  - Inventory tracking systems
  - Logistics management
- Retail
  - Inventory control and customer insight
  - Auto checkout with reverse logistics
- Security
  - Access control
  - Counterfeiting and Theft control/prevention
- Location Tracking
  - Traffic movement control and parking management
  - Wildlife/Livestock monitoring and tracking

# Smart Groceries

- Add an RFID tag to all items in the grocery
- As the cart leaves the store, it passes through an RFID transceiver
- The cart is rung up in seconds



# Smart Cabinet



1. Tagged item is removed from or placed in "Smart Cabinet"
2. "Smart Cabinet" periodically interrogates to assess inventory
3. Server/Database is updated to reflect item's disposition
4. Designated individuals are notified regarding items that need attention (cabinet and shelf location, action required)

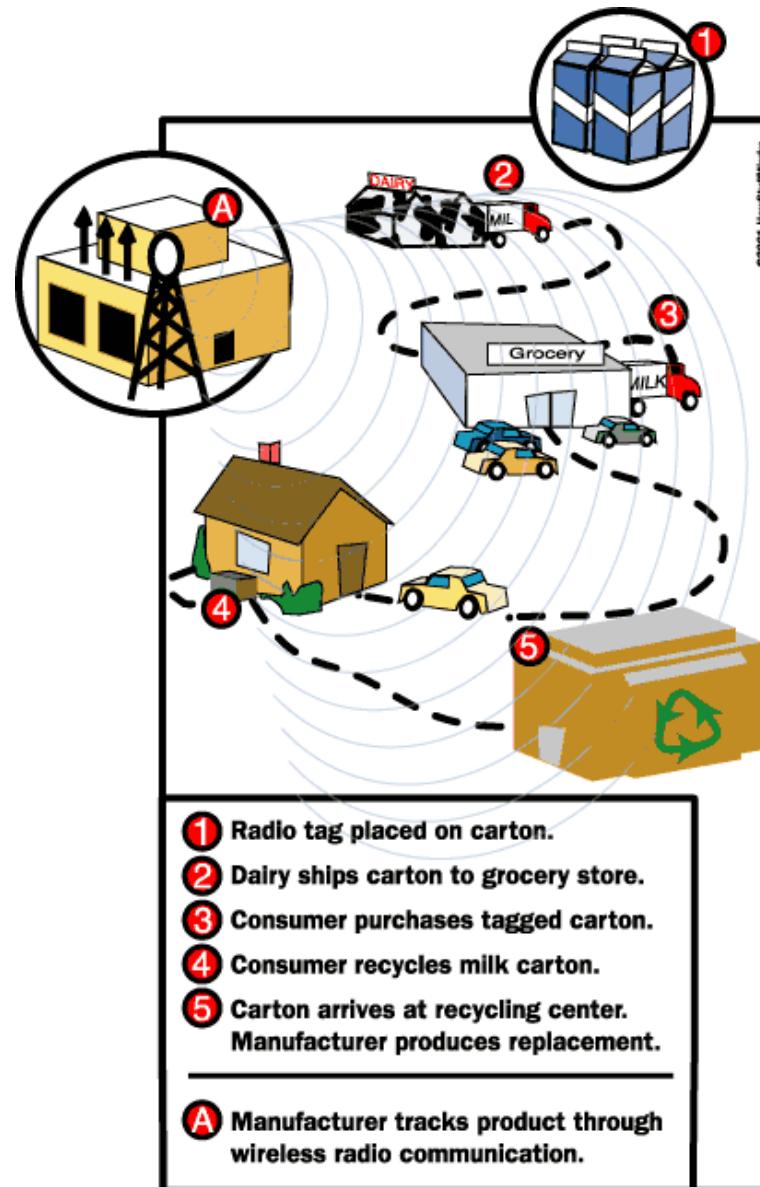
# Smart Fridge

- Recognizes what's been put in it
- Recognizes when things are removed
- Creates automatic shopping lists
- Notifies you when things are past their expiration
- Shows you the recipes that most closely match what is available



# Smart Groceries Enhanced

- Track products through their entire lifetime

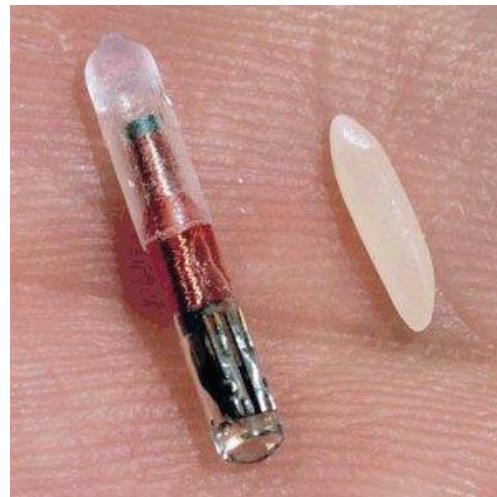
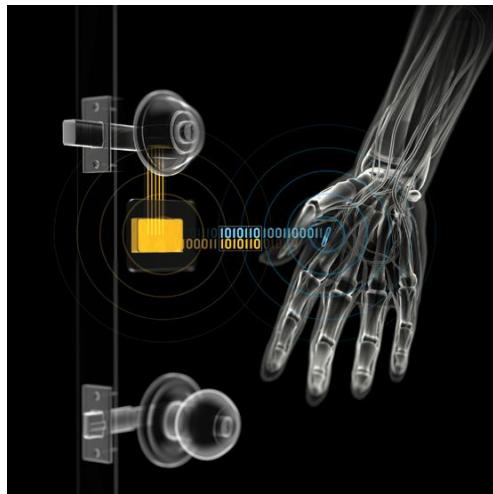


# Some More Smart Applications

- “Smart” appliances:
  - Closets that advice on style depending on clothes available
  - Ovens that know recipes to cook pre-packaged food
- “Smart” products:
  - Clothing, appliances, CDs, etc. tagged for store returns
- “Smart” paper:
  - Airline tickets that indicate your location in the airport
- “Smart” currency:
  - Anti-counterfeiting and tracking
- “Smart” people ??

# Implants

- It is the most controversial application
- Small glass cylinders approximately 2 or 3mm wide and between 1 and 1.5cm long
- Consists of a microchip, a coiled antenna, and a capacitor
- Implanted typically under the skin of arm or the back of the neck



<https://www.youtube.com/watch?v=HkKhILzoGR8>

# Security/Privacy Issues and Solutions

- **Unauthorized Reading:**

- Scan closed boxes and find out what is inside
- Read RFID enabled credit card or ID (metal foil in passports)

- **Unauthorized Writing:**

- Can change UPC/price of an item
- Can kill a tag

- **RFID Zapper:**

- Can burn a tag using overcurrent

- **RSA Blocker Tag:**

- Placed near another RFID; prevents its reading

- **Put Tag to Sleep:**

- Can wake up later; reuse tags

- **Re-label Tag and Dual-Use Tag:**

- Customer sees differed info or can over-write tag with useful information

- **Authentication:**

- Reader has to know PIN

# Near-Field Communication (NFC)

- NFC is one of the latest wireless communication technologies. As a short-range wireless connectivity technology, NFC offers safe yet simple communication between electronic devices
- It enables exchange of data between devices over a distance of 4 cm or less
- NFC operates at 13.56 MHz and rates ranging from 106 kbit/s to 848 kbit/s

# How NFC Works

- NFC is based on **RFID technology** that uses magnetic field induction between electronic devices in close proximity
- For two devices to communicate using NFC, one device must have an **NFC reader/writer** and one must have an **NFC tag**. The tag is essentially an integrated circuit containing data, connected to an antenna, that can be read or written by the reader

# How NFC Works

- The technology is a simple extension of the ISO/IEC14443 proximity-card standard (contactless card, RFID) that **combines the interface of a smartcard and a reader into a single device**
- An NFC device can **communicate with both existing ISO/IEC14443 smartcards and readers, as well as with other NFC devices**, and is thereby compatible with contactless infrastructure already in use for public transportation and payment
- NFC is primarily aimed at usage in **mobile phones**
- 2015: ~600 million NFC-equipped phones in use (estimate that 5% are used at least once a month)

# NFC Applications

There are currently three main uses of NFC:

- **Card emulation:** The NFC device behaves like an existing contactless card
- **Reader mode:** The NFC device is active and reads a passive RFID tag, for example for interactive advertising
- **P2P mode:** Two NFC devices communicating together and exchanging information

# NFC Applications

- **Mobile payment**
- **Mobile/electronic ticketing**
- **Smart objects**
- **Electronic keys**
- **P2P data transfers**
- NFC can be used to configure and initiate other wireless network connections such as Bluetooth or Wi-Fi



# Future of RFID and NFC



## Navigating the Airport of Tomorrow

A technology roadmap



- Trends for Brighter, Bolder, Better travel
- For more information visit: [www.amadeus.com/AirportOfTomorrow](http://www.amadeus.com/AirportOfTomorrow)
  - AMADEUS  
Your technology partner

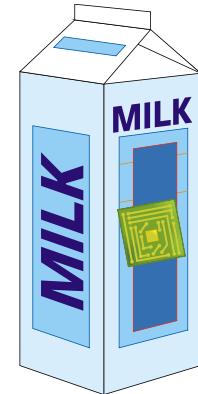
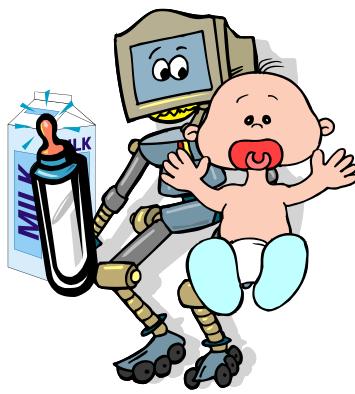


# 2030: Week in the Life of a Milk Carton



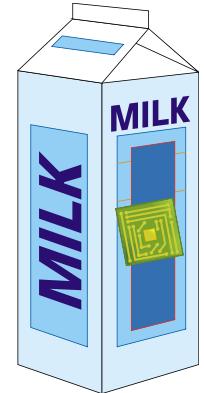
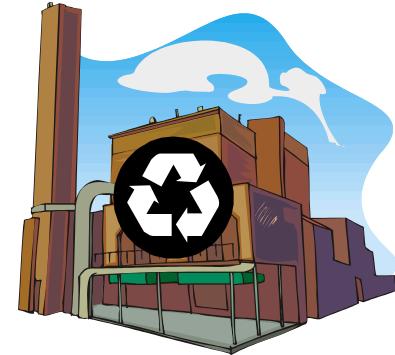
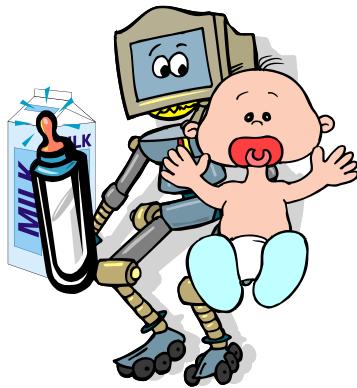
- 30 April: RFID-tagged cow 'Bessie' produces milk
- 30 April: Milk transferred to RFID-tagged tank
  - Cow identity and milking time recorded in tank-tag database
- 1 May: RFID portal on truck records loading of refrigeration tanks
  - (Truck also has active RFID (+GPS) to track geographical location and RFID transponder to pay tolls)
- 2 May: Chemical-treatment record written to database record for milk barrel
  - Bessie's herd recorded to have consumed bitter grass; compensatory sugars added
- 3 May: Milk packaged in RFID-tagged carton; milk pedigree recorded in database associated with carton tag
- 4 May: RFID portal at supermarket loading dock records arrival of carton
- 5 May: 'Smart' shelf records arrival of carton in customer area
- 5 May 0930h: 'Smart' shelf records removal of milk
- 5 May 0953h: Point-of-sale terminal records sale of milk (to Alice)

# 2030: Week in the Life of a Milk Carton



- 6 May 0953h: Supermarket transfers tag ownership to Alice's smart home
- 6 May 1103h: Alice's refrigerator records arrival of milk
- 6 May 1405h: Alice's refrigerator records removal of milk; refrigerator looks up database-recorded pedigree and displays: "*Woodstock, Vermont, Grade A, light pasturization, artisanal, USDA organic, breed: Jersey, genetic design #81726*"
- 6 May 1807h: Alice's 'smart' home warns domestic robot that milk has been left out of refrigerator for more than four hours
- 6 May 1809h: Alice's refrigerator records replacement of milk
- 7 May 0530h: Domestic robot uses RFID tag to locate milk in refrigerator; refills baby bottle

# 2030: Week in the Life of a Milk Carton



- 7 May 0530h: Domestic robot uses RFID tag to locate milk in refrigerator; refills baby bottle
- 7 May 0531h: Robot discards carton; ‘Smart’ refrigerator notes absence of milk; transfers order to Alice’s PDA/phone/portable server grocery list
- 7 May 2357h: Recycling center scans RFID tag on carton; directs carton to paper-brick recycling substation

# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# MQTT Protocol

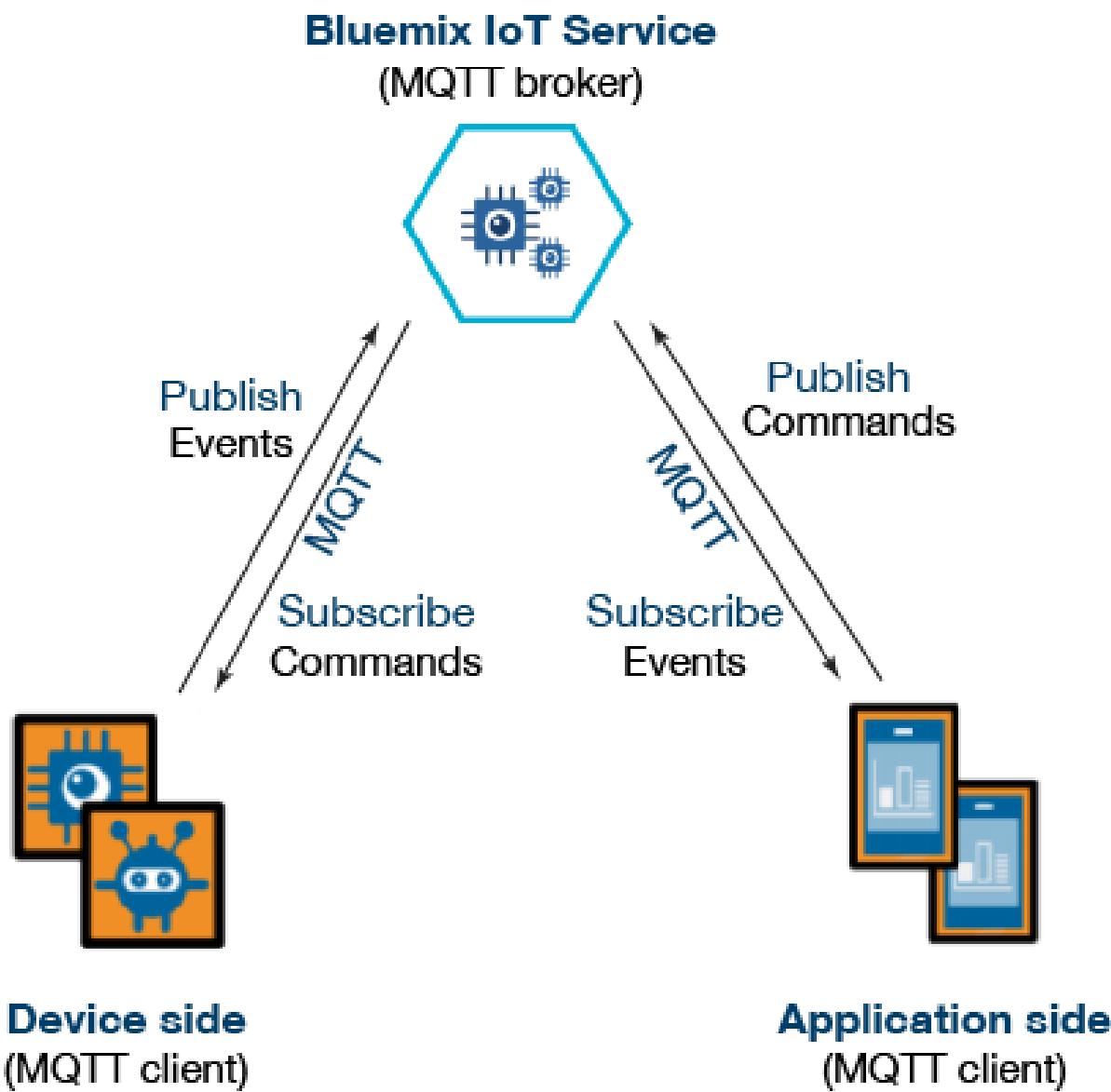
Internet-of-Things (IoT)

COCSC20

# MQTT

- Message Queuing Telemetry Transport
- A lightweight publish/subscribe protocol with predictable bi-directional message delivery.
- It is M2M/IoT connectivity protocol.
- MQTT is an Event based IoT middleware (one to many)
- In a nutshell, MQTT consist of three parts:
  - Broker
  - Subscribers
  - Publishers

# MQTT



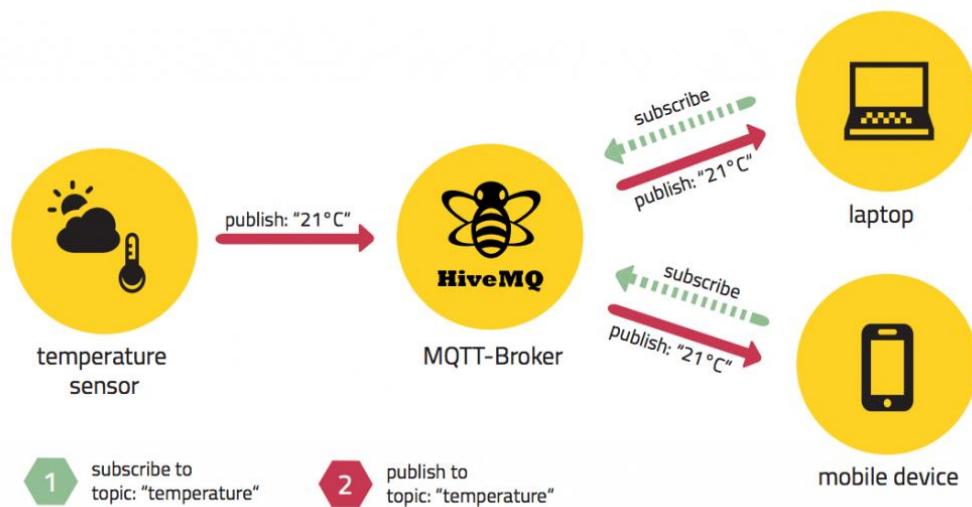
# MQTT

- MQTT was invented by Andy Stanford-Clark (IBM) and Arlen Nipper back in 1999, where their use case was to create a protocol for **minimal battery loss** and **minimal bandwidth** connecting oil pipelines over satellite connections. They specified the following goals, which the future protocol should have:
  - Simple to implement
  - Provide a Quality of Service Data Delivery
  - Lightweight and Bandwidth Efficient
  - Data Agnostic
  - Continuous Session Awareness

# MQTT

- Built for **proprietary embedded systems**; now shifting to IoT.
- You can send **anything as a message**; up to 256 MB.
- Built for **unreliable** networks.
- Enterprise scale implementations down to **hobby projects**
- Decouples readers and writers.
- Message have a topic, quality of service, and retain status associated with them.

# Publish/Subscribe Concept



## Decoupled in space and time:

The clients do not need each others IP address and port (space) and they do not need to be running at the same time (time).

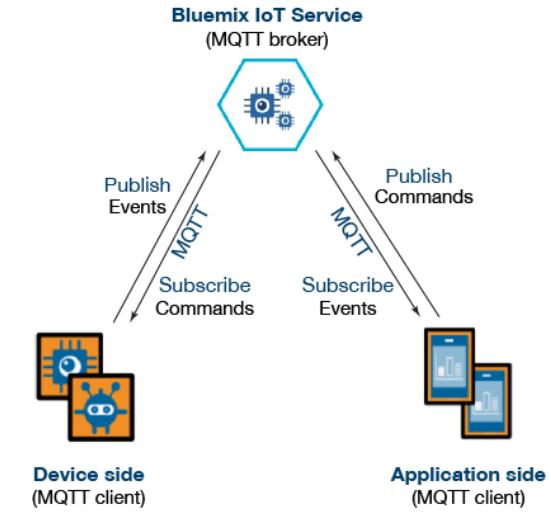
The **broker's IP and port** must be known by clients.

**Namespace** hierarchy used for topic filtering.

It may be the case that a published message is never consumed by any Subscriber.

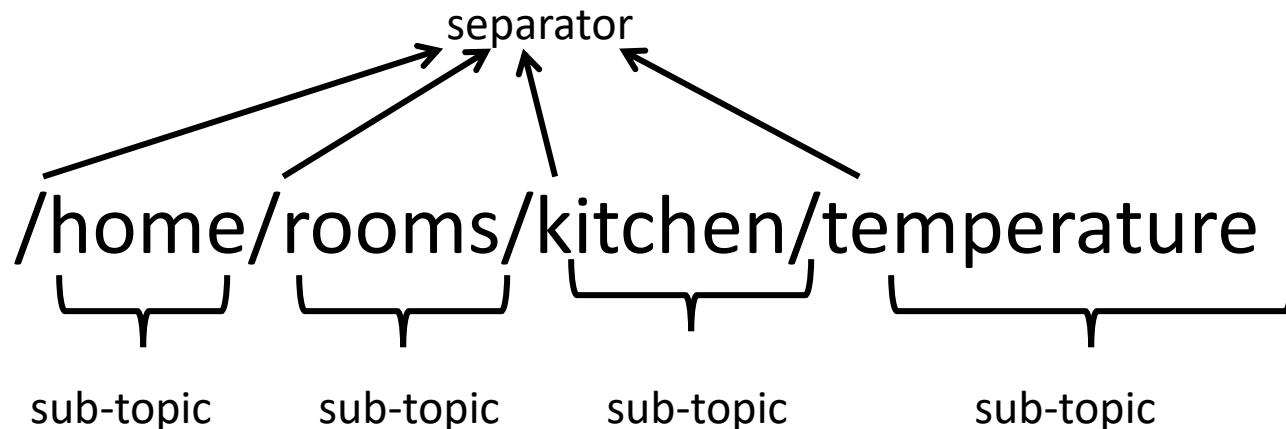
# MQTT: Example

- Clients **connect** to a “Broker”
- Clients **subscribe** to topics e.g.,
  - `client.subscribe('toggleLight/1')`
  - `client.subscribe('toggleLight/2')`
  - `client.subscribe('toggleLight/3')`
- Clients can **publish** messages to topics:
  - `client.publish('toggleLight/1', 'toggle');`
  - `client.publish('toggleLight/2', 'toggle');`
- All clients receive all messages published to topics they subscribe to
- **Messages can be anything**
  - Text
  - Images
  - etc.



# Topics

- Each published data specifies a topic
- Each subscriber subscribed to that topic will receive it.
- Topic format:



# Durable/Transient Subscriptions

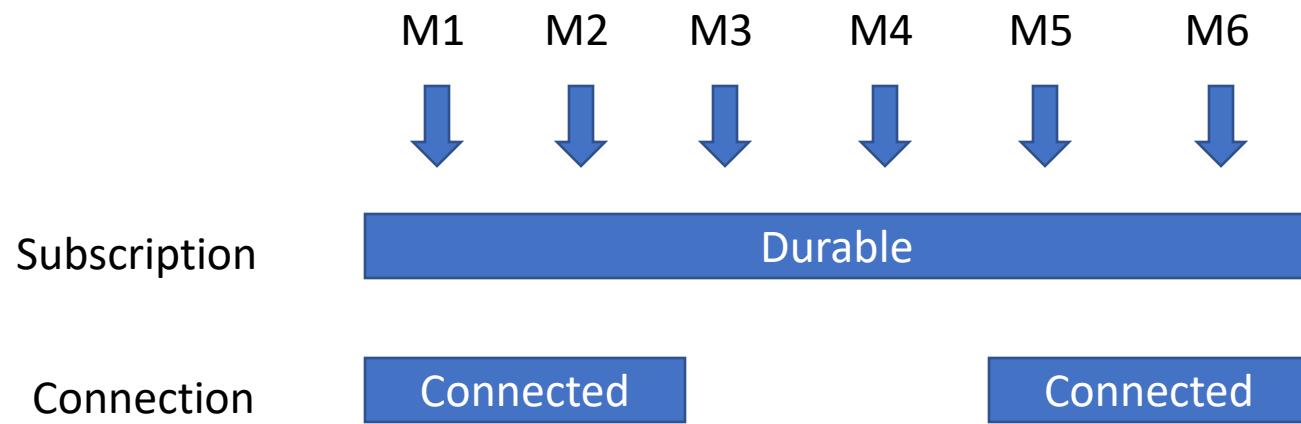
- Subscriptions

- Durable

- If the subscriber disconnects, messages are buffered at the broker and delivered upon reconnection

- Non-durable

- Connection lifetime gives subscription lifetime



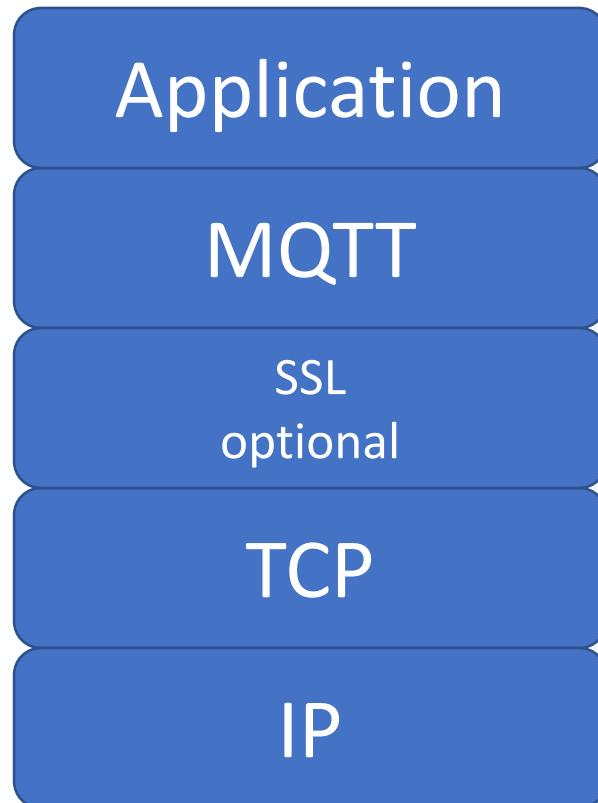
# State Retention

- Publications
  - Retained (“persistent” message)
    - The subscriber upon first connection receives the last good publication (i.e., does not have to wait for new publication).
  - One flag set both in the publish packet to the broker and subscribers
    - Only the most recent persistent message is stored and distributed

# Session Aware

- Last Will and Testament (LWT) – topic published upon disconnecting a connection.
- Any client can register an LWT.
- Anybody subscribing to the LWT topic will know when a certain device (that registered a LWT) disconnected.

# Protocol Stack



TCP/IP Port: 1883

When running over SSL, TCP/IP port 8883

SSL: Secure Socket Layer (encryption)

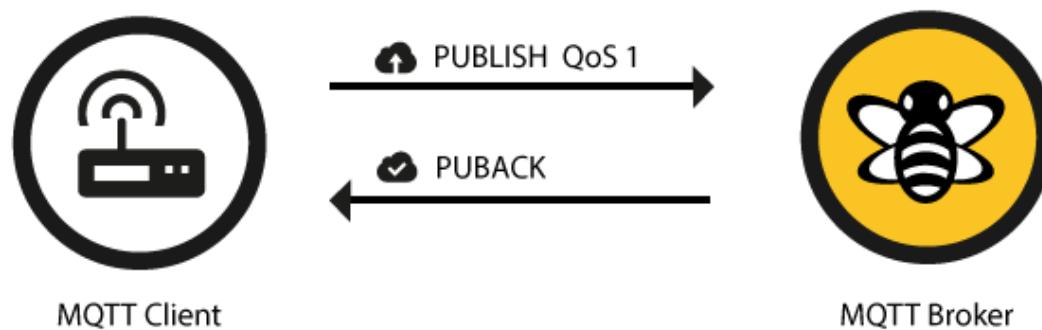
# Publishing “QoS” (Reliability)

- 0 – unreliable (aka “at most once”)
  - OK for continuous streams, least overhead (1 message)
  - “Fire and forget”
  - TCP will still provide reliability



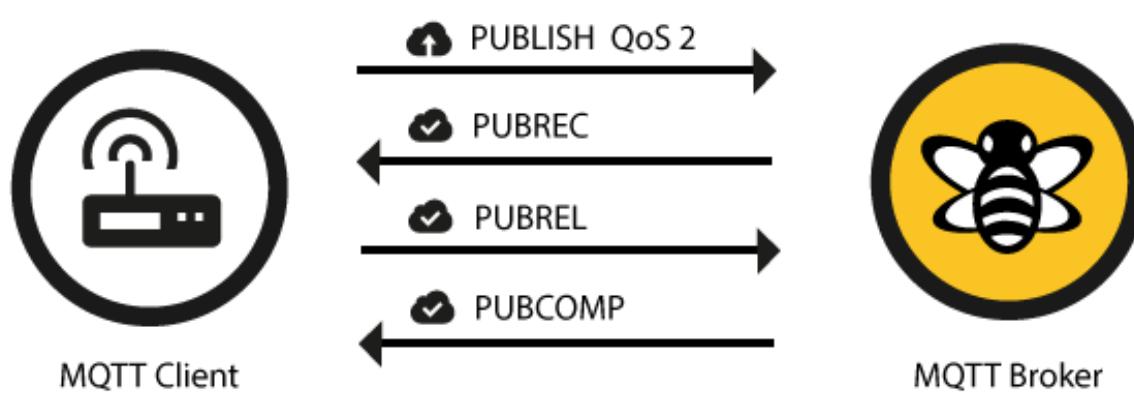
# Publishing “QoS” (Reliability)

- 1 – delivery “at least once” (duplicates possible)
  - Used for alarms – more overhead (2 messages)
  - Contains message ID (to match with ACKed message)



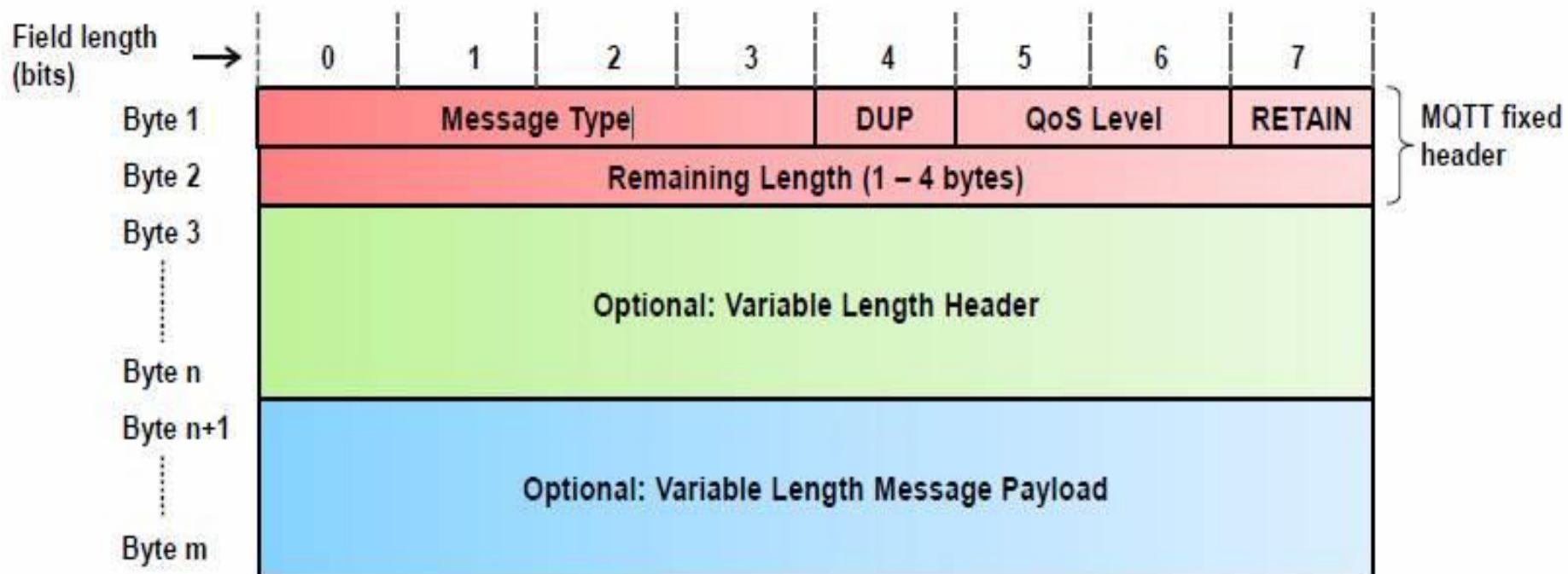
# Publishing “QoS” (Reliability)

- 2 – delivery “exactly once”
  - Utmost reliability is important – most overhead (4 messages) and slowest



# MQTT Message Format

Shortest Message is Two Bytes



# Message Types

Name	Value	Direction of flow	Description
Reserved	0	Forbidden	Reserved
CONNECT	1	Client to Server	Client request to connect to Server
CONNACK	2	Server to Client	Connect acknowledgment
PUBLISH	3	Client to Server or Server to Client	Publish message
		Server to Client	
PUBACK	4	Client to Server or Server to Client	Publish acknowledgment
PUBREC	5	Client to Server or Server to Client	Publish received (assured delivery part 1)
PUBREL	6	Client to Server or Server to Client	Publish release (assured delivery part 2)
PUBCOMP	7	Client to Server or Server to Client	Publish complete (assured delivery part 3)
SUBSCRIBE	8	Client to Server	Client subscribe request
SUBACK	9	Server to Client	Subscribe acknowledgment
UNSUBSCRIBE	10	Client to Server	Unsubscribe request
UNSUBACK	11	Server to Client	Unsubscribe acknowledgment
PINGREQ	12	Client to Server	PING request
PINGRESP	13	Server to Client	PING response
DISCONNECT	14	Client to Server	Client is disconnecting
Reserved	15	Forbidden	Reserved

# MQTT Fixed Header

Message fixed header field	Description / Values	
Message Type	0: Reserved	8: SUBSCRIBE
	1: CONNECT	9: SUBACK
	2: CONNACK	10: UNSUBSCRIBE
	3: PUBLISH	11: UNSUBACK
	4: PUBACK	12: PINGREQ
	5: PUBREC	13: PINGRESP
	6: PUBREL	14: DISCONNECT
	7: PUBCOMP	15: Reserved
DUP	Duplicate message flag. Indicates to the receiver that this message may have already been received. 1: Client or server (broker) re-delivers a PUBLISH, PUBREL, SUBSCRIBE or UNSUBSCRIBE message (duplicate message).	
QoS Level	Indicates the level of delivery assurance of a PUBLISH message. 0: At-most-once delivery, no guarantees, «Fire and Forget». 1: At-least-once delivery, acknowledged delivery. 2: Exactly-once delivery. Further details see <a href="#">MQTT QoS</a> .	
RETAIN	1: Instructs the server to retain the last received PUBLISH message and deliver it as a first message to new subscriptions. Further details see <a href="#">RETAIN (keep last message)</a> .	
Remaining Length	Indicates the number of remaining bytes in the message, i.e. the length of the (optional) variable length header and (optional) payload. Further details see <a href="#">Remaining length (RL)</a> .	

# Thank You

Contact me:

[gauravsingal789@gmail.com](mailto:gauravsingal789@gmail.com)

[Gaurav.singal@nsut.ac.in](mailto:Gaurav.singal@nsut.ac.in)

[www.gauravsingal.in](http://www.gauravsingal.in)

LinkedIn: <https://www.linkedin.com/in/gauravsingal789/>

Twitter: [https://twitter.com/gaurav\\_singal](https://twitter.com/gaurav_singal)

# Comparison CoAP & MQTT

Both used in IoT

- CoAP:
  - one-to-one communication
  - UDP/IP
  - unreliable
  - lightweight and easy to implement
- MQTT:
  - many-to-many communication
  - TCP/IP
  - focus on message delivery; reliable
  - higher overheads (protocol data, processing costs)

# CoAP

IoT/Web Application Layer Protocols

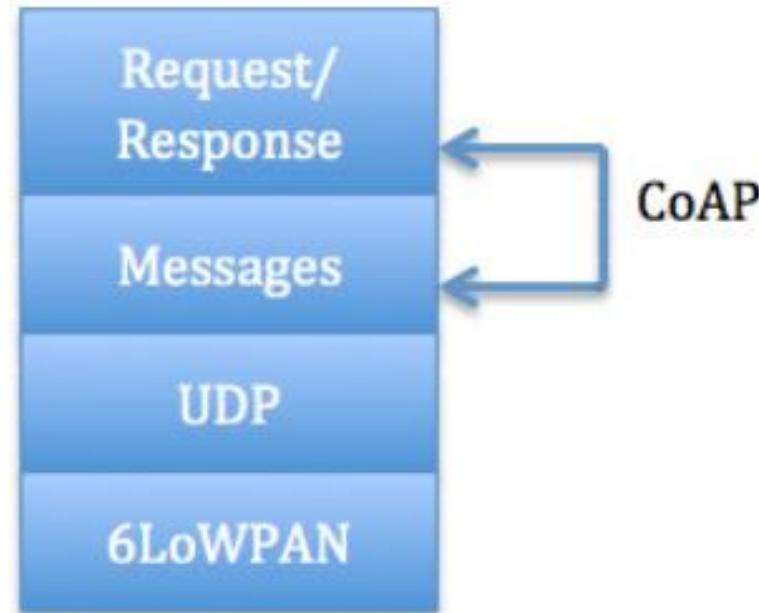
COCSC20 (Internet of Things)

# CoAP (Constrained Application Protocol)

- **REST-based** web transfer protocol.
- **manipulates** Web resources using the same methods as HTTP: GET, PUT, POST, and DELETE
- **subset** of HTTP functionality re-designed for low power embedded devices such as sensors (for IoT and M2M).
- Constrained RESTful Environments (CoRE) **workgroup** who has designed CoAP.

# CoAP

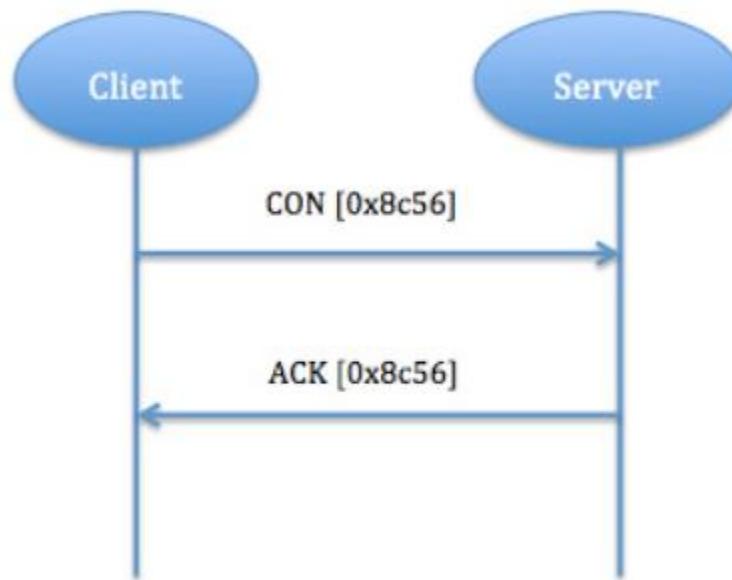
- TCP overhead is too high and its flow control is not appropriate for short-lived transactions.
- UDP has lower overhead and supports multicast.



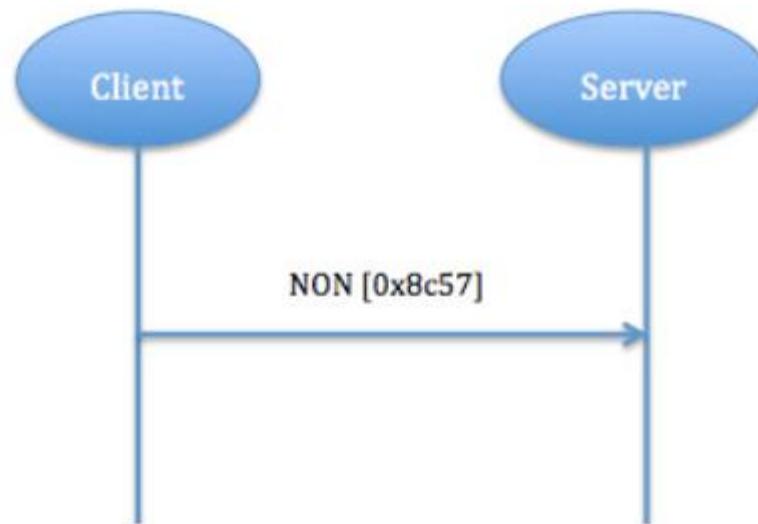
# CoAP: Message layer

- Four message types:
  - **Confirmable (00)** – requires an ACK
  - **Non-confirmable (01)** – no ACK needed
  - **Acknowledgement (10)**– ACKs a Confirmable
  - **Reset (11)**- indicates a Confirmable message has been received but context is missing for processing

# Reliable Message Transport

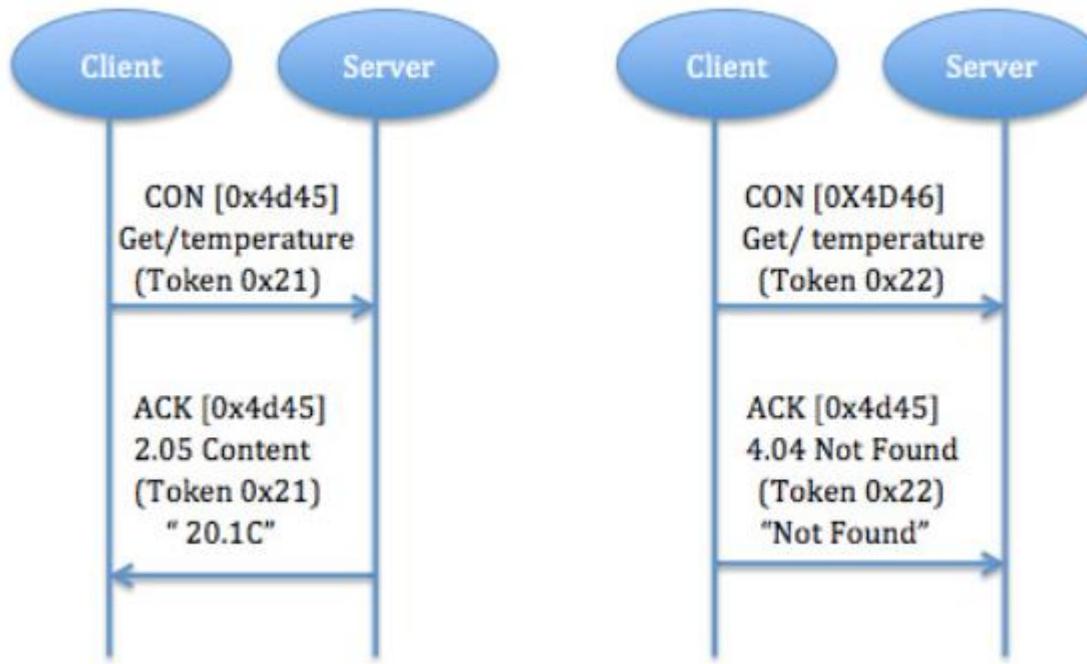


# Unreliable Message Transport



# CoAP: request/response Layer

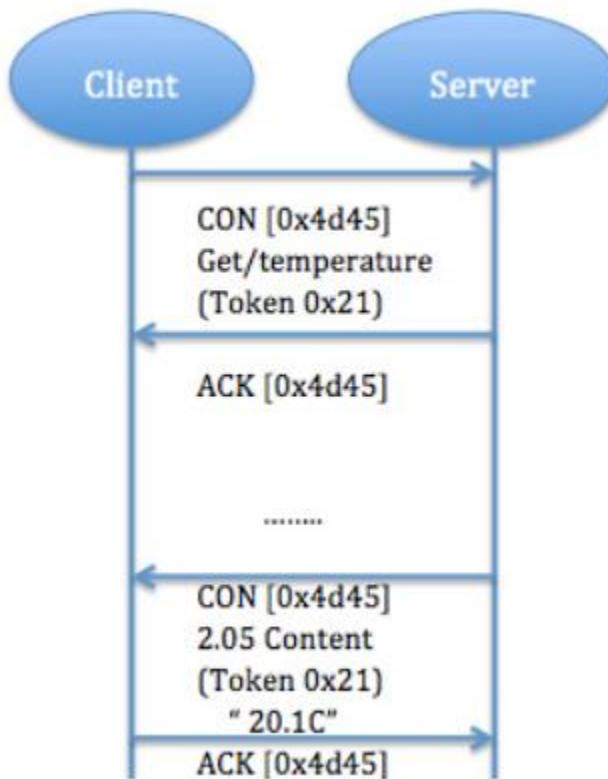
Piggy-backed



The successful and failure response results of GET method

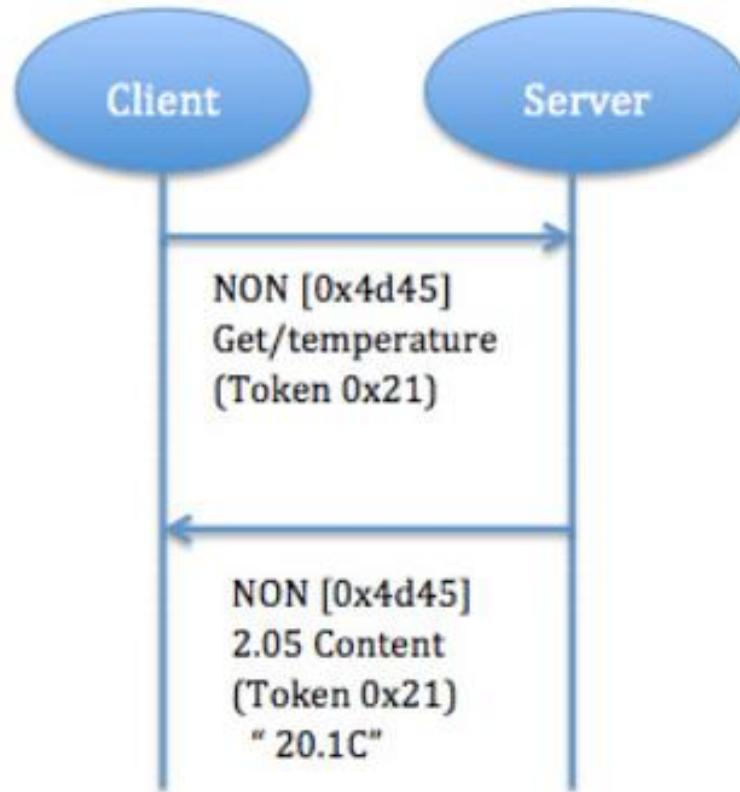
# CoAP: request/response Layer

Separate response

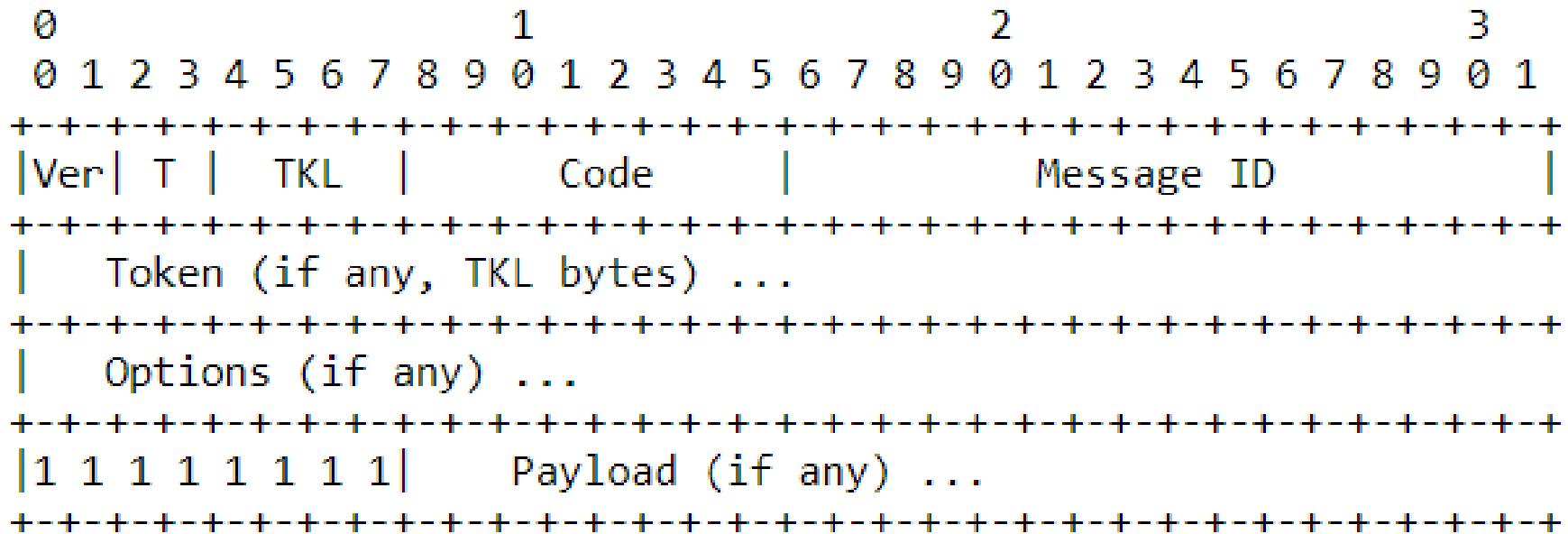


# CoAP: request/response Layer

Non confirmable request and response



# Message Format



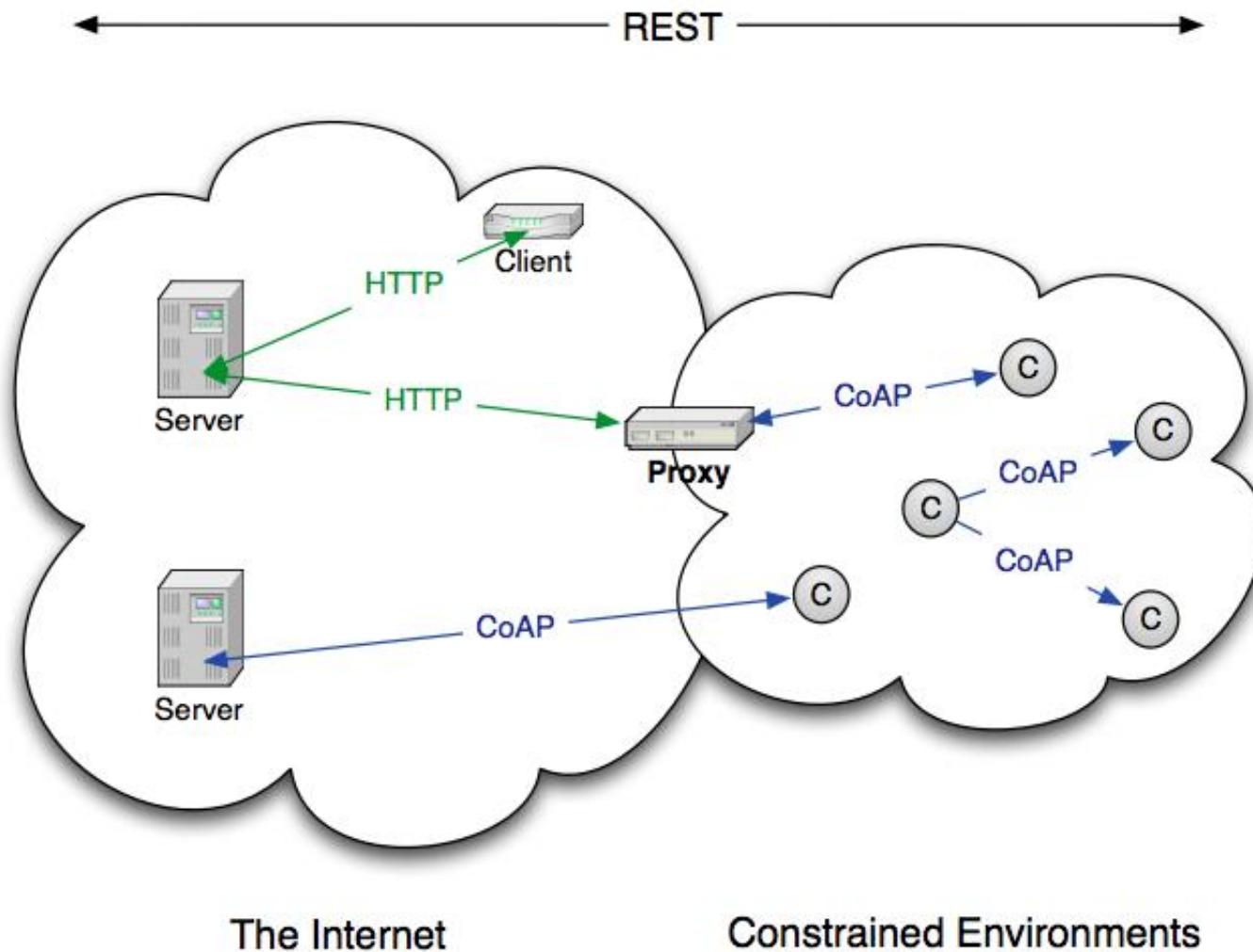
Example:

44 01 C4 09 74 65 73 74 B7 65 78 61 6D 70 6C 65

44 01 C4 09 74 65 73 74 B7 65 78 61 6D 70 6C 65

Field	HEX	Bits	Meaning
Ver	44	01	Version 01, which is mandatory here.
T		00	Type 0: confirmable.
TKL		0100	Token length: 4.
Code	01	000 00001	Code: 0.01, which indicates the GET method.
Message ID	C4 09	2 Bytes equal to hex at left	Message ID. The response message will have the same ID. This can help out identification.
Token	74 65 73 74	4 Bytes equal to hex at left	Token. The response message will have the same token. This can help out identification.
Option delta	B7	1011	Delta option: 11 indicates the option data is Uri-Path.
Option length		0111	Delta length: 7 indicates there are 7 bytes of data following as a part of this delta option.
Option value	65 78 61 6D 70 6C 65	7 Bytes equal to hex at left	Example.

# CoAP



# CoAP

- CoAP provides reliability without using TCP as transport protocol.
- CoAP enables asynchronous communication
  - e.g., when CoAP server receives a request which it cannot handle immediately, it first ACKs the reception of the message and sends back the response in an off-line fashion
- Also supports multicast and congestion control.

# What CoAP Is

- CoAP is
  - A RESTful protocol
  - Both synchronous and asynchronous
  - For constrained devices and networks
  - Specialized for M2M applications
  - Easy to proxy to/from HTTP

# Comparison between CoAP & MQTT

Both used in IoT

- CoAP:
  - One-to-one communication
  - UDP/IP
  - Unreliable
  - Lightweight and easy to implement
- MQTT:
  - Many-to-many communication
  - TCP/IP
  - Focus on message delivery; reliable
  - Higher overheads (protocol data, processing costs)

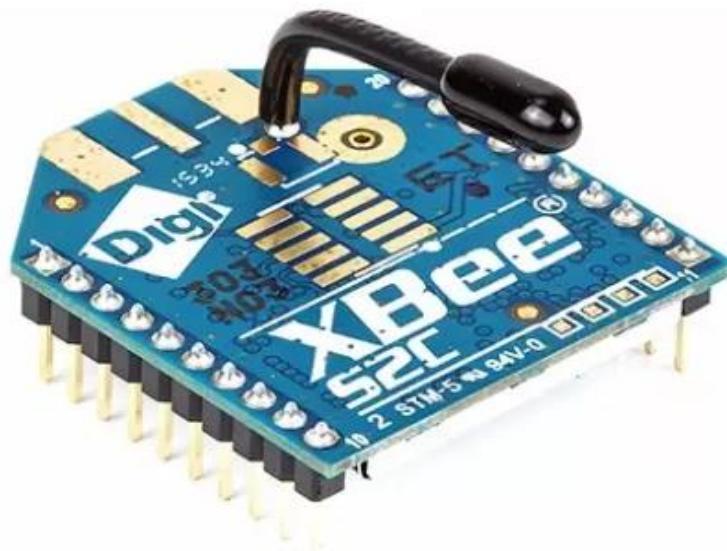
# Performance Evaluation - Background

Criteria	HTTP	CoAP	MQTT
Architecture	Client/Server	Client/Server or Client/Broker	Client/Broker
Abstraction	Request/Response	Request/Response or Publish/Subscribe	Publish/Subscribe
Header Size	Undefined	4 Byte	2 Byte
Message size	Large and Undefined (depends on the web server or the programming technology)	Small and Undefined (normally small to fit in single IP datagram)	Small and Undefined (up to 256 MB maximum size)
Semantics/Methods	Get, Post, Head, Put, Patch, Options, Connect, Delete	Get, Post, Put, Delete	Connect, Disconnect, Publish, Subscribe, Unsubscribe, Close
Quality of Service (QoS) /Reliability	Limited (via Transport Protocol - TCP)	Confirmable Message or Non-confirmable Message	QoS 0 - At most once QoS 1 - At least once QoS 2 - Exactly once
Transport Protocol	TCP	UDP, TCP	TCP (MQTT-SN can use UDP)
Security	TLS/SSL	DTLS/IPSEC	TLS/SSL
Default Port	80/443 (TLS/SSL)	5683 (UDP)/5684 (DTLS)	1883/8883 (TLS/SSL)

\* N. Naik, "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP," 2017 IEEE International Systems Engineering Symposium (ISSE), Vienna, 2017, pp. 1-7.

Thank You

# ZigBee RF Module



# ZigBee

- ZigBee is a technology standard to design for control and sensor network
- Based on IEEE 802.15.4 standard
- Ad-hoc and Mesh network creation
- Routing (pass messages on) and Self-healing
- Example of WPAN Technology
- Created by ZigBee Alliance
  - Global standards for reliable, cost-effective, low power wireless applications



# Radio Communication

- Electromagnetic Waves
- No medium required
- Modulation
- Wireless/Airwaves
- Inverse Square Law

$$\rho \propto \frac{1}{r^2}$$

r = distance,  $\rho$ = power.



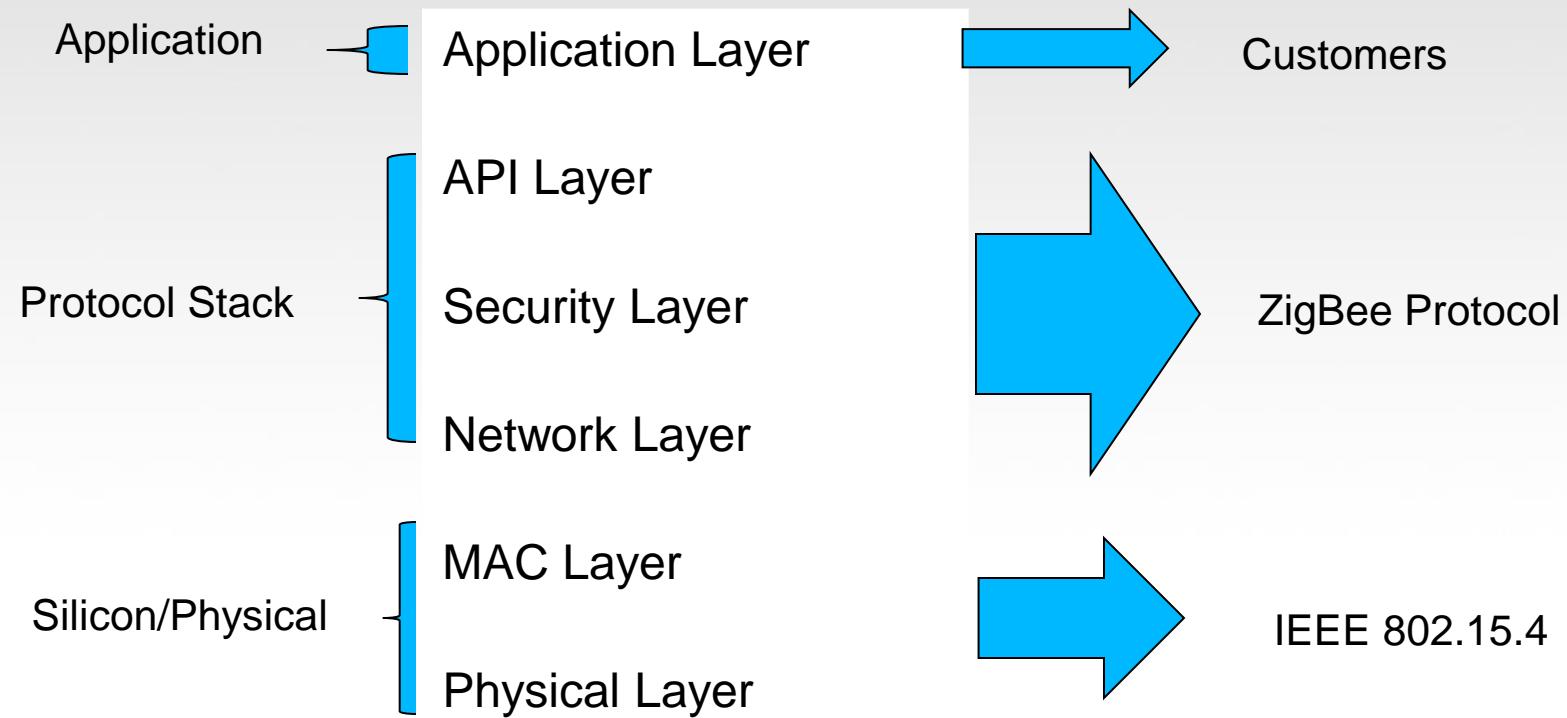
# Need of ZigBee?

- Development started 1998, when many engineers realized that Wi-Fi and Bluetooth were going to be unsuitable for many applications as
- Direct /indirect communication between multiple machines
- Non standardized IoT devices
- Low power and small size devices
- Cost effective and security.

# IEEE 802.15.4

- Low Power consumption
- Low bandwidth
- Affordable
- Small
- Standardized
- Popular

# Layering Architecture of ZigBee

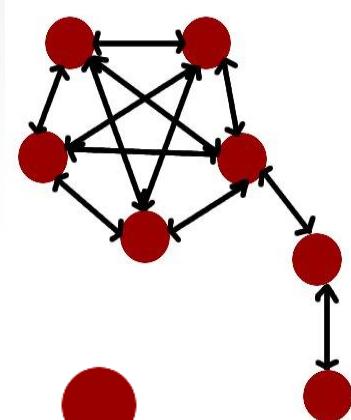


# IEEE 802.15.4 Configurations

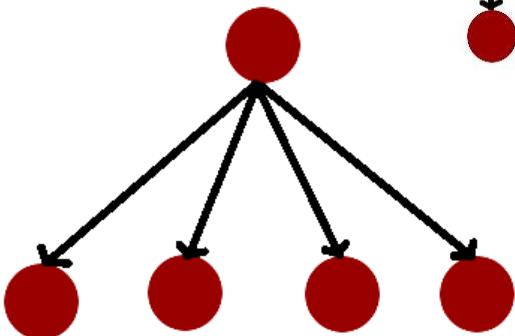
- Single Peer



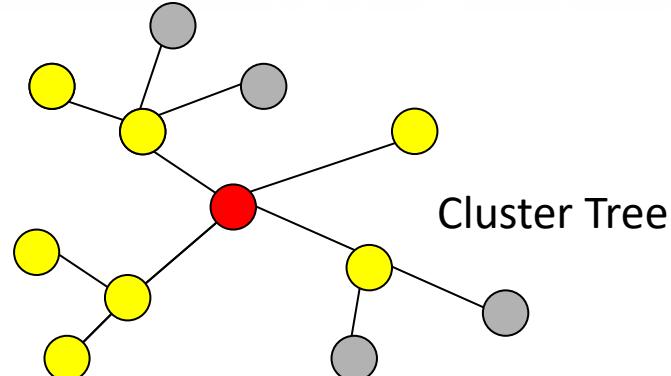
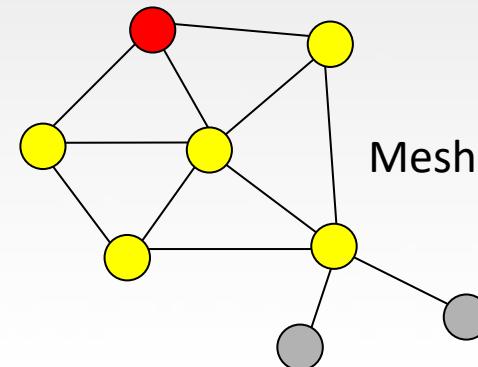
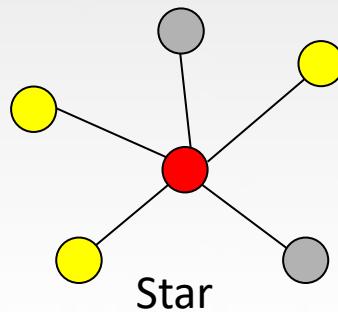
- Multi Peer



- Broadcast



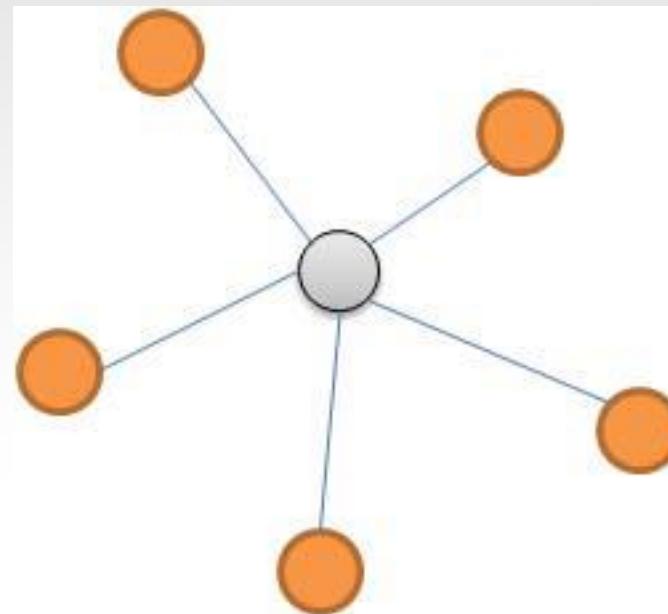
# ZigBee Network Topologies



- PAN coordinator
- Full Function Device
- Reduced Function Device

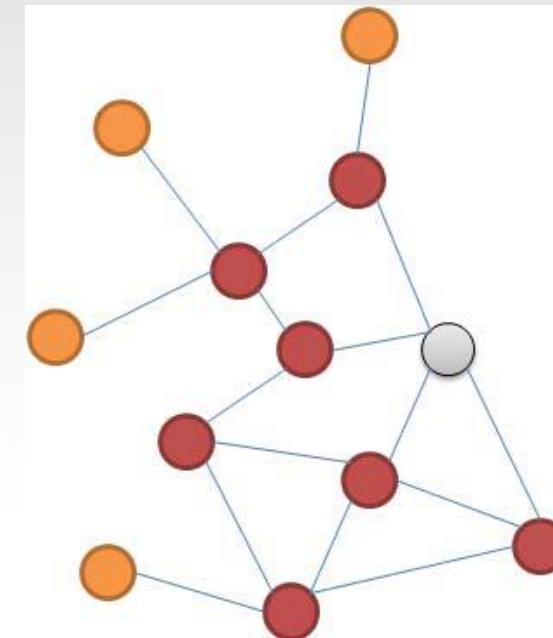
# ZigBee Network Topologies

- Star Topology
  - Advantage
    - Easy to synchronize
    - Low latency
  - Disadvantage
    - Small scale



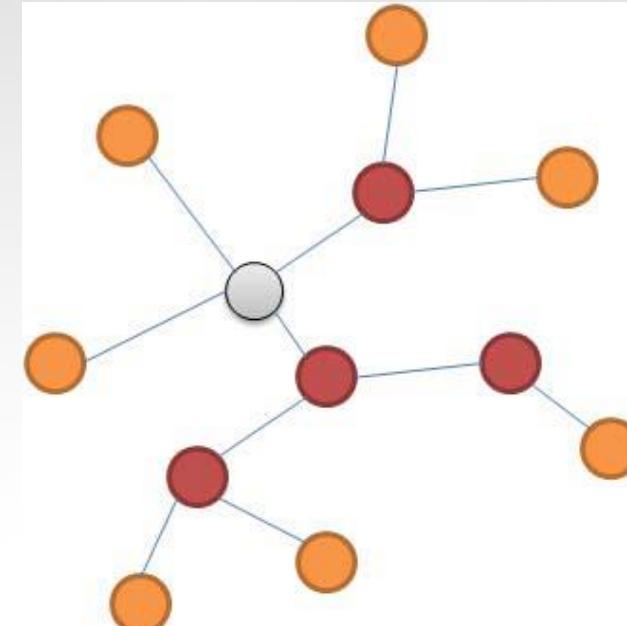
# ZigBee Network Topologies

- Mesh Topology
  - Advantage
    - Robust multi-hop communication
    - Network is more flexible
    - Lower latency
  - Disadvantage
    - Route discovery is costly
    - Needs storage for routing table



# ZigBee Network Topologies

- Cluster Tree
  - Advantage
    - Low routing cost
    - Allow multihop communication
  - Disadvantage
    - Route reconstruction is costly
    - Latency may be quite long

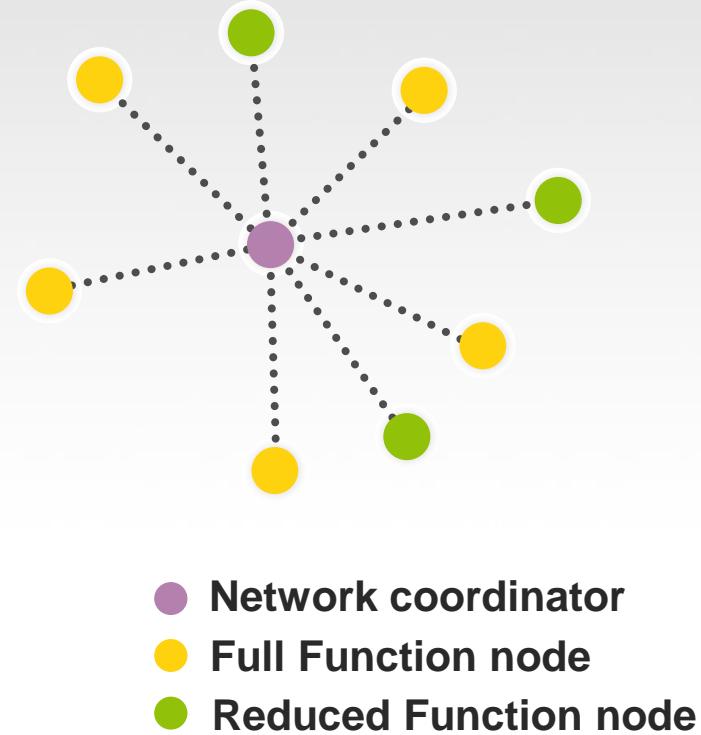


# Sensor/Control Network Requirements

- Networks form by themselves, scale to large sizes and operate for years without manual intervention
- Extremely long battery life (years on AA cell),
  - low infrastructure cost (low device & setup costs)
  - low complexity and small size
- Low device data rate and QoS (Quality-of-Service)
- Standardized protocols allow multiple vendors to interoperate.

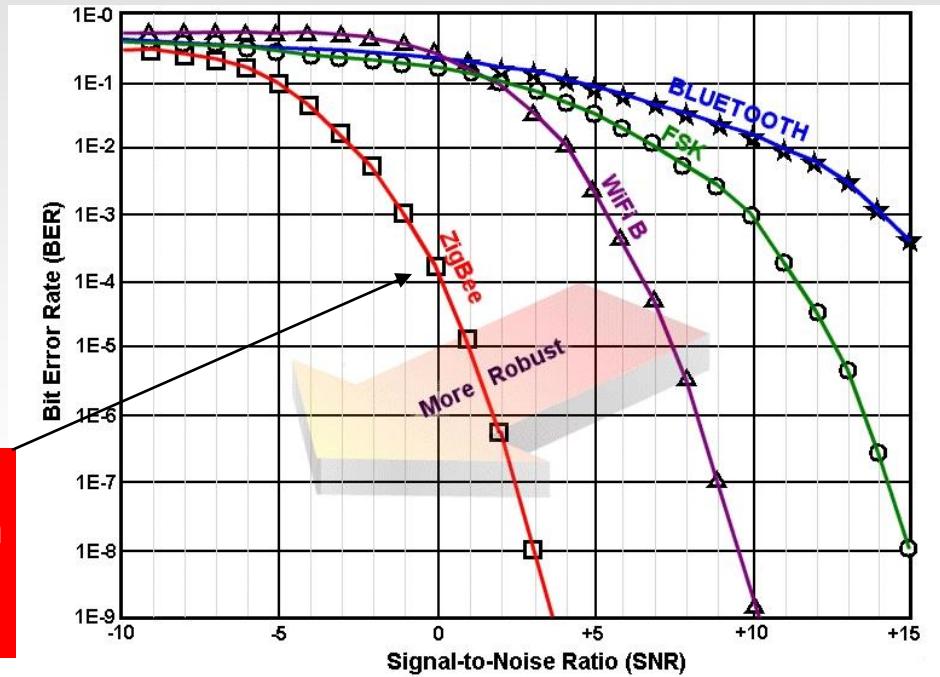
# Basic Network Characteristics

- 65,536 network (client) nodes
- 27 channels over 2 bands
- 250Kbps data rate
- Optimized for time-critical applications and power management
- Full mesh networking support



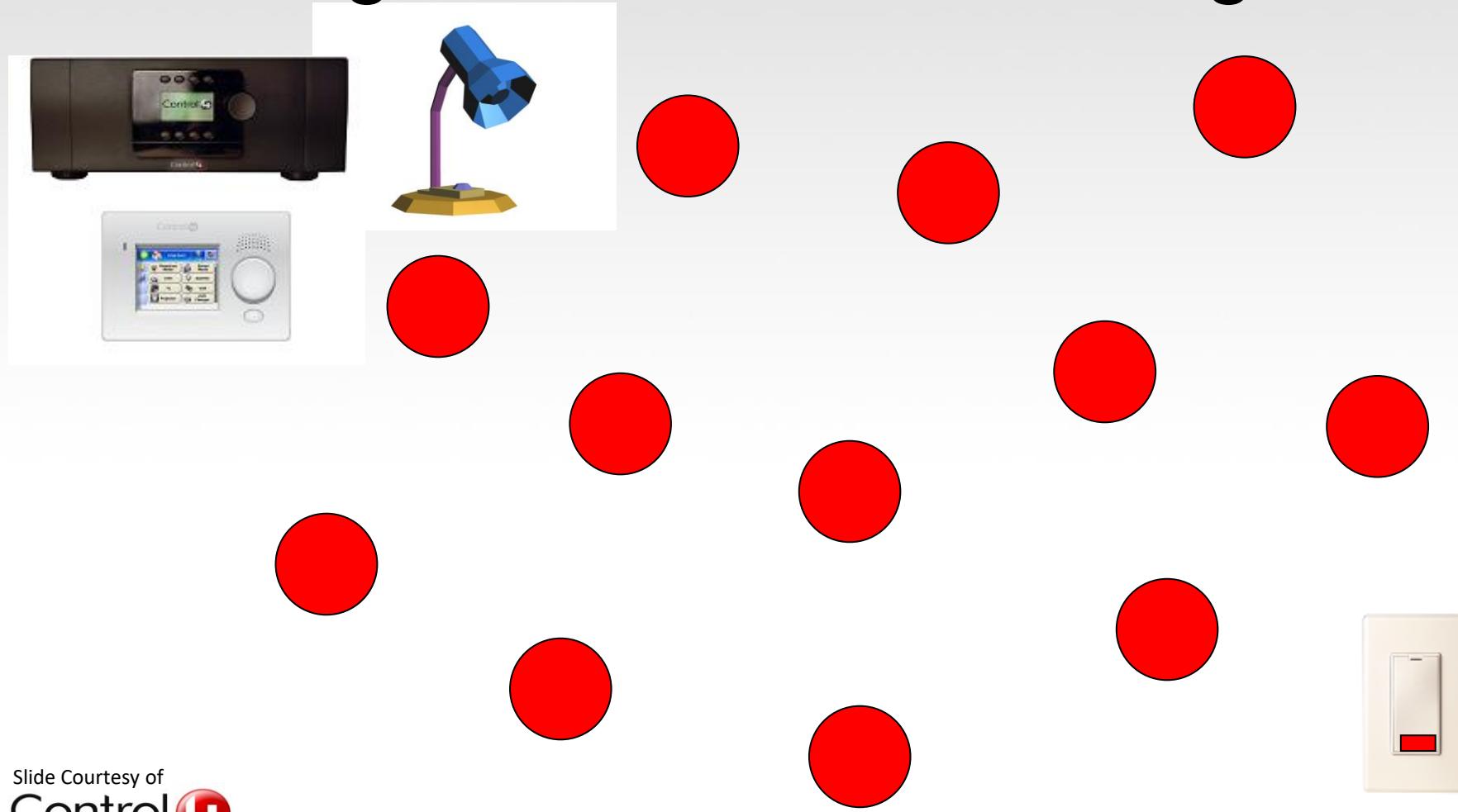
# Basic Radio Characteristics

ZigBee technology relies upon IEEE 802.15.4, which has excellent performance in low SNR environments



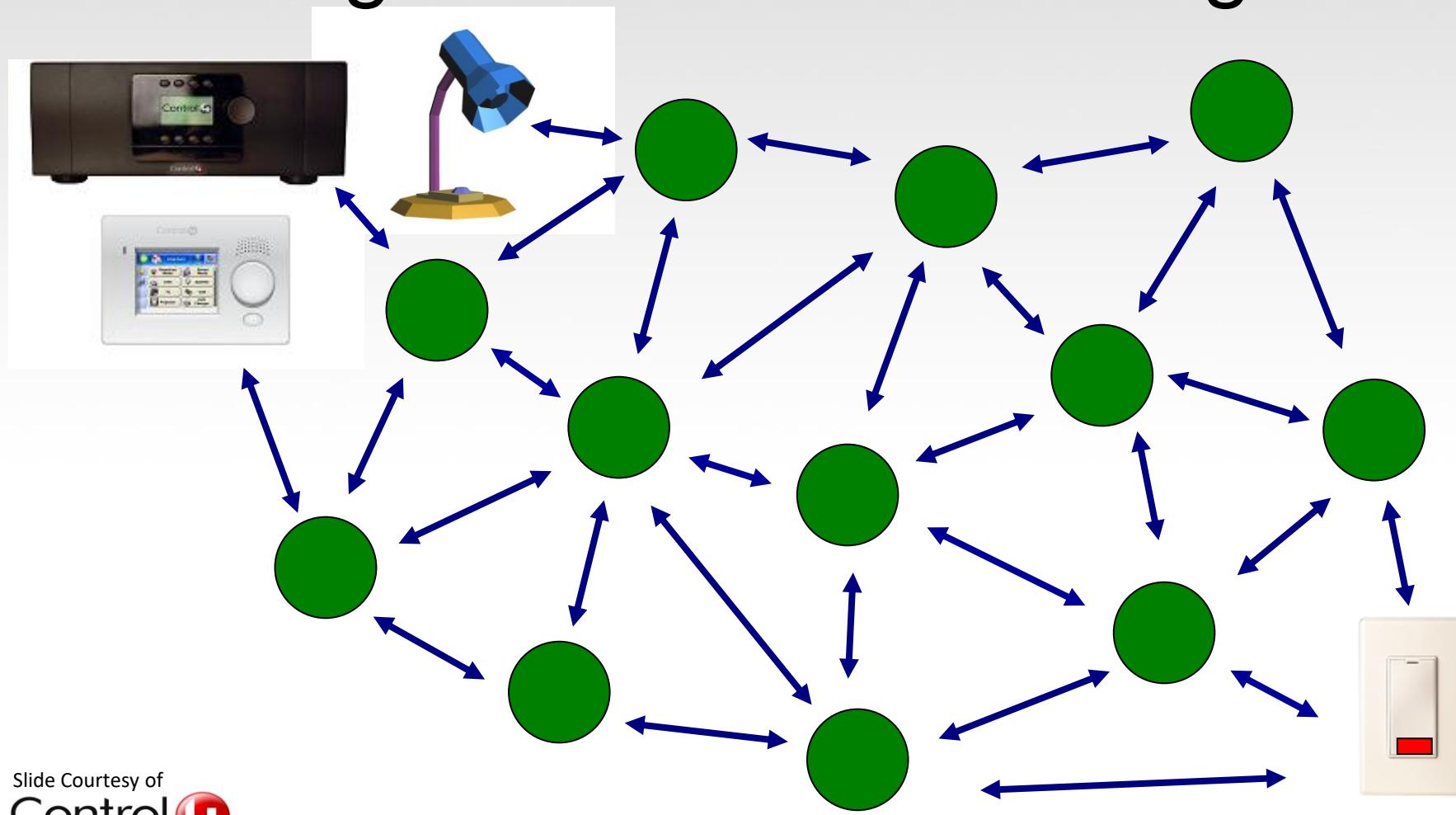
Frequency Band	License Required?	Geographic Region	Data Rate	Channel Number(s)
868.3 MHz	No	Europe	20kbps	0
902-928 MHz	No	Americas	40kbps	1-10
2405-2480 MHz	No	Worldwide	250kbps	11-26

# ZigBee Mesh Networking

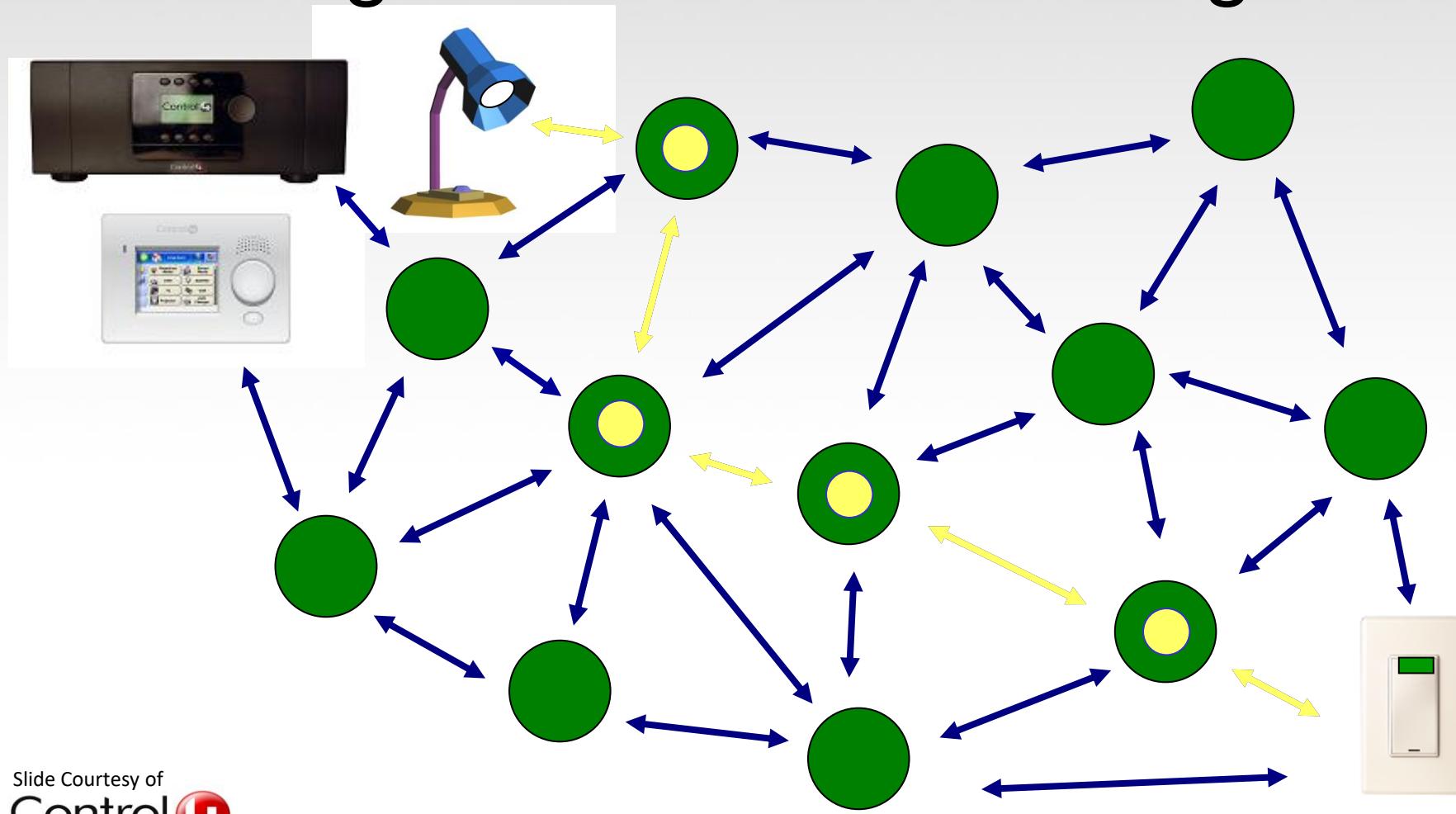


Slide Courtesy of  
**Control+**

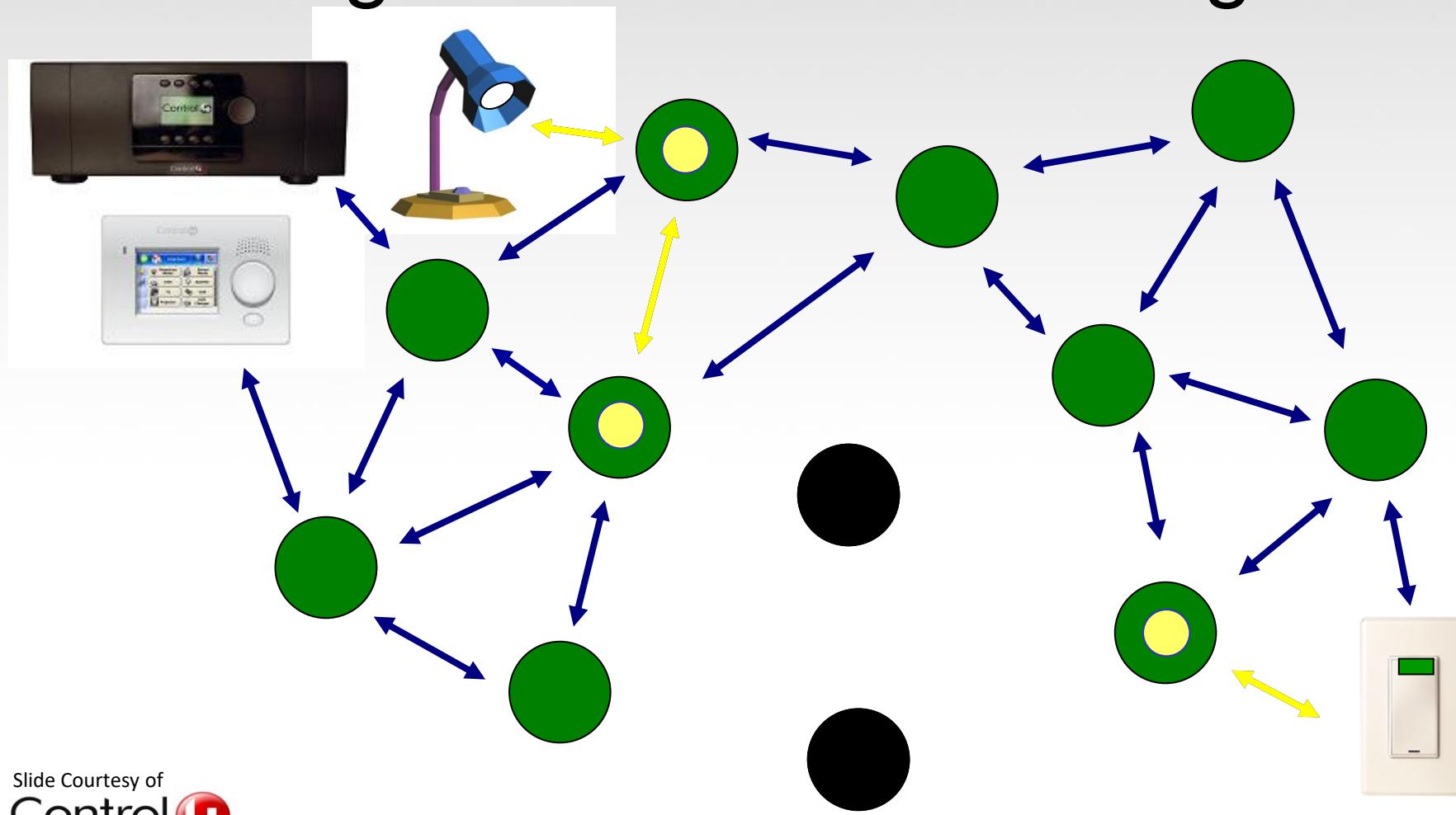
# ZigBee Mesh Networking



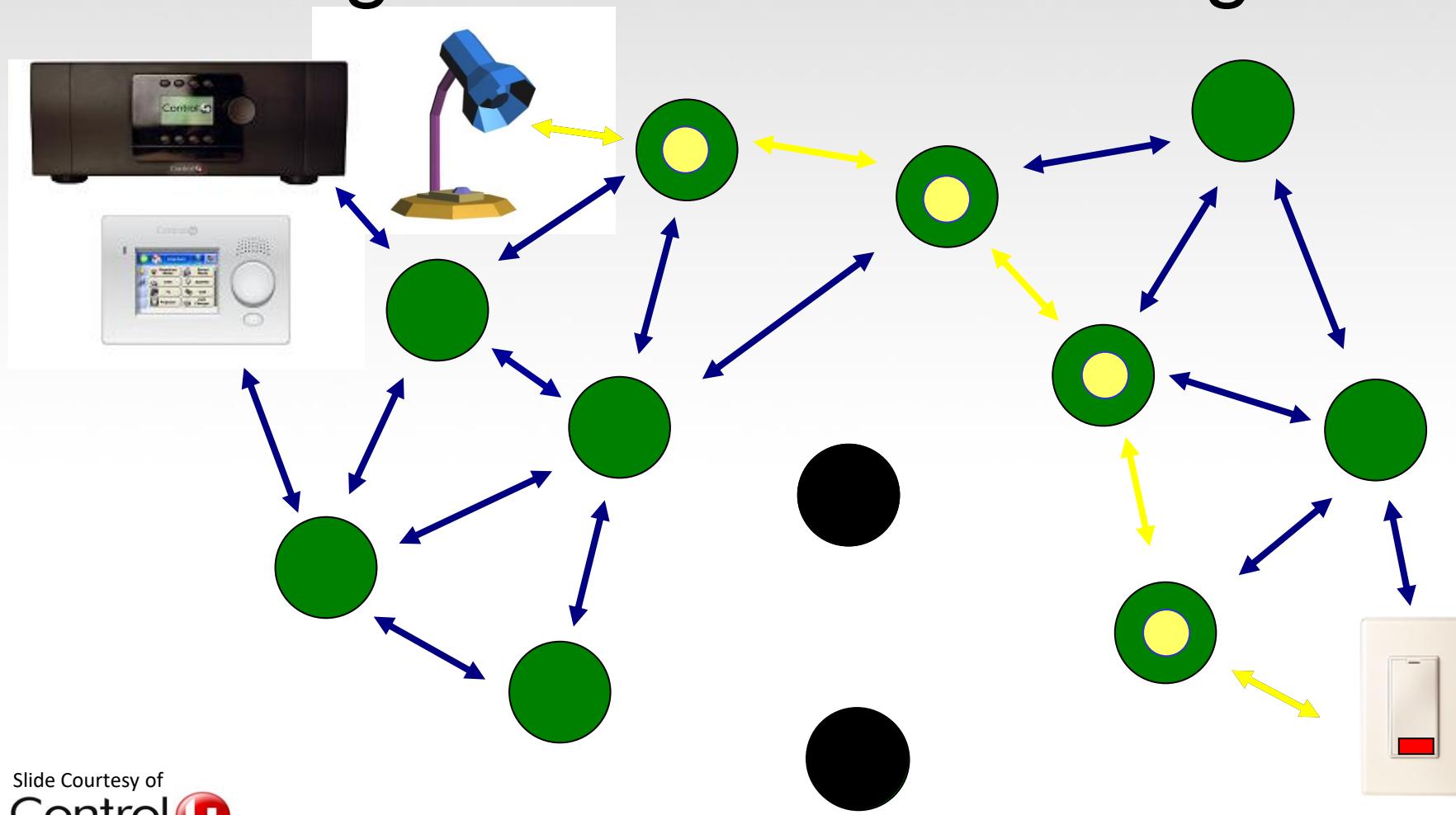
# ZigBee Mesh Networking



# ZigBee Mesh Networking



# ZigBee Mesh Networking



Wireless Connectivity Comparison				
Category	BLE	Bluetooth Classic	ZIG-BEE	Wi-Fi
				
IEEE Standard	802.15.1	802.15.1	802.15.4	802.11 (a, b, g, n)
Frequency (GHz)	2.4	2.4	0.8, 0.9, 2.4	2.4 to 5
Bandwidth (Mbps)	Low ( 1 )	Medium ( 1 to 3)	Very low (0.25)	High (54-a, 11-b, 54-g, 600-n)
Power Consumption	Very Low	Medium	Very low	High
Battery Life	Months to Years	Days	Months to Years	Hours
Network Size	Undefined	7	64,000 +	255
Range	Short to Medium	Short	Medium	Medium to Long
Ease of use	Simple to use	Fairly simple to use	Fairly simple to use, but not yet adopted in mobiles and computers	It is more complex and requires configuration of Hardware and Software
Adaptability	High	High	Low	High
Primary Applications	Wearables, Mobile phones	Mobile Phones, mouse, keyboards, office and industrial automation devices	Industrial Automation, Home Automation, Smart metering	Notebook computers, desktop computers, servers.

# XBee as a Coordinator

- Each network has 1 coordinator
- Coordinator selects channel and Create PAN ID
- Other devices then join the PAN ID
- Usually powered by something stable
- 16-bit network address is always 0
- Assigns 16-bit address for the router and end devices

# XBee as a Routers

- Router as Optional
- Often powered by something stable
- Can have as many as you want
- Issues a request on start to find a coordinator/network so it can join
- Can talk to any device
- Coordinator can act as a “super router”

# XBee as End Devices

- End devices as Optional
- Usually, battery powered
- Can have as many as you want
- Issues a request on start to find a network, it can join a parent device (router or coordinator)
- Can only communicate with its parent

# XBee ZB S1 Vs XBee ZB S2 Vs XBee s2c

Specification	XBee ZB S1	XBee ZB S2	XBee ZB S2C
Indoor/Urban range	up to 100 ft. (30m)	up to 133 ft. (40m)	up to 200 ft (60 m)
Outdoor RF line-of-sight range	up to 300 ft. (100m)	up to 400 ft. (120m)	up to 4000 ft (1200m)
Transmit Power Output	1 mW (0dbm)	2 mW (+3dbm)	6.3mW (+8dBm) Boost mode 3.1mW (+ddBm) Normal mode
RF Data Rate	250 Kbps	250 Kbps	250 Kbps
Receiver Sensitivity	-92dBm (1% PER)	-98dBm (1% PER)	-102dBm (1% PER) Boost mode -100dBm (1% PER) Normal Mode

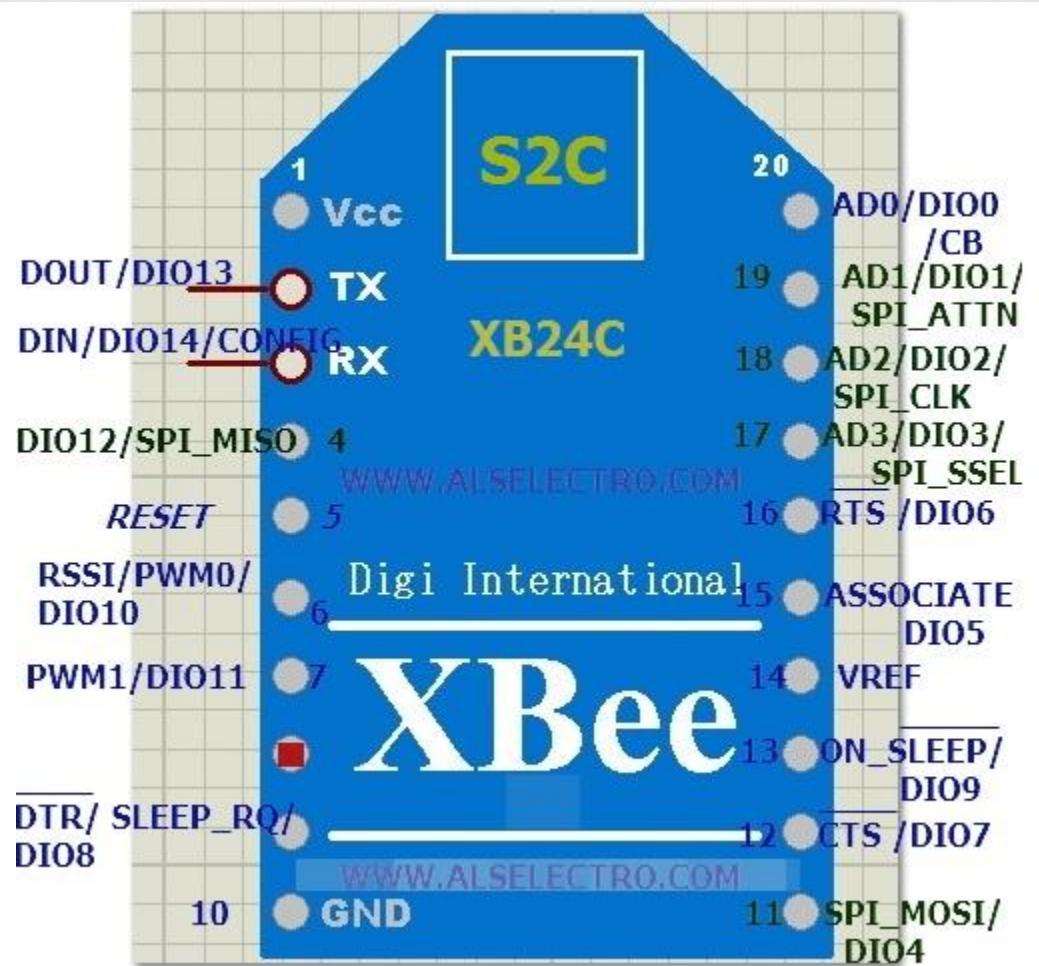
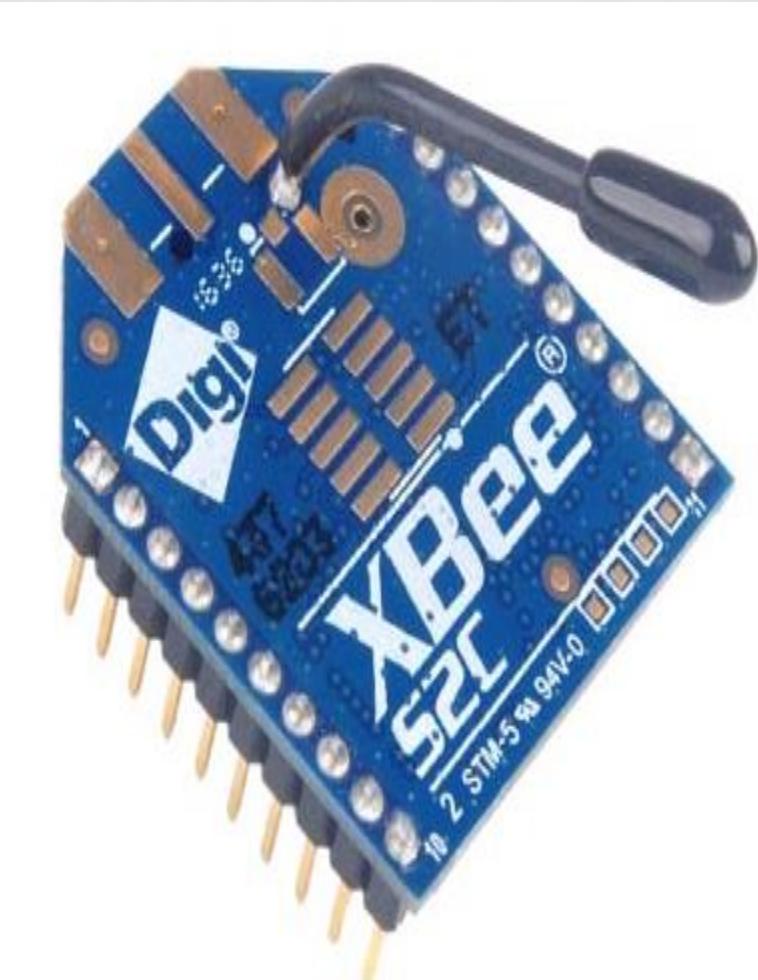
# XBee ZB S1 Vs XBee ZB S2 Vs XBee s2c

Supply Voltage	2.8 - 3.4 V	2.8 - 3.6 V	2.1 - 3.6V
Transmit Current (typical)	45 mA (@ 3.3 V)	40 mA (@ 3.3 V)	45 mA (+8dBm) Boost Mode 33 mA (+5dBm) Normal Mode
Idle/Receive Current (typical)	50 mA (@ 3.3 V)	40 mA (@ 3.3 V)	31 mA (+8dBm) Boost Mode 28 mA (+5dBm) Normal Mode
Power-down Current	10 uA	1 uA	<1uA
Frequency	ISM 2.4 GHz	ISM 2.4 GHz	ISM 2.4 GHz

# XBee ZB S1 Vs XBee ZB S2 Vs XBee s2c

Network Topologies	Point to point, Star, Mesh (with DigiMesh firmware)	Point to point, Star, Mesh	Point to point, Star, Mesh
Number of Channels	16 Direct Sequence Channels	16 Direct Sequence Channels	16 Direct Sequence Channels
Filtration Options	PAN ID, Channel & Source/Destination	PAN ID, Channel & Source/Destination	PAN ID, Channel & Source/Destination

# XBee s2c Pin out Diagram



# Pin signals for XBee s2c

Pin	Name	Direction	Description
1	VCC		Power supply
2	DOUT	Output	UART data out
3	DIN/CONFIG	Input	UART data In
4	DIO12/SPI_MISO	Both	Digital I/O 12 / Serial Peripheral Interface (SPI) Data Out
5	RESET	Input	Module reset (reset pulse must be at least 200 ns). This must be driven as an open drain/collector. The device drives this line low when a reset occurs. Never drive this line high.
6	DIO10/PWM0/RSSI PWM	Both	Digital I/O 10 / PWM output 0 / RX signal strength indicator
7	DIO11/PWM1	Both	Digital I/O 11 / PWM output 1
8	[Reserved] - Do not connect		
9	DIO8/SLEEP_RQ/DTR	Both	Digital I/O 8 / Pin sleep control line
10	GND		Ground
11	DIO4/SPI_MOSI	Both	Digital I/O 4 / SPI Data In
12	DIO7/CTS	Both	Digital I/O 7 / Clear-to-send flow control
13	ON/SLEEP	Output	Device sleep status indicator
14	VREF		Feature not supported on this device. Used on other XBee devices for analog voltage reference.

# Pin signals for XBee s2c Cont..

<b>Pin</b>	<b>Name</b>	<b>Direction</b>	<b>Description</b>
15	DIO5/ASSOC	Both	Digital I/O 5 / Associated indicator
16	DIO6/RTS	Both	Digital I/O 6 / Request-to-send flow control
17	DIO3/AD3/SPI_SSEL	Both	Digital I/O 3 / Analog input 3 / SPI select
18	DIO2/AD2/SPI_CLK	Both	Digital I/O 2 / Analog input 2 / SPI clock
19	DIO1/AD1/SPI_ATTN	Both	Digital I/O 1 / Analog input 1 / SPI Attention
20	DIO0/AD0	Both	Digital I/O 0 / Analog input 0

# Troubleshooting

- Only use 3.3V, more than 5V will release magic smoke
- Use decoupling capacitors with a voltage regulator
- TX->RX
- RX->TX
- Don't overwhelm them, try putting in a small delay

# Firmware

- Must install with X-CTU (on Windows)
- API firmware
- Coordinator, Router, End Device
- Each Firmware has different settings

# Thank You

[https://www.robolab.in/zigbee-xbee-s2c-how-to-configure-  
as-coordinator-router-end-device/](https://www.robolab.in/zigbee-xbee-s2c-how-to-configure-as-coordinator-router-end-device/)

# Getting Started With XCTU

# Installation of XCTU

- Step 1: Click on the link :  
<https://www.digi.com/products/embedded-systems/digi-xbee-tools/xctu>
- Click on Download XCTU



The screenshot shows the XCTU software interface. At the top, there's a menu bar with 'XCTU', 'Working modes', 'Tools', and 'Help'. Below the menu is a toolbar with various icons. The main window is titled 'Radio Configuration' for a module named 'RT1-R009' (Zigbee Router API). The left panel lists 'Radio Modules' and 'Remote modules' with their names, functions, ports, MAC addresses, and icons. The right panel contains configuration settings for the selected module, including 'Networking' (Change networking settings), 'Joining' (Joining settings), 'Operational' (Operational settings), and 'Addressing' (Addressing settings). A watermark at the bottom right says 'Activate Windows' and 'Go to Settings to activate Win...'.

**XCTU**  
Next Generation Configuration Platform  
for XBee/RF Solutions

- XCTU is a **free, multi-platform** application compatible with Windows, MacOS and Linux
- **Graphical Network View** for simple wireless network configuration and architecture
- **API Frame Builder** is a simple development tool for quickly building XBee API frames
- **Firmware Release Notes Viewer** allows users to explore and read firmware release notes

DOWNLOAD XCTU

Activate Windows  
Go to Settings to activate Win...

HAVE A QUESTION?

# Installation of XCTU Cont..

Step 2: You can download XCTU software for windows x86/x64, Linux x64/x86 and mac os x. Click on **XCTUv6.4.3 windows x86/x64.**

## UTILITIES

### DOWNLOAD XCTU

- [XCTU v. 6.4.3 Windows x86/x64](#)
- [XCTU v. 6.4.3 MacOS X](#)
- [XCTU v. 6.4.3 Linux x64](#)
- [XCTU v. 6.4.3 Linux x86](#)
- [XCTU License Agreement](#)
- [XCTU v. 6.4.3 Release Notes](#)

### DOWNLOAD LEGACY XCTU

- [XCTU ver. 5.2.8.6 installer](#)  
Last old-gen version of XCTU: Contains features from previous versions, plus adds support for XBee Wi-Fi modules, Compatible with Windows 2000, XP, 2003, Vista, 7.  
Does not support the Digi XLR PRO.

- [XCTU 32-bit ver. 5.2.8.6 installer release notes](#)
- [XCTU ver. 5.1.0.0 installer](#)

This older version of X-CTU is required for XStream Ethernet RF modems, as well as XCite RF modules and modems. X-CTU 5.1.0.0 is compatible with Windows 2000, XP, 2003 only.

# Installation of USB driver for XBee Cont..

- Step 1: Click on link

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

- To download USB drivers → Clink on Download VCP(23MB)

## Download Software

The CP210x Manufacturing DLL and Runtime DLL have been updated and must be used with v6.0 and later of the CP210x Windows VCP Driver. Application Note Software downloads affected are AN144SW.zip, AN205SW.zip and AN223SW.zip. If you are using a 5.x driver and need support you can download archived Application Note Software.

[Legacy OS software and driver package download links and support information >](#)

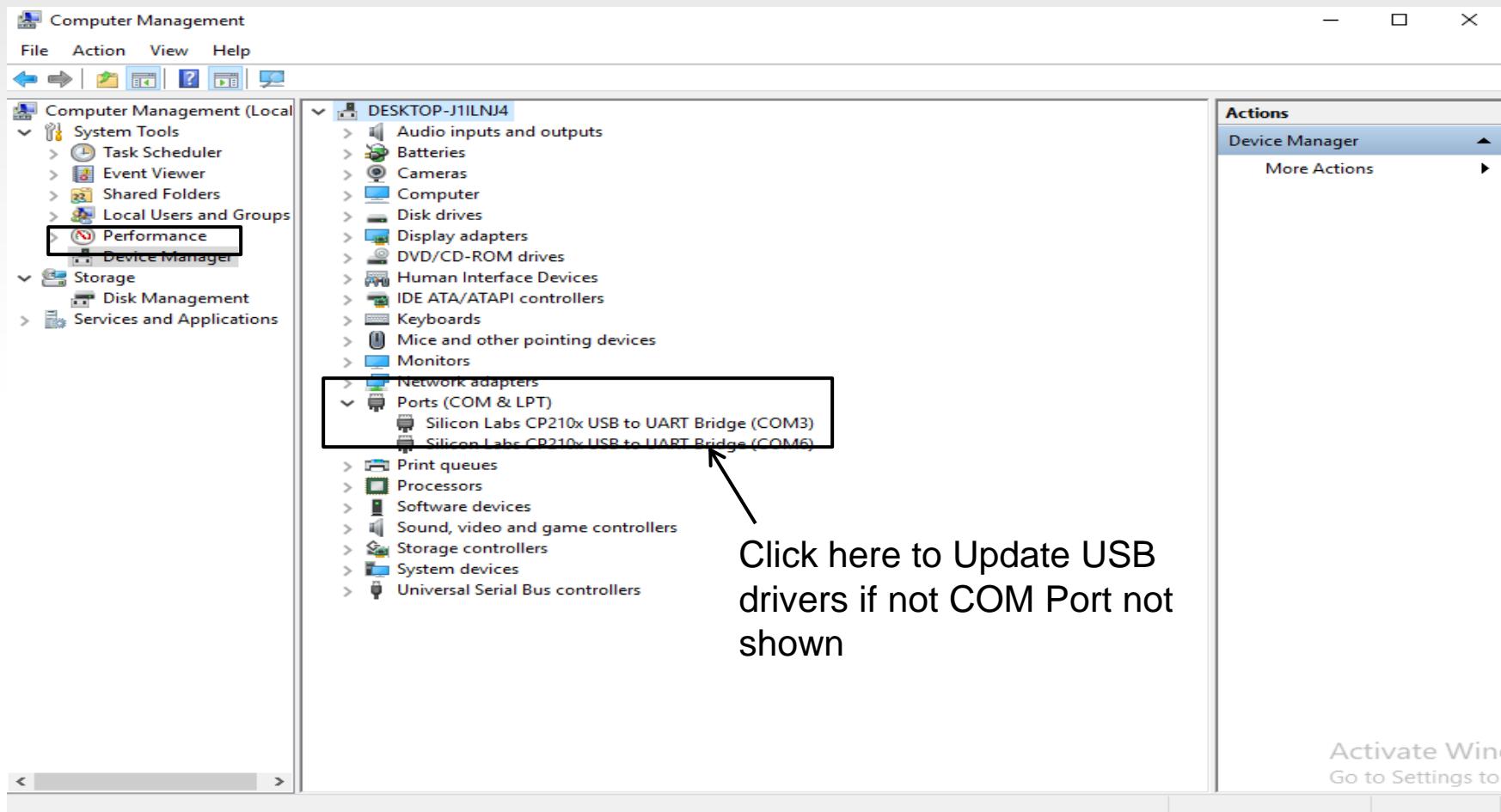
## Download for Windows 10 Universal (v10.1.8)

Note: The latest version of the Universal Driver can be automatically installed from Windows Update.

Platform	Software	Release Notes
 Windows 10 Universal	<a href="#">Download VCP (2.3 MB)</a>	<a href="#">Download VCP Revision History</a>

# Update USB driver for XBee

- Step 2: Check USB drivers for XBee. Go to Right click on My Computer--> open computer Management → click on Device Manager → and check here for USB driver for XBEE i.e. CP210x USB to UART Bridge VCP Drivers



# Communication Between Two XBee's

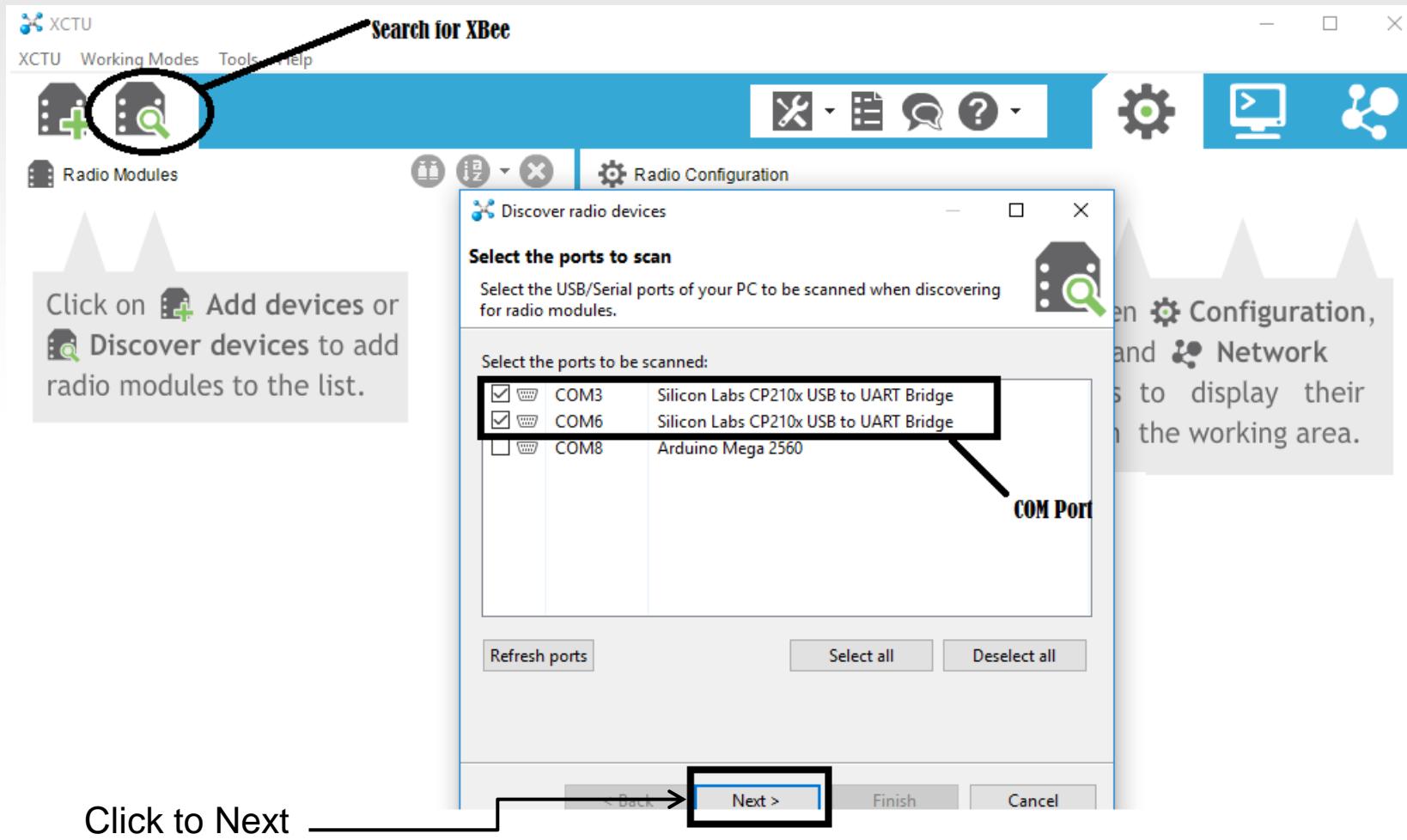
# Components Required

- Two XBee s2c with breakout board
- Two USB cables
- One or Two laptop/computer system

# Configuration for Coordinator XBee

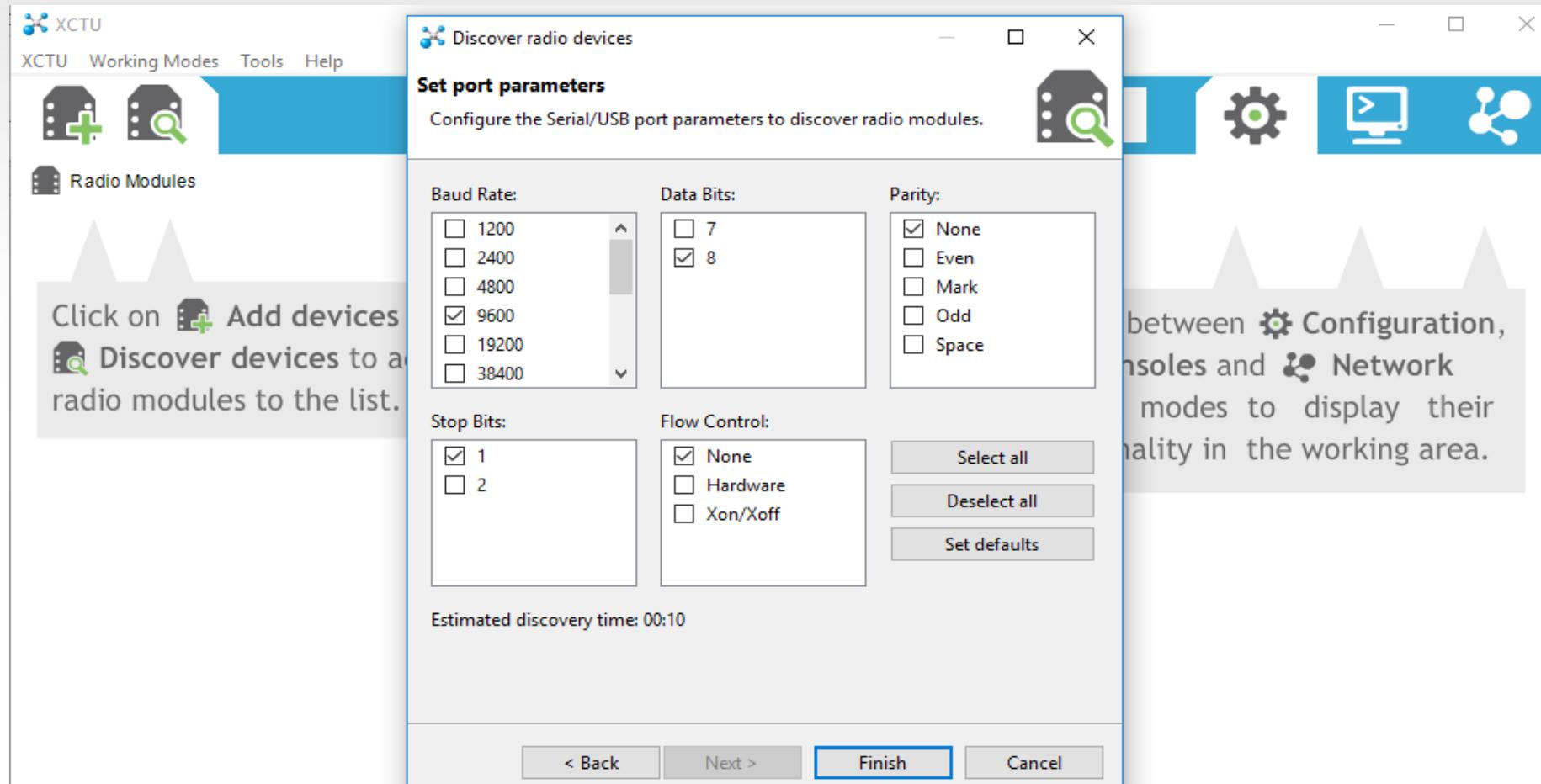
Steps:

1. Open the XCTU software and Click on the SEARCH icon on top to detect the USB ports.



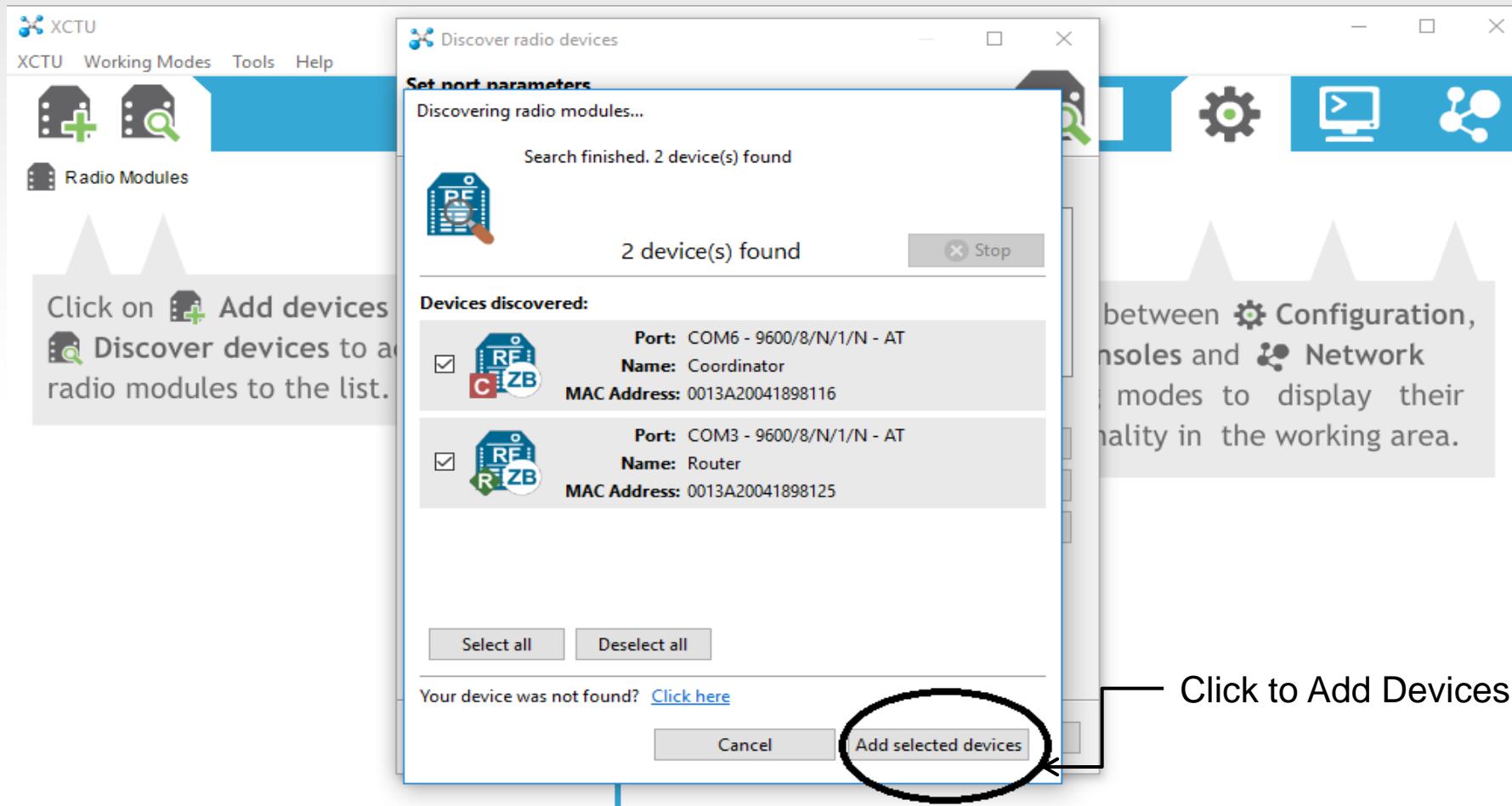
# Configuration for Coordinator XBee Cont..

- Click on NEXT & accept the default PORT PARAMETERS, 9600 is the BAUD RATE, 8 Data Bits, No Parity, Stop bit 1 and Flow Control None. Then Finish.



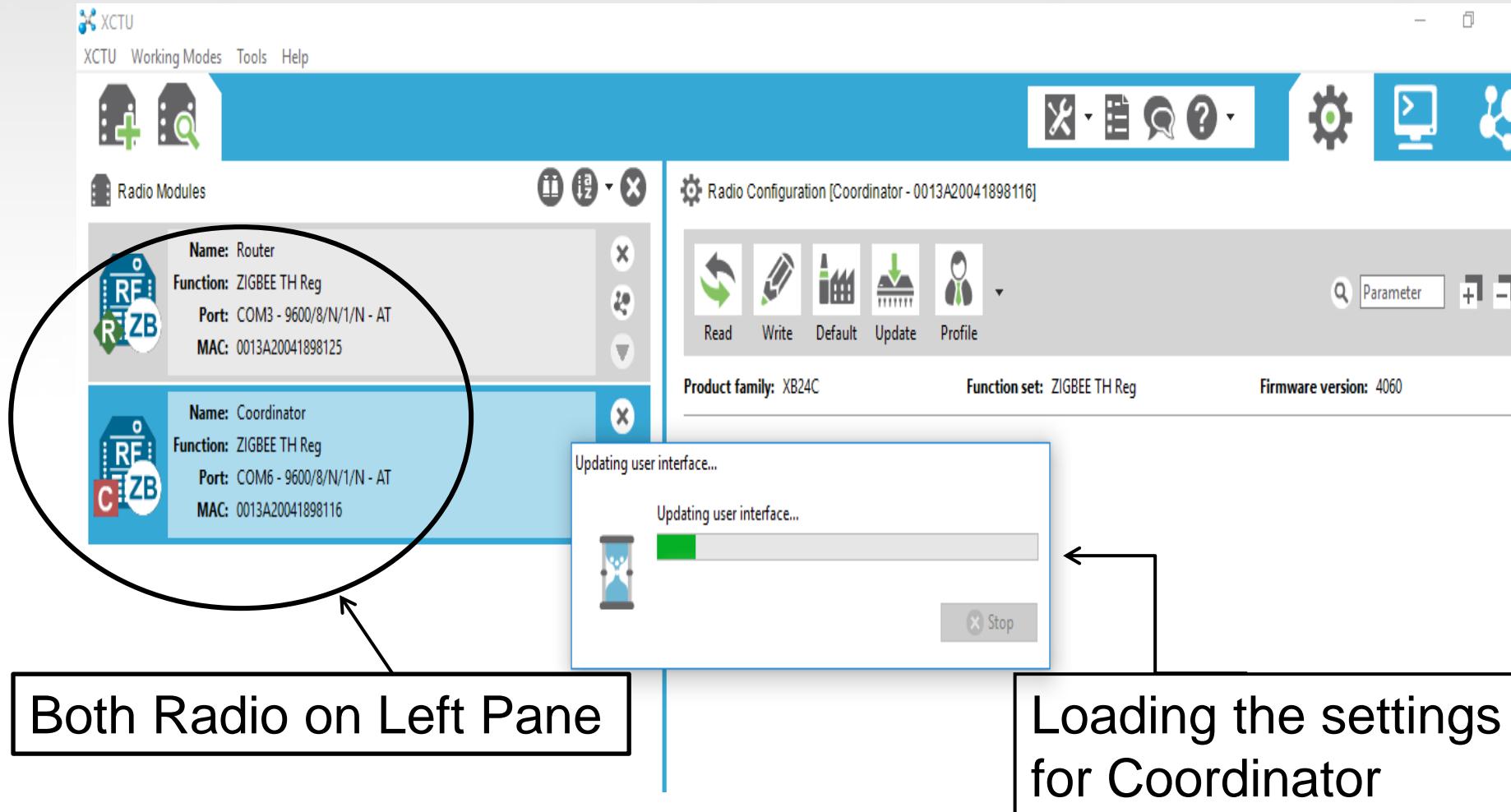
# Configuration for Coordinator XBee Cont..

3. The XCTU scans the USB ports selected & lists the RADIOS found with their unique 64-bit address and Select both the devices & click ADD SELECTED DEVICES.



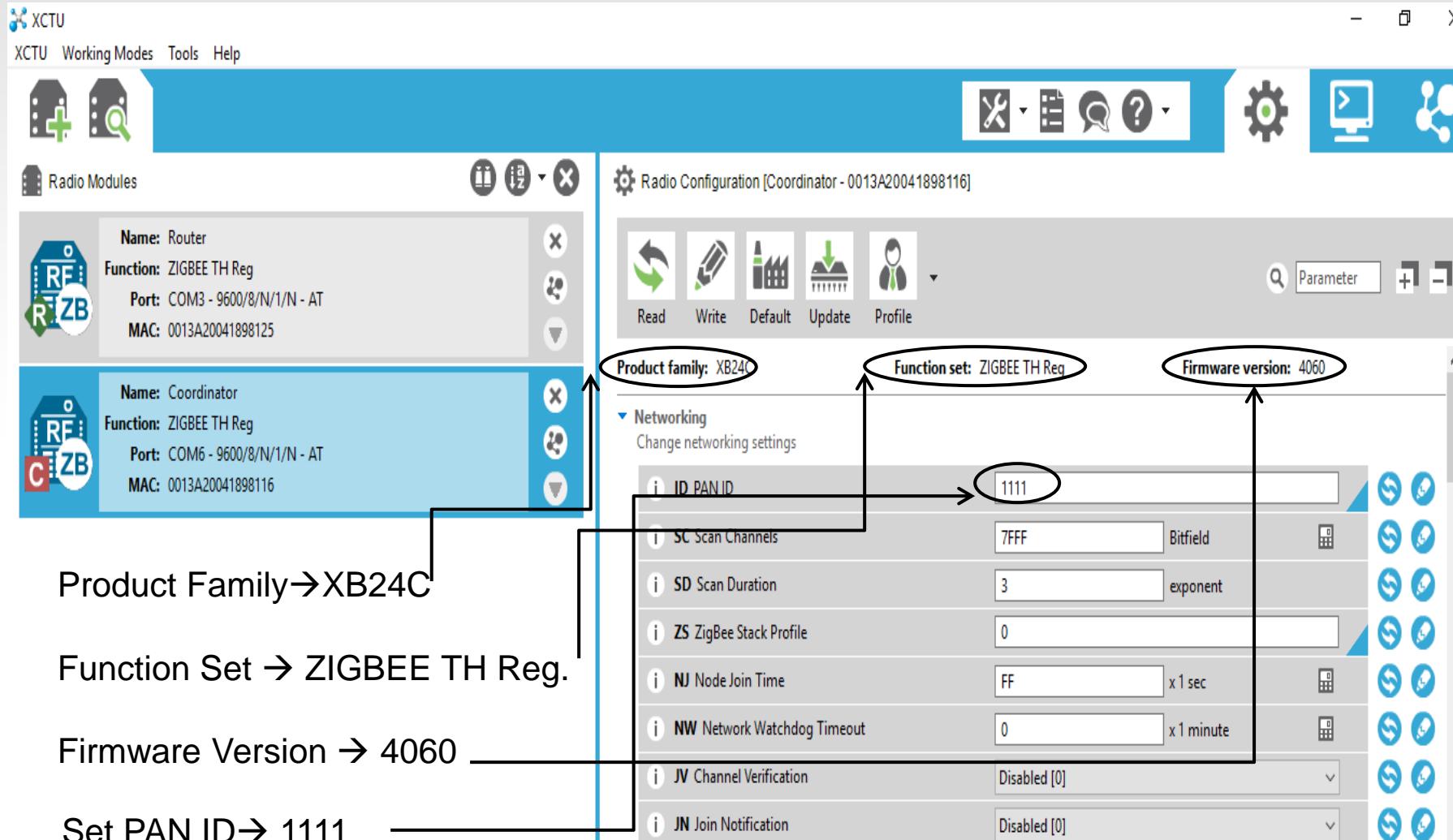
# Configuration for Coordinator XBee Cont..

4. Now both the Radios appear on the left pane. Let us configure the RADIO at COM6 as COORDINATOR first and Load the Settings by Clicking on it.



# Configuration for Coordinator XBee Cont..

5. 1<sup>st</sup> thing to set PAN ID of the network, this can be from 0 to FFFF Hex. In my case, it is 1111.



# Configuration for Coordinator XBee Cont..

The screenshot shows the XCTU (Zigbee Control Center) software interface for configuring a Coordinator XBee. The left pane displays two radio modules: a Router (Name: Router, Function: ZIGBEE TH Reg, Port: COM3 - 9600/8/N/1/N - AT, MAC: 0013A20041898125) and a Coordinator (Name: Coordinator, Function: ZIGBEE TH Reg, Port: COM6 - 9600/8/N/1/N - AT, MAC: 0013A20041898116). The right pane shows the 'Radio Configuration' screen for the Coordinator.

Annotations on the left side map configuration settings to their corresponding fields in the XCTU interface:

- CE → Enables[1]**: Points to the 'CE Coordinator Enable' dropdown, which is set to **Enabled [1]**.
- DH → 0**: Points to the 'DH Destination Address High' field, which contains **0**.
- DL → FFFF**: Points to the 'DI Destination Address Low' field, which contains **FFFF**.
- NI → Coordinator**: Points to the 'NI Node Identifier' field, which contains **Coordinator**.

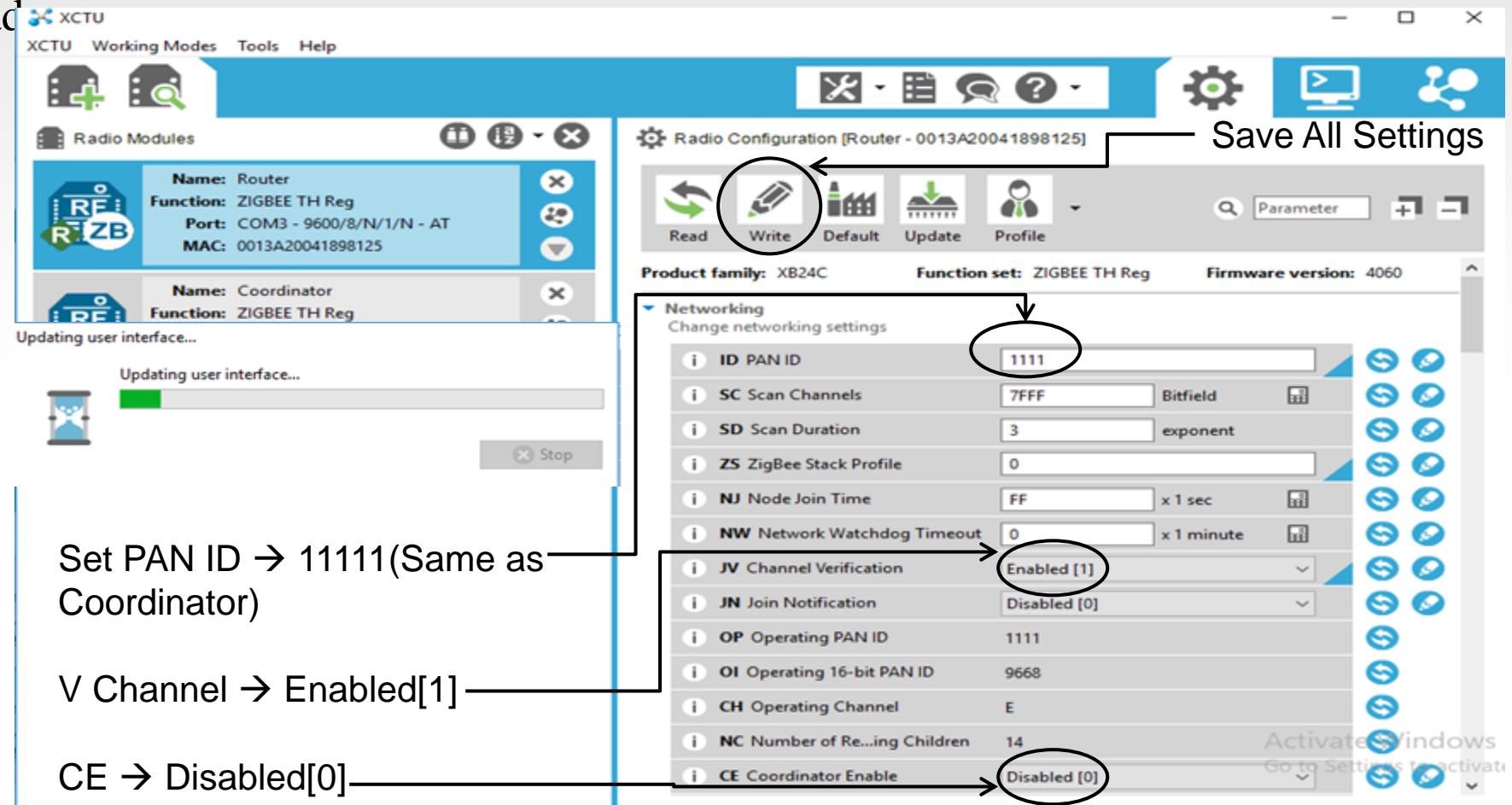
Annotations on the right side highlight specific fields in the configuration screen:

- A circled arrow points to the **Write** button in the toolbar.
- A circled arrow points to the **Enabled [1]** dropdown under the 'CE Coordinator Enable' section.
- A circled arrow points to the **0** value in the 'DH Destination Address High' field.
- A circled arrow points to the **FFFF** value in the 'DI Destination Address Low' field.
- A circled arrow points to the **Coordinator** value in the 'NI Node Identifier' field.

The top right of the configuration screen has a **Save All Settings** button.

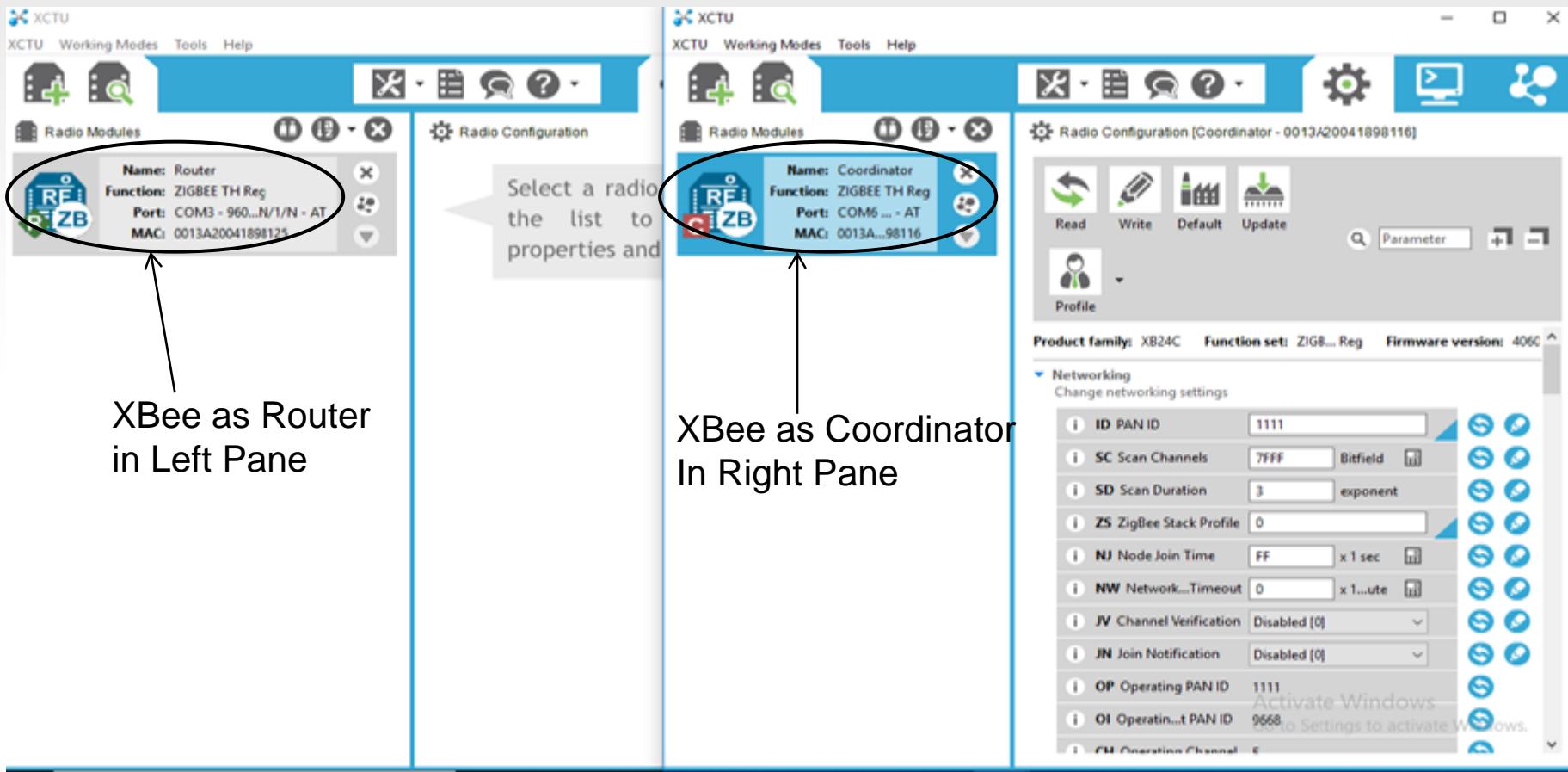
# Configuration for Router XBee

Click on the 2<sup>nd</sup> radio on the left pane to load the setting and then Enter the PANID as 1111 , same as that of Coordinator. JV CHANNEL VERIFICATION is Enabled, CE Coordinator is DISABLED, Destination Address DL is left to default 0, and NI (Node Identifier) as “ ROUTER”. Now Click on WRITE button to save the changes made.



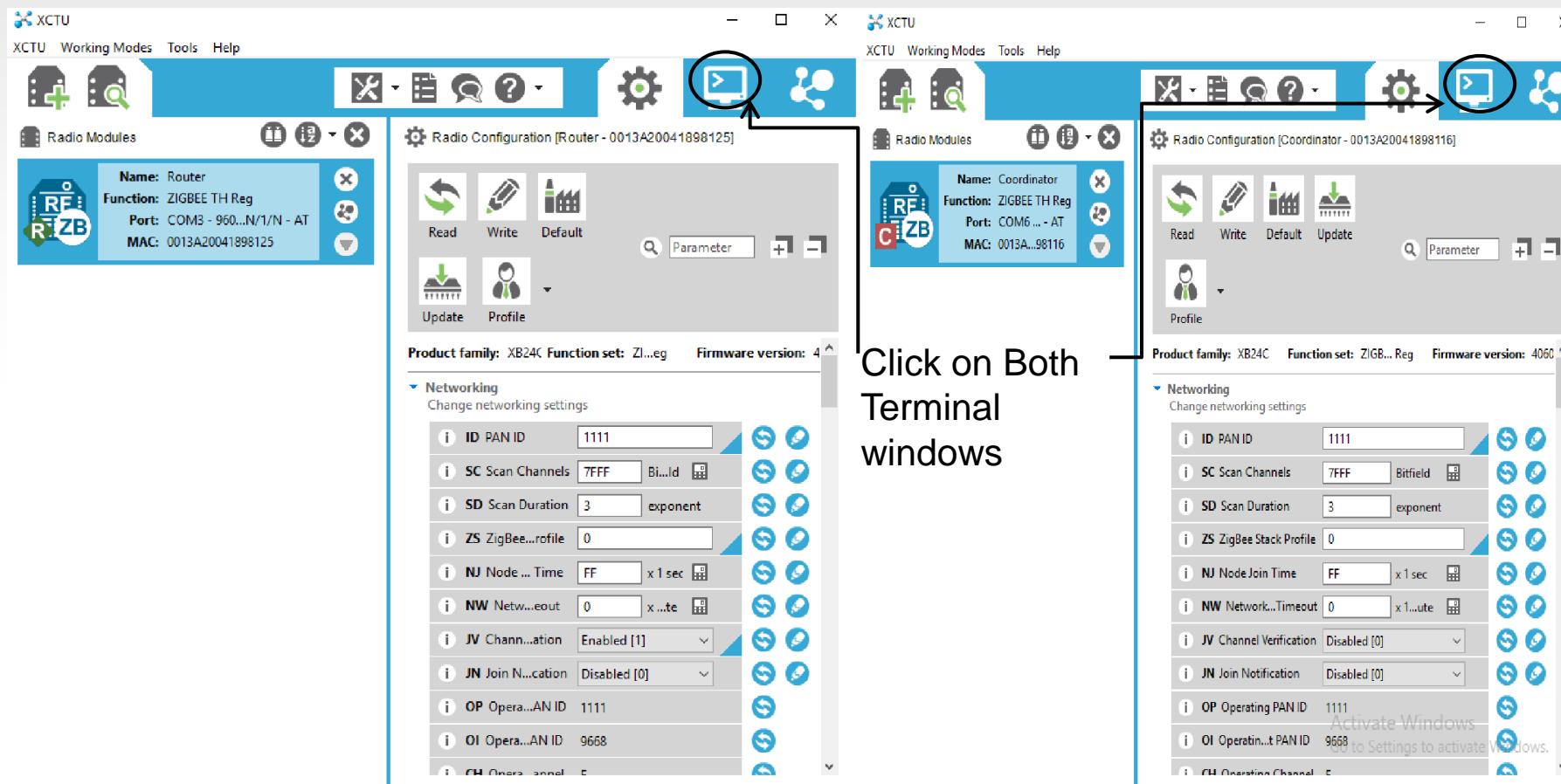
# Communication Between Coordinator and Router

1. The modules are paired and ready for communication. Now let us test the communication on the XCTU window delete the second Radio. Click on the first Radio to load the settings.



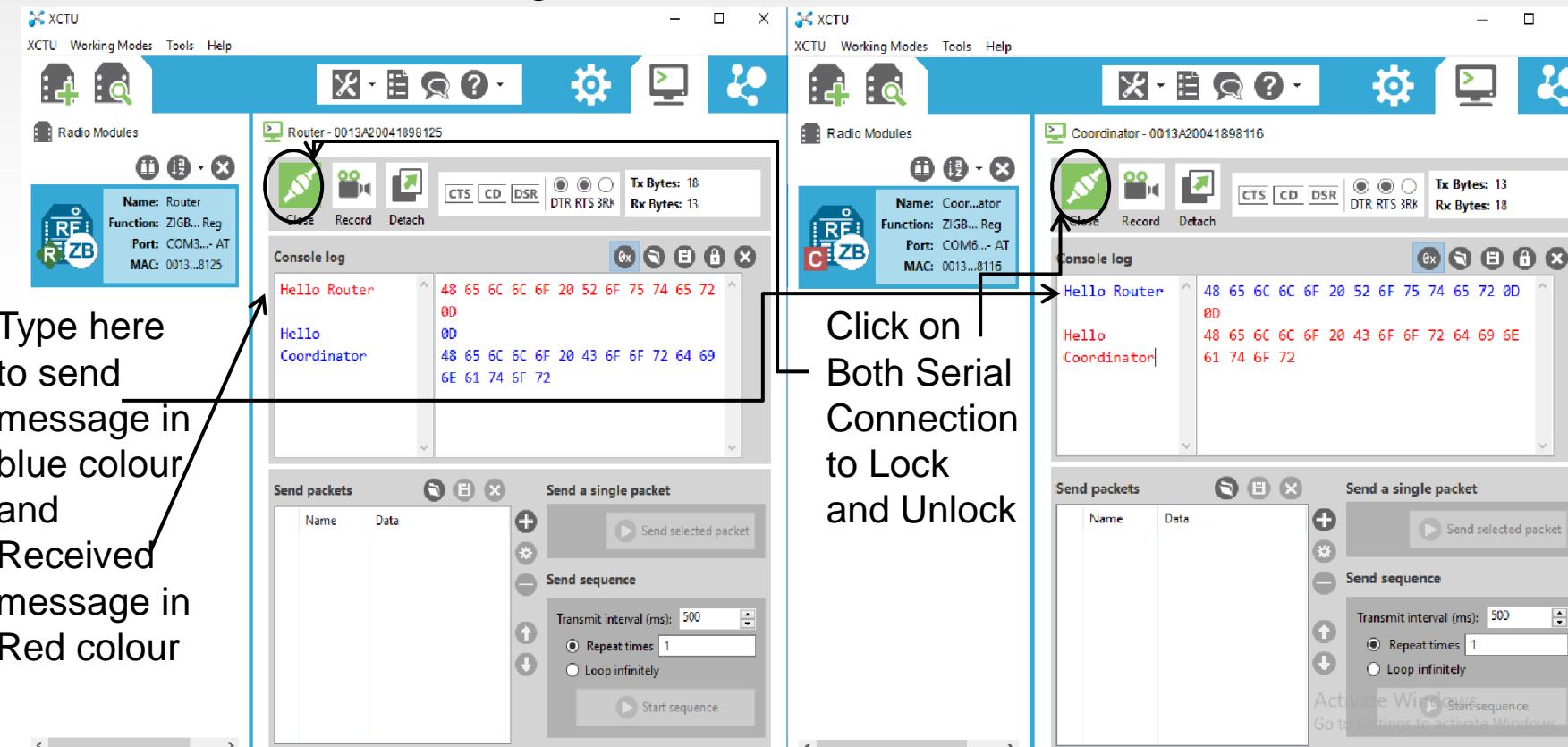
# Communication Between Coordinator and Router

2. Click the TERMINAL icon on both the windows to enter Terminal mode. Click on the SERIAL CONNECTION icon on both the windows to enter the serial connection mode.



# Communication Between Coordinator and Router

3. You can see the SERIAL Icon in LOCK mode & the AT CONSOLE Status changes to CONNECTED. Now you can type any message inside console log window & see that received on the other Radio. The transmit message is in BLUE & received message in RED.



# Communication Between XBee and Ultrasonic Sensor Using Arduino

# Component Required

- Two XBee s2c with breakout board
- Two USB cables
- One Arduino Mega-2560 board
- Jumper wires
- One/Two laptops/computer systems
- One Ultrasonic sensor

# Installation of Arduino IDE

- Must installed XCTU software for configuration of XBee.
- Must installed Arduino IDE for configuration of Arduino mega- 2560 board and also programmed it.

<https://www.arduino.cc/en/main/software>

The screenshot shows the Arduino website's download page. At the top, there is a navigation bar with links for HOME, STORE, SOFTWARE (which is highlighted in blue), EDU, RESOURCES, COMMUNITY, and HELP. There is also a search icon, a shopping cart icon, and a SIGN IN link. Below the navigation bar, the text "Download the Arduino IDE" is displayed. To the left, there is a large teal circle containing the Arduino logo (a white infinity symbol with a minus sign on the left and a plus sign on the right). To the right of the logo, the text "ARDUINO 1.8.9" is shown in bold, followed by a description of the software. Further down, there are download links for Windows, Mac OS X, and Linux, along with links for getting started and release notes.

HOME STORE SOFTWARE EDU RESOURCES COMMUNITY HELP

Download the Arduino IDE

**ARDUINO 1.8.9**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board. Refer to the [Getting Started](#) page for installation instructions.

**Windows** Installer, for Windows XP and up  
**Windows** ZIP file for non admin install

**Windows app** Requires Win 8.1 or 10  
[Get](#)

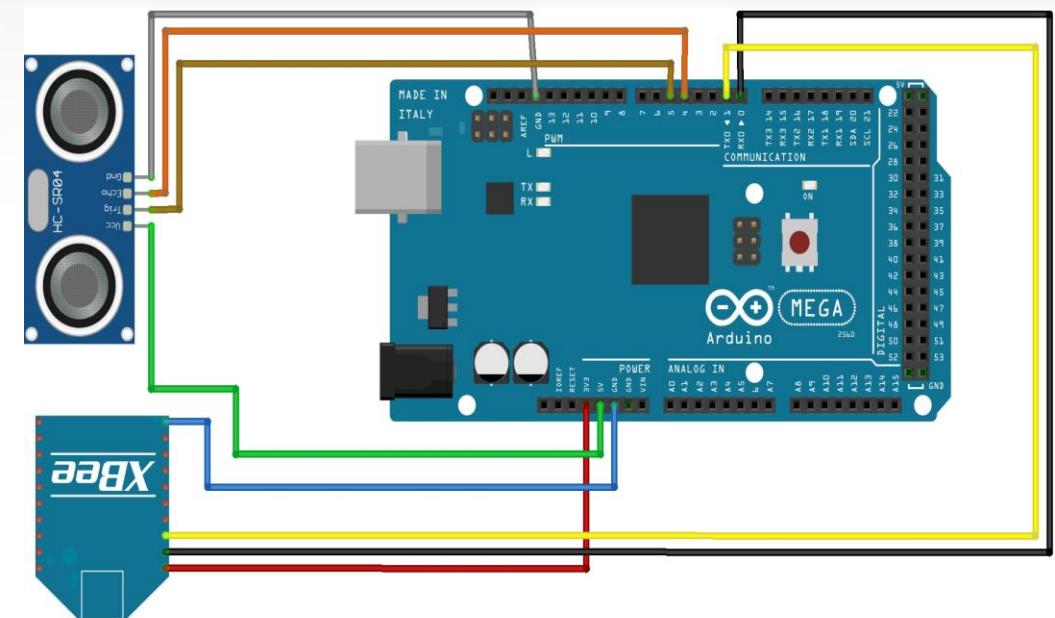
**Mac OS X** 10.8 Mountain Lion or newer

**Linux** 32 bits  
**Linux** 64 bits  
**Linux** ARM 32 bits  
**Linux** ARM 64 bits

[Release Notes](#)  
[Source Code](#)  
[Checksums \(sha512\)](#)

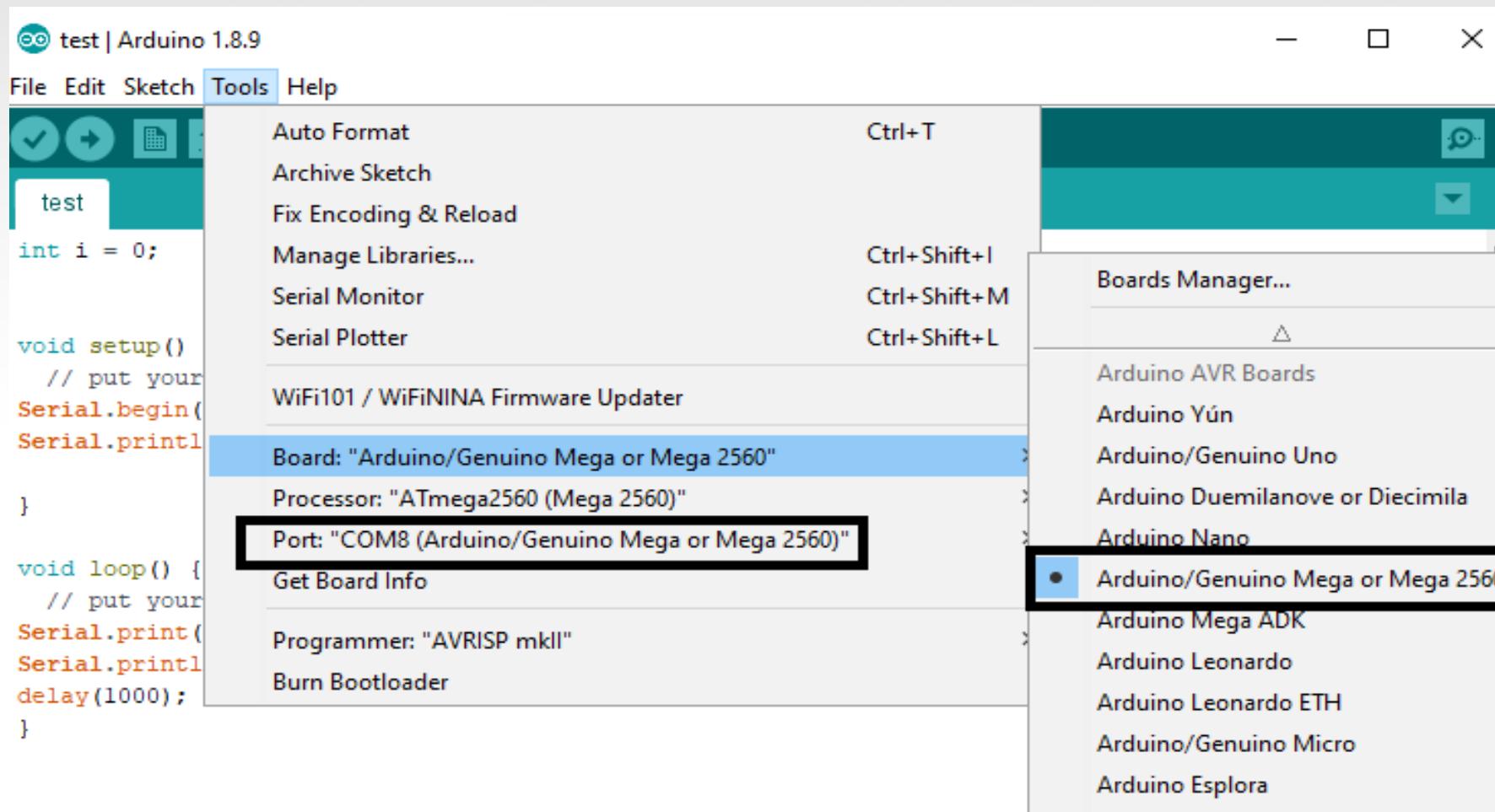
# Connections Between XBee, Arduino Mega and Ultrasonic Sensor

- After installation XTCU software and Arduino ide
- Now make the connections between XBee, Arduino mega and ultrasonic sensor.
- 3vcc of XBee to 3vcc of Arduino mega
- Pin 2-TX , Pin 3-RX of XBee to RX0-0,  
TX0-1 of Arduino Mega
- GND- XBee to GND -AM
- 5VCC of Ultrasonic sensor to 5vcc-AM
- GND- US to GND-AM
- TRIG and Echo to Pin 5 and 4 resp.



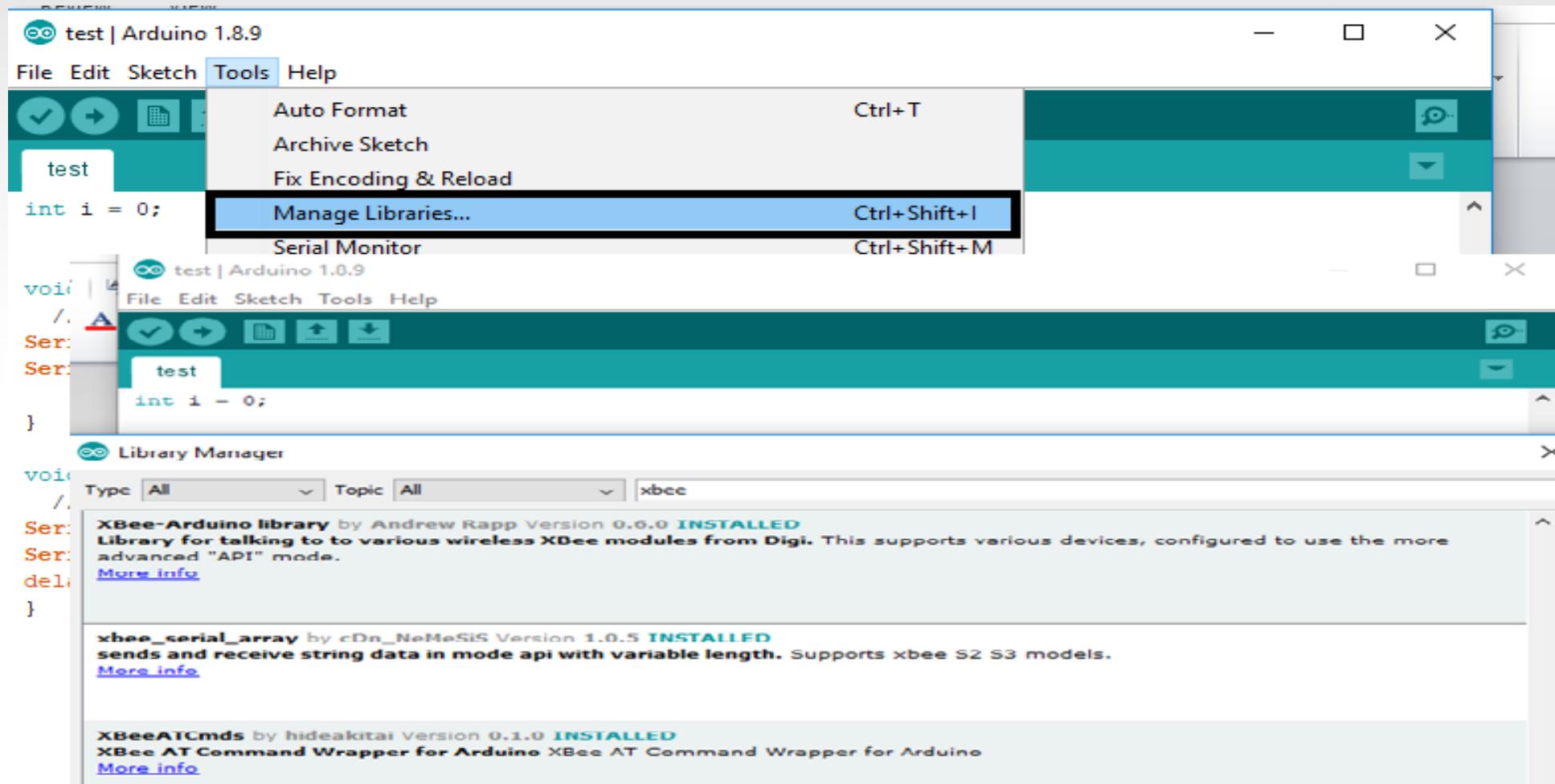
# Configuration on Arduino IDE

- Click on tool → board → Arduino Mega 2560 and port → COM8



# Manage Libraries on Arduino IDE

- Then install XBee library by click on tool → manage library → search XBee library → XBee – Arduino library by Andrew Rapp, xbee\_serial\_array by cDn\_NeMesis and XBeeATCmds by hideakitai .



# Code for Communication with Ultrasonic sensor



The screenshot shows the Arduino IDE interface with a sketch named "sensor". The code is written in C++ and performs the following tasks:

- Defines constants for the trigPin (5) and echoPin (4).
- Declares variables for duration and distance.
- In the setup() function:
  - Configures trigPin as an output and echoPin as an input.
  - Initializes the Serial port at 9600 bps.
- In the loop() function:
  - Sets trigPin LOW for 2 microseconds.
  - Sets trigPin HIGH for 10 microseconds.
  - Sets trigPin LOW again.
  - Reads the duration of the pulse on echoPin.

```
sensor | Arduino 1.8.9
File Edit Sketch Tools Help
sensor
const int trigPin = 5;
const int echoPin = 4;

float duration, distance;

void setup() {
    // put your setup code here, to run once:

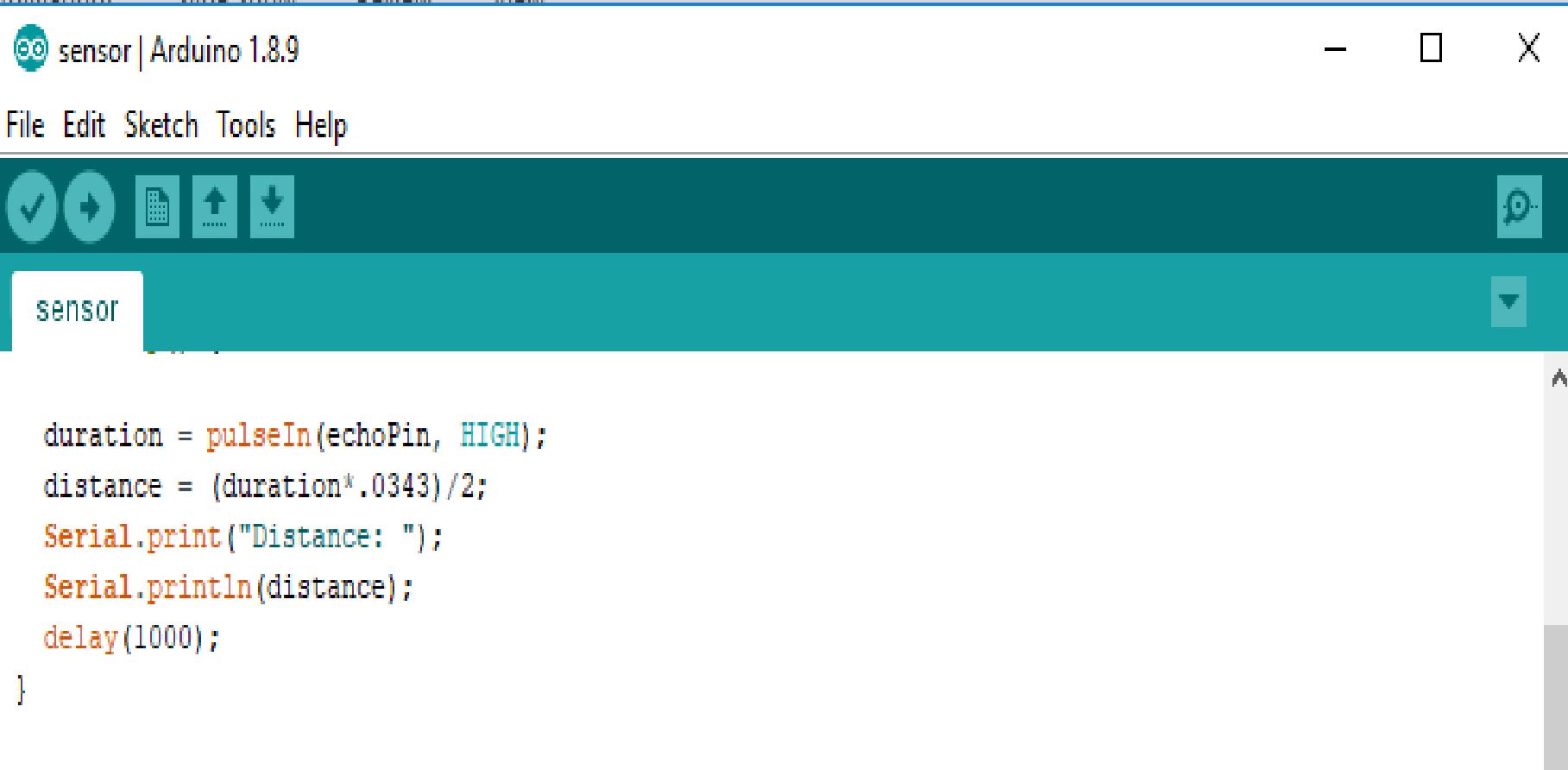
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    Serial.begin(9600);
}

void loop() {
    // put your main code here, to run repeatedly:

    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    duration = pulseIn(echoPin, HIGH);
```

# Code for Communication with Ultrasonic Sensor Cont..



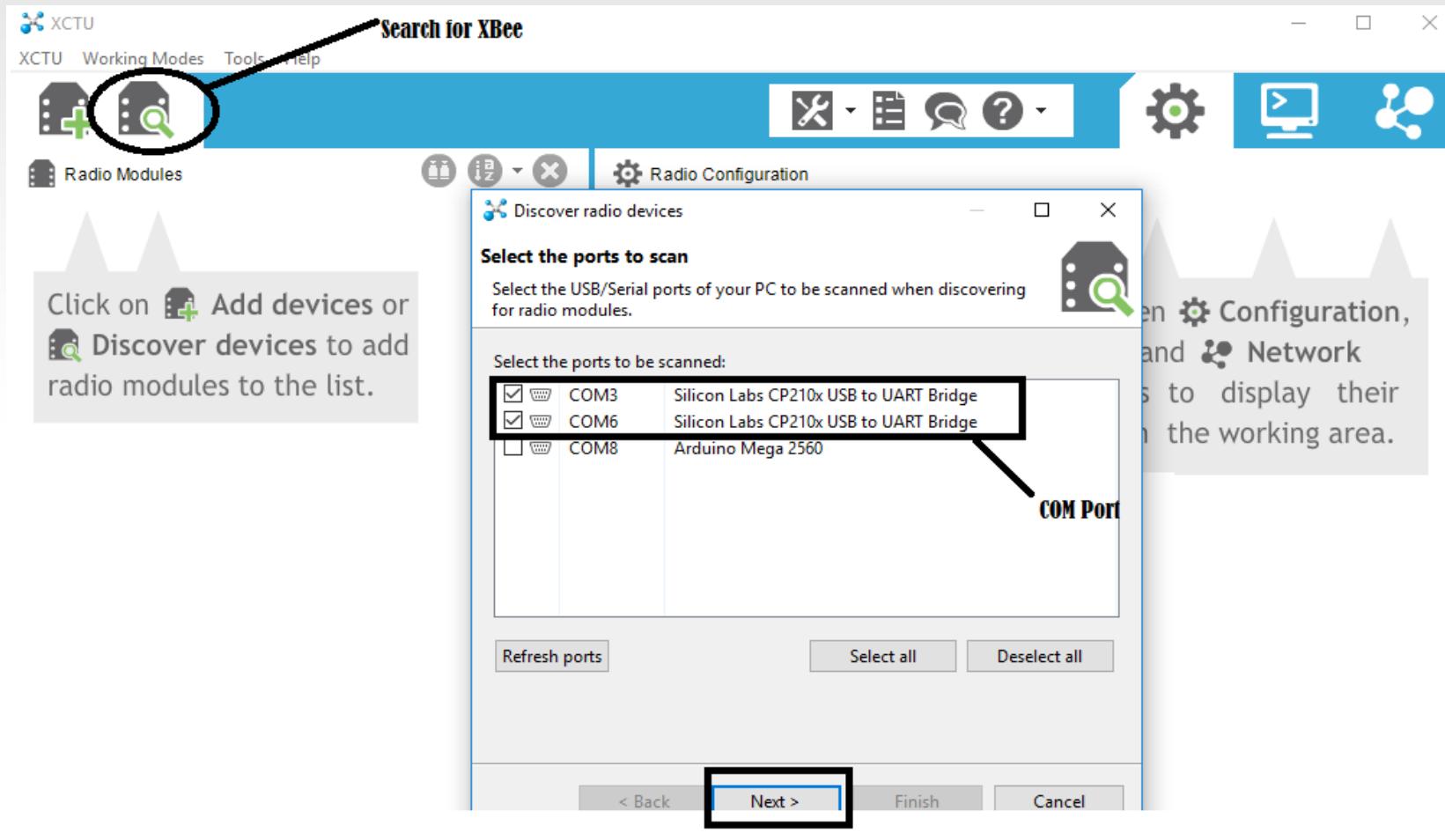
The image shows a screenshot of the Arduino IDE interface. The title bar reads "sensor | Arduino 1.8.9". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu is a toolbar with icons for upload, refresh, and other functions. The main workspace contains the following code:

```
duration = pulseIn(echoPin, HIGH);
distance = (duration*.0343)/2;
Serial.print("Distance: ");
Serial.println(distance);
delay(1000);
}
```

# Configuration of XBee using XCTU

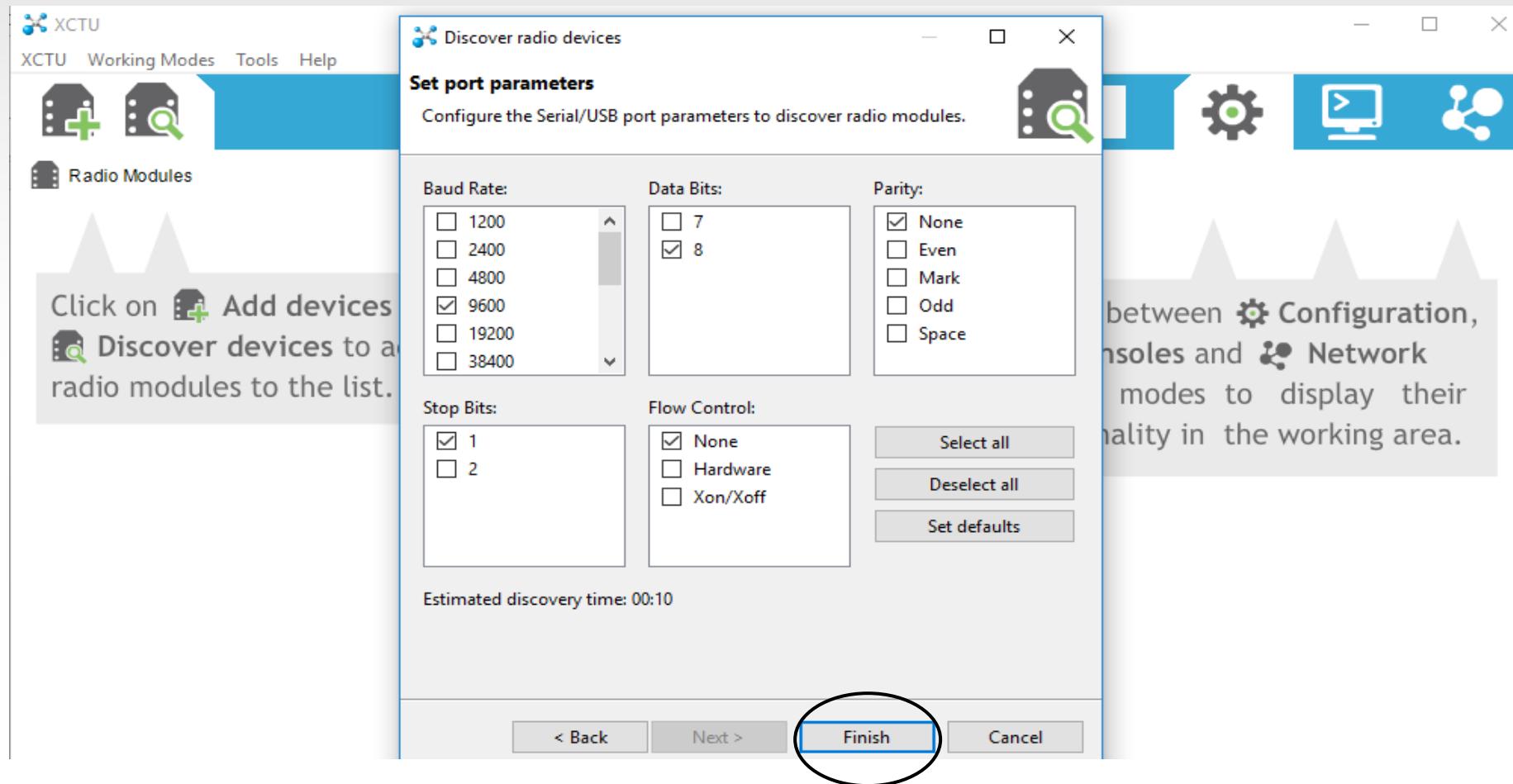
Steps:

1. Open the XCTU software and Click on the SEARCH icon on top to detect the USB ports.



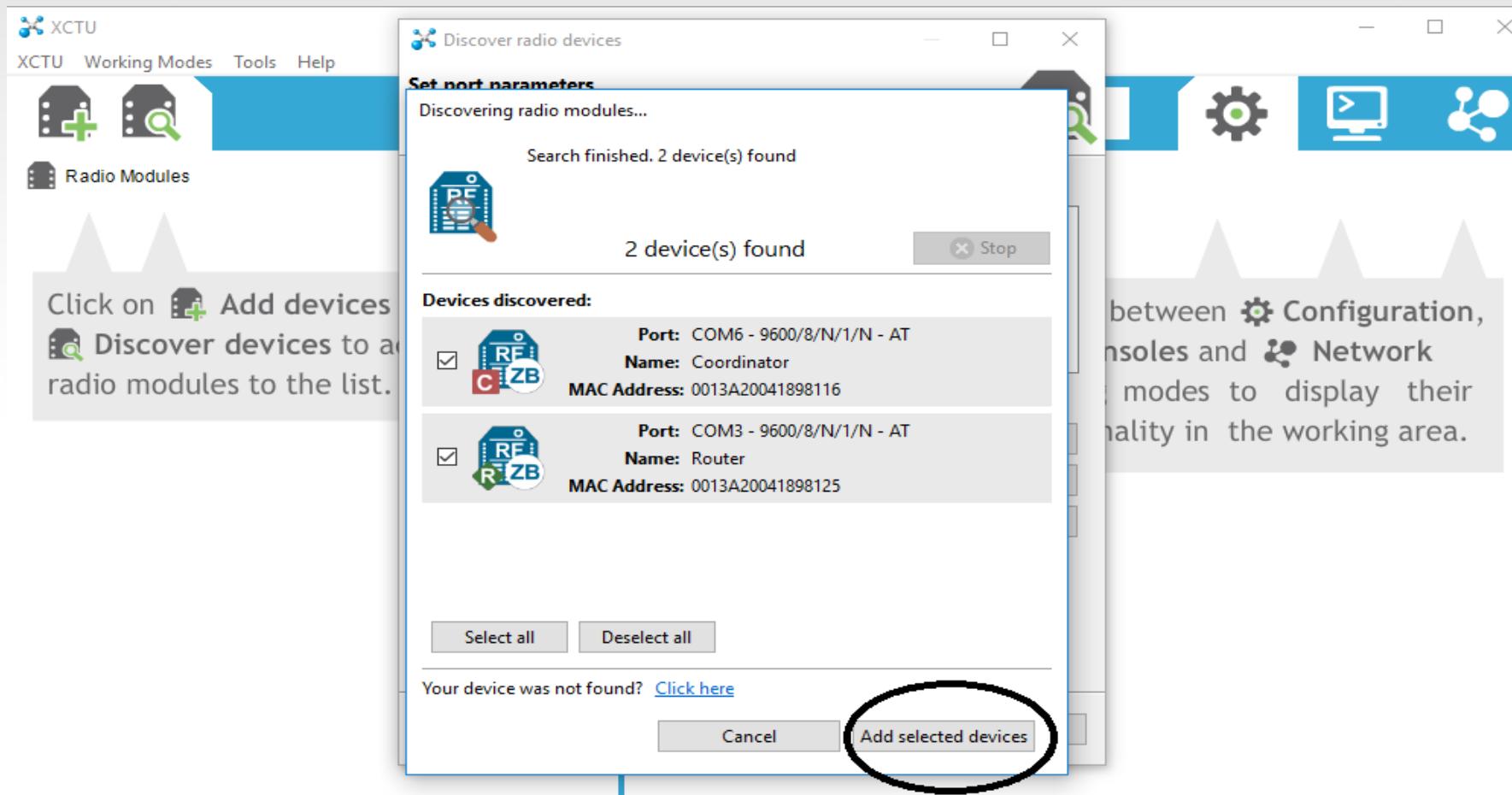
# Configuration of XBee using XCTU Cont..

2. Click on NEXT & accept the default PORT PARAMETERS, 9600 is the BAUD RATE, 8 Data Bits, No Parity, Stop bit 1 and Flow Control None and then Finish.



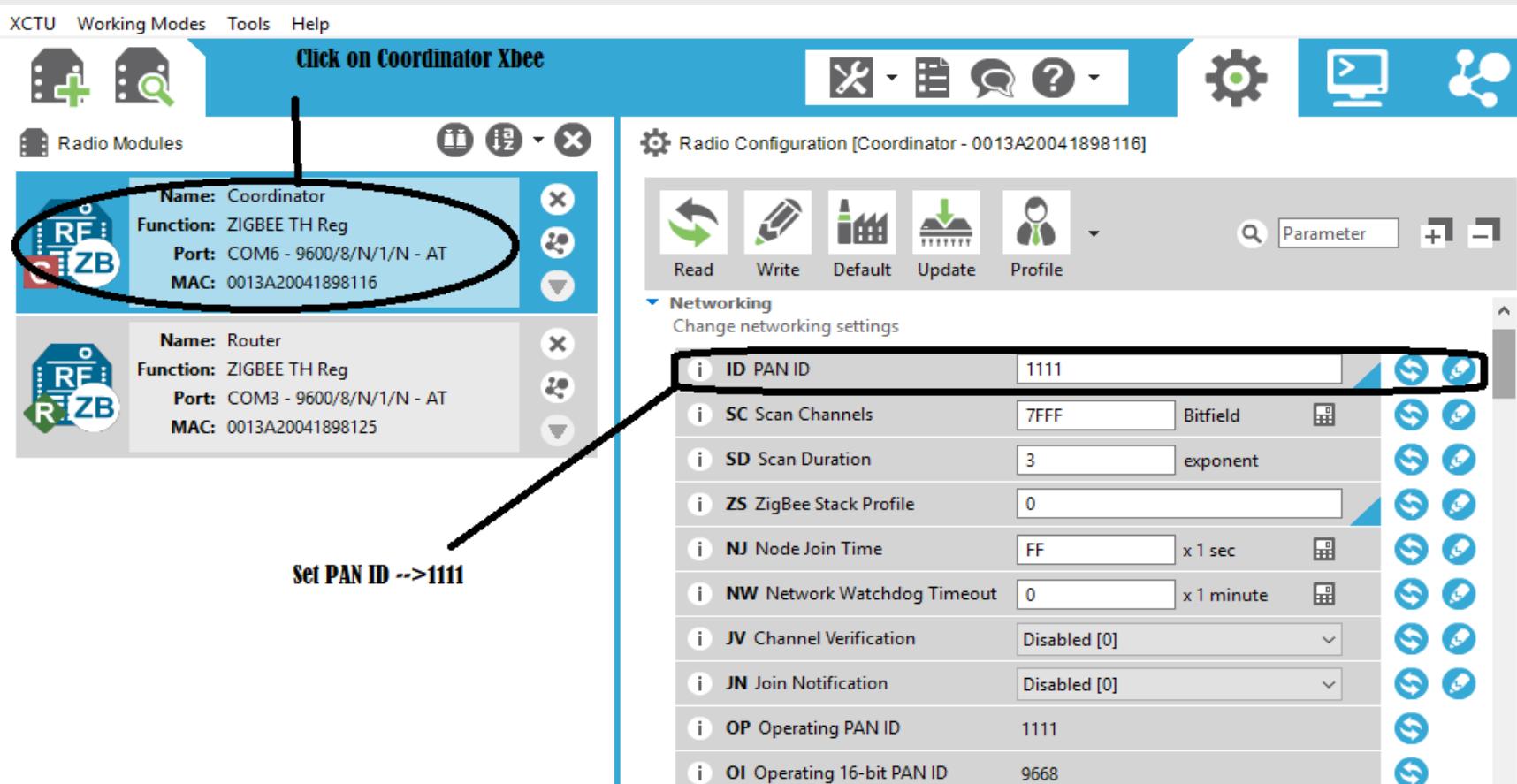
# Configuration of XBee using XCTU Cont..

3. The XCTU scans the USB ports selected & lists the RADIOS found with their unique 64-bit address and Select both the devices & click ADD SELECTED DEVICES.



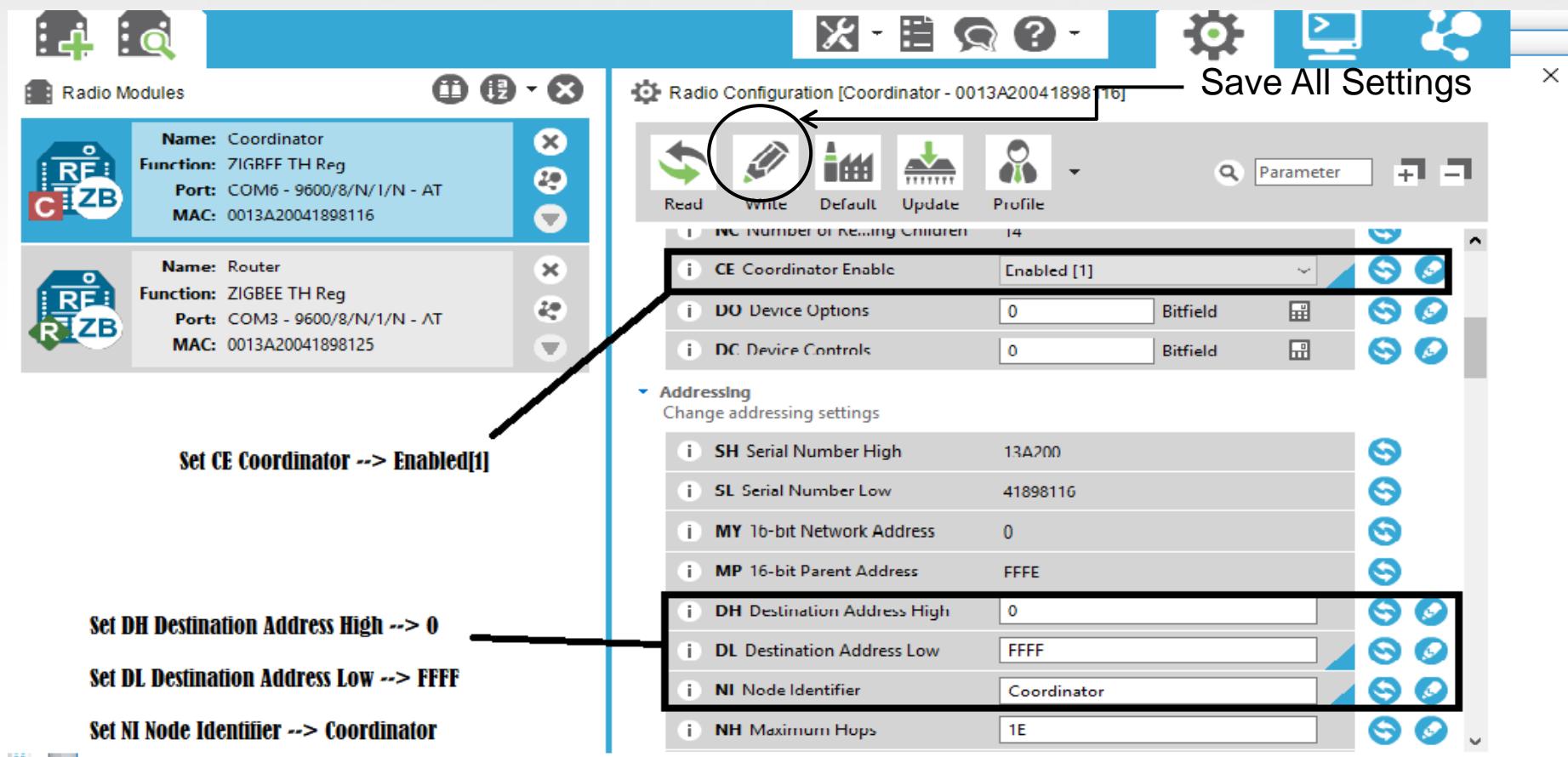
# Configuration of Coordinator XBee using XCTU

4. Now both the Radios appear on the left pane. Let us configure the RADIO at COM6 as COORDINATOR first and Load the Settings by Clicking on it and then set the PAN ID → 1111



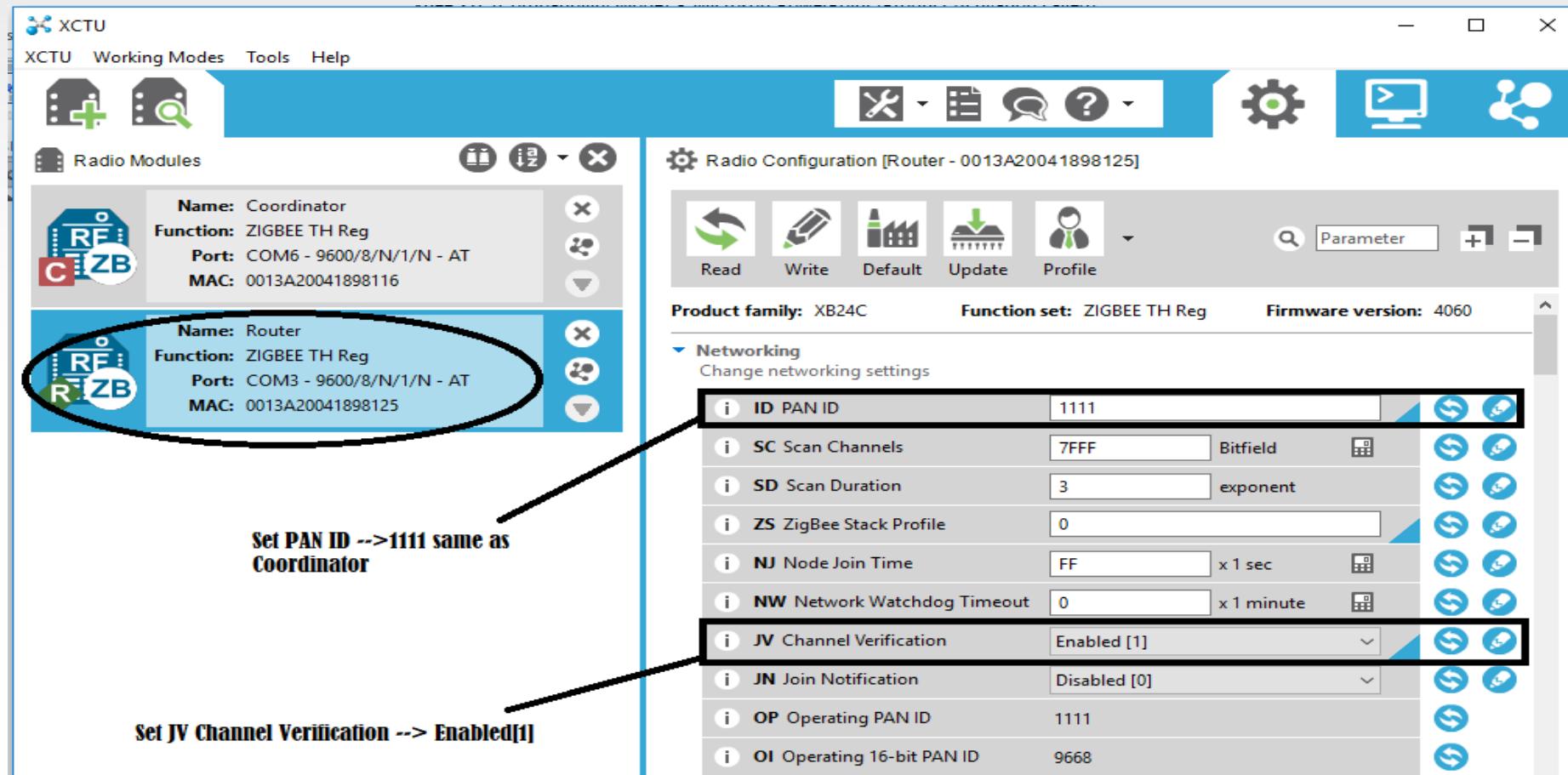
# Configuration of Coordinator XBee using XCTU Cont..

5. Scroll Down and set CE → Enabled[1], Set DH, DL →0,FFFF and Set NI → Coordinator and then Click on the PENCIL icon on top to WRITE the changes made. This is done with Coordinator configuration.



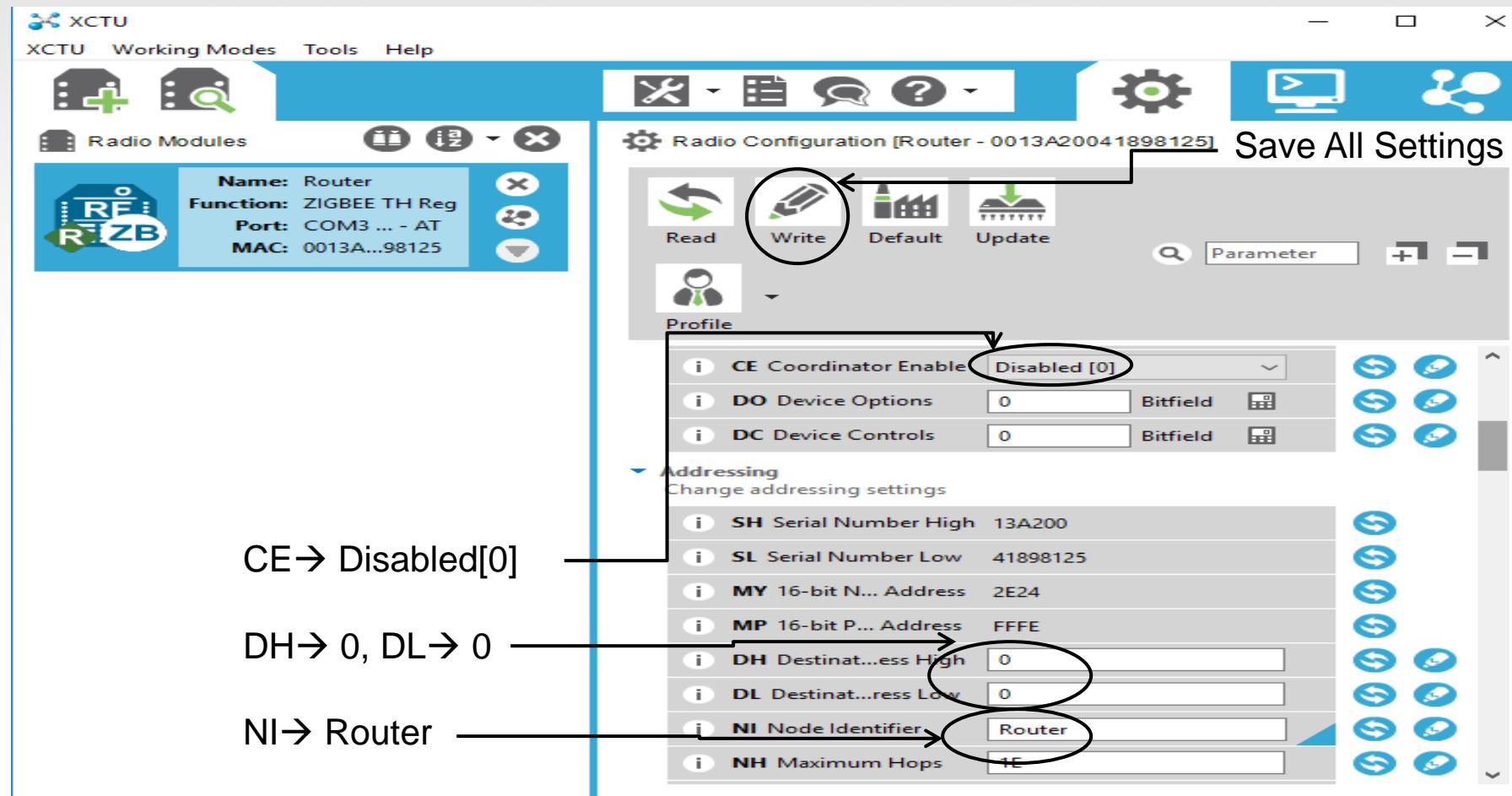
# Configuration for Router XBee

1. Click on the 2<sup>nd</sup> radio on the left pane to load the setting and then Enter the PAN ID as 1111 , same as that of Coordinator. JV CHANNEL VERIFICATION is Enabled and rest settings are same as default and then Click on the PENCIL icon on top to WRITE the changes made. This is done with Router configuration.



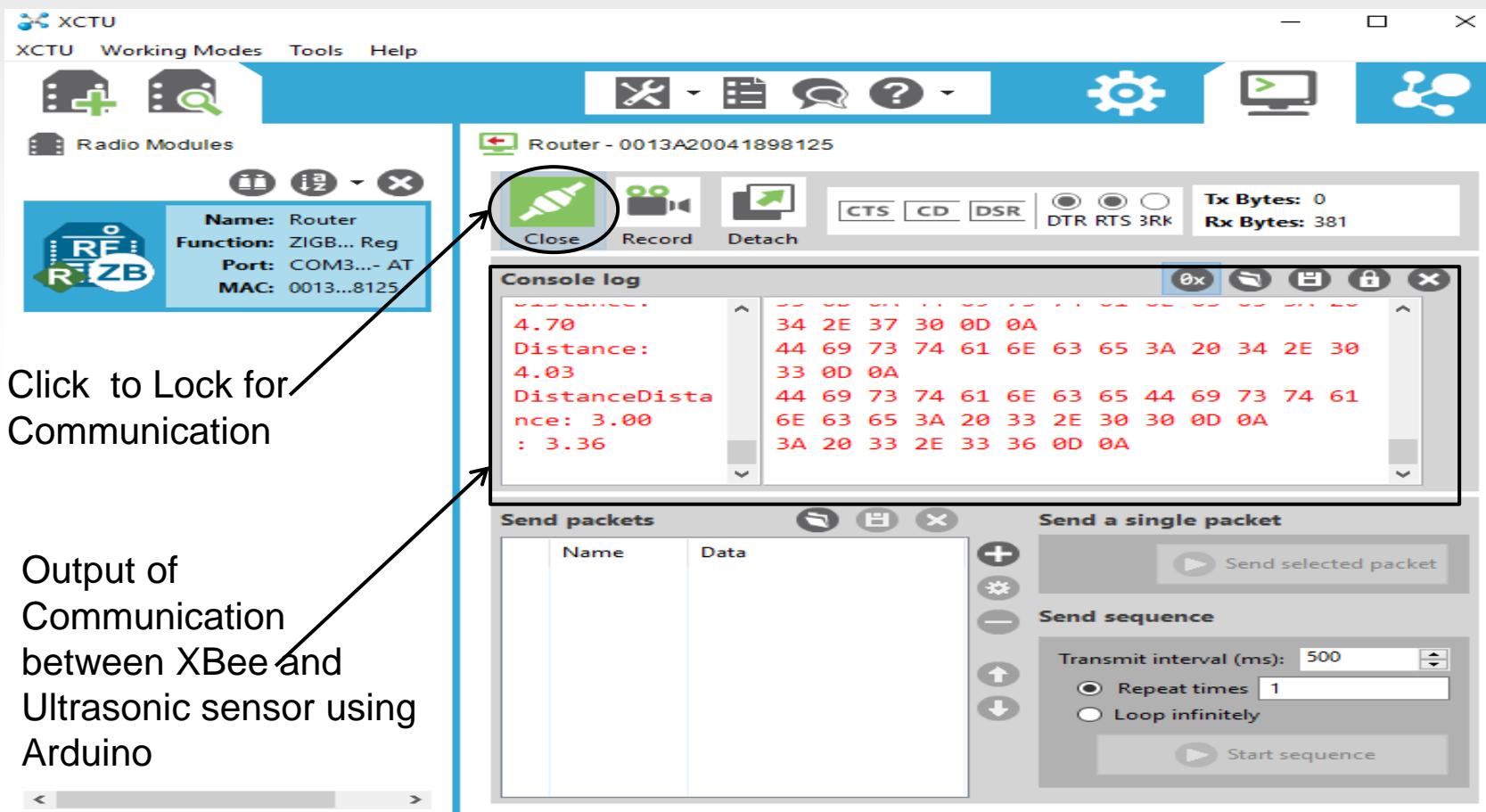
# Configuration for Router XBee Cont..

2. CE Coordinator is DISABLED, Destination Address DL is left to default 0. and NI Node Identifier as “ROUTER”.  
Now Click on WRITE button to save the changes made.



# Communication Between XBee and Ultrasonic Sensor using Arduino

3. The modules are paired and ready for communication. Now let us test the communication on the XCTU Router Window as:



# Thank You

# Software Defined Networking



# TABLE

## OF

## Contents

1. SDN Introduction.
2. SDN integrated IOT
3. Attacks in SDN.
4. DDOS Attack detection using ML.
5. ARP Poisoning attack detection in SDN.
6. Blockchain integrated SDN.
7. References.

# Traditional Network

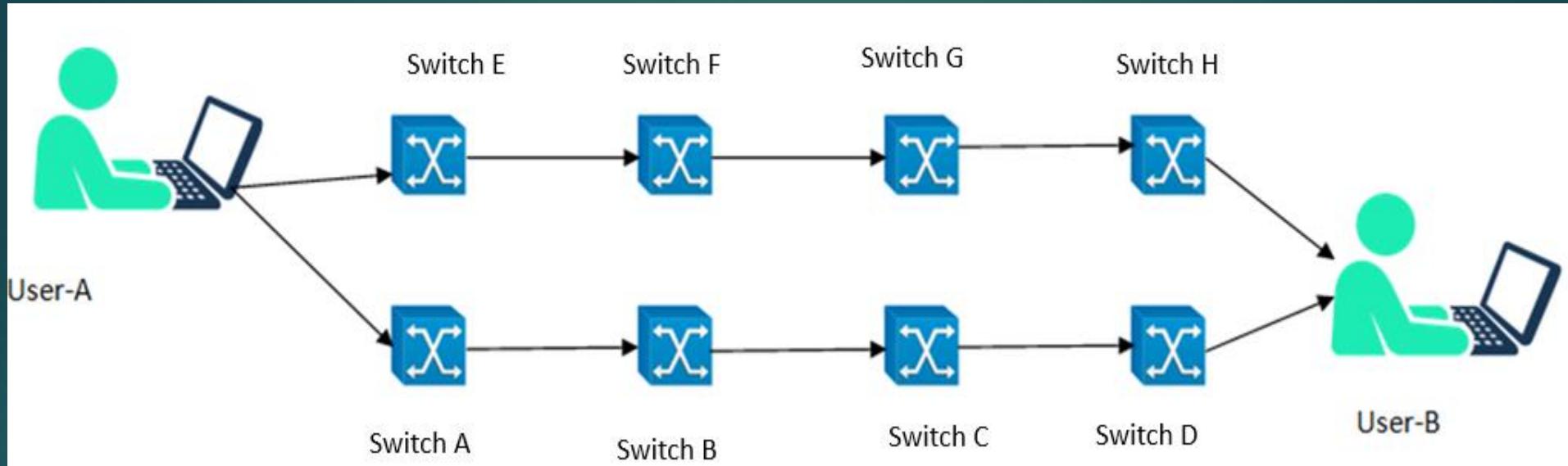


Fig : Traditional Network Architecture

- Traditional Network[1] is based on destination address forwarding.
- Routing is done based on the protocol configured in the switch.(RIP, OSPF, IGRP, EIGRP etc.)
- Switch here are layer 3 switch in which OSPF protocol is executing.
- Every switch forwards packet based on the routing table[4] it posses. Switches does not have global view of the network.

# Traditional Network in attack Scenario

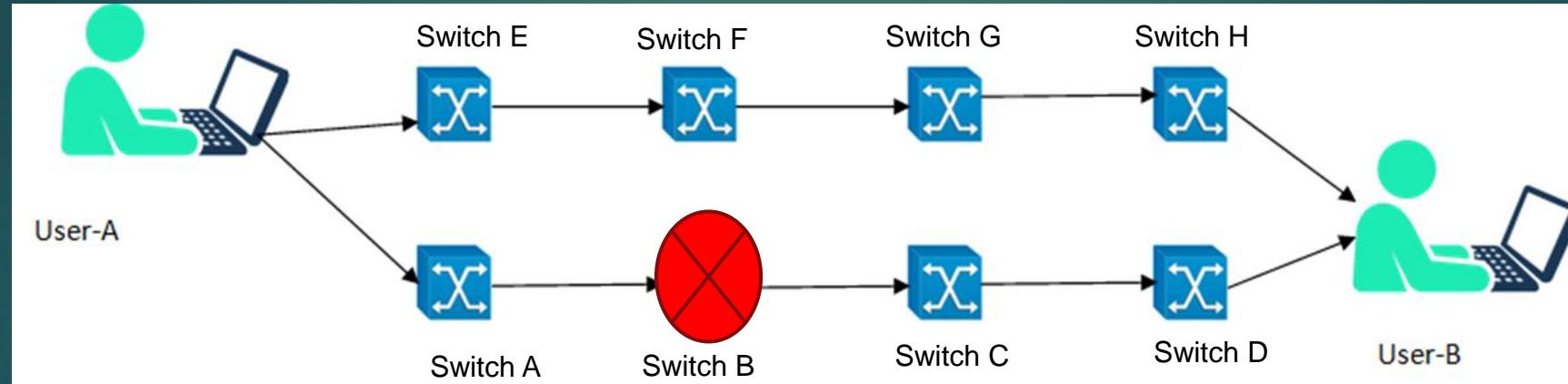


Fig : Attack Scenario in traditional architecture

- Switch C got attacked.
- Need to found other route through which the traffic can be routed.
- But there is no centralized entity which can take care of the rerouting in present network.
- Depending on the protocol the rerouting will occur.

# Software-Define Networking (SDN)



Host A



Controller

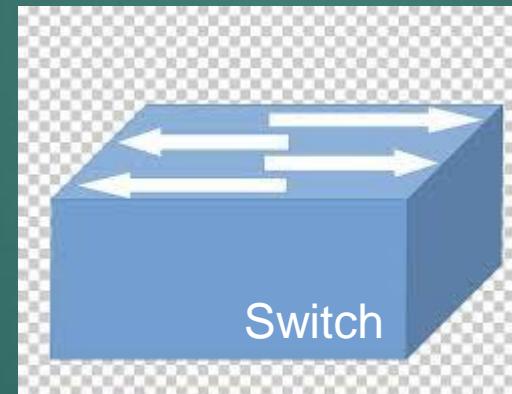


Fig : SDN Concept



Host B

- When the switch has no entry in the flow table, the switch contacts the controller.
- Controller add the route to the destination in the switch's flow table [3].
- By using the flow entry switch will send the subsequent requests.

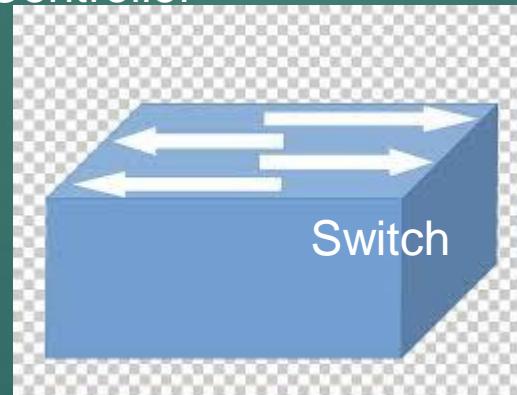
# ‘ SDN (Continued.....)



Host A



Controller



Host B

Fig : SDN Concept

- Once the switch's flow table [1] has entry of a route.
- It use that entry and send the packet to the destination host.

# SDN Architecture

- Separation of Control & Forwarding Plane
- Programmability & Automation
- Network Device Control Based on Policies.

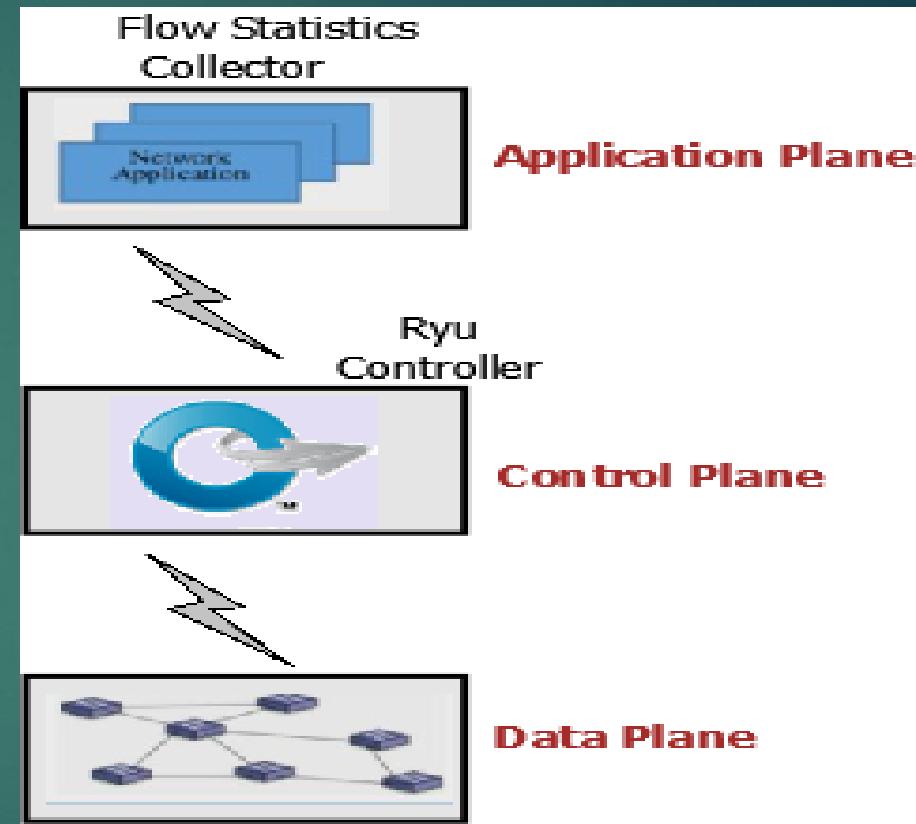


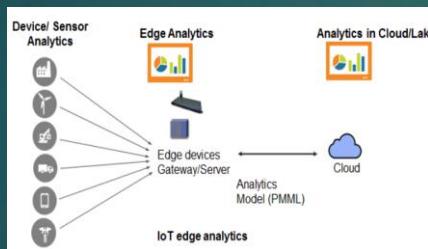
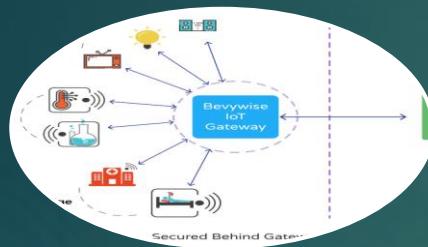
Fig : SDN Architecture

# Traditional Network vs SDN

Table-1: Traditional network VS Software defined networking

Category	Traditional Networking	SDN
Type of Network	Human intervened	Programmable
Communication Overhead	No	Yes
Vendor Dependent Switches	Yes	No
Error Prone Configuration	Yes	No
Expensive	Yes	No
Network Status Availability	No	Yes

# Simplified IOT Architecture



Data collection from sensors . Actuators act upon the data collected and provide some output.

Sensor data aggregation and analog to digital data conversion.

Edge IT System performs enhanced analytics and preprocessing on the data.

Analysis, Management and storage of data.

# Challenges of IOT

Real Time  
Programming of  
sensor nodes.

Vendor specific  
architecture of  
sensor nodes.

Resource  
Constrained  
Nodes.

Limited Memory  
& Distributed  
Control of  
Sensor Nodes.

# Opportunities in IOT

- Programming the sensor nodes in real time.
- Changing the forwarding path in real time.
- Integrating different sensor nodes in WSN.
- SDN can be used to manage and control IOT network.
- Wireless sensor nodes and network can be controlled using SDN based applications.
- Network performance can be improved significantly using SDN based approaches over traditional approaches.

# SDN Integrated with IOT

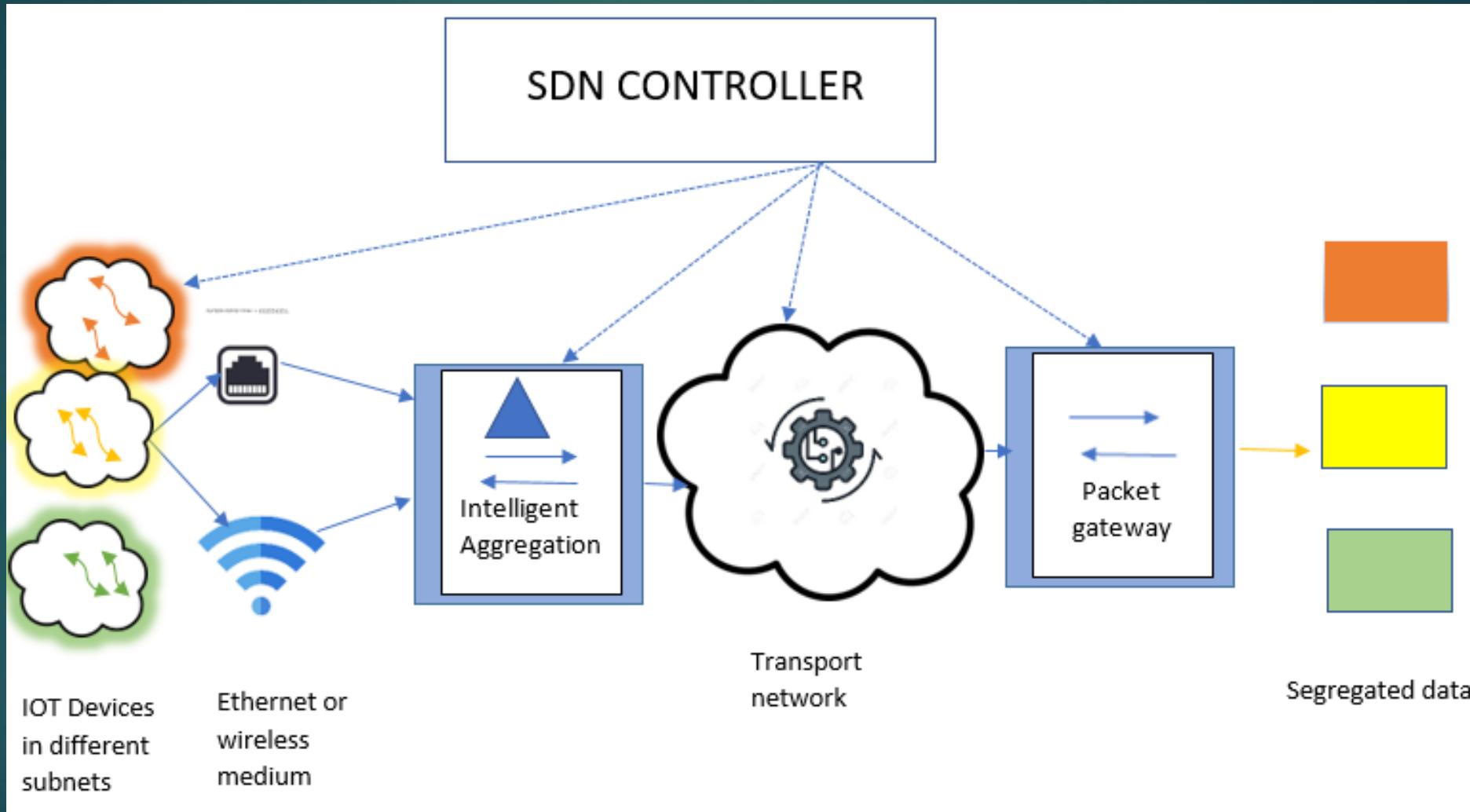


Fig: SDN integrated IOT architecture.

# Benefits of integrating SDN to IOT

Intelligent  
routing  
decision

Simplify  
Information  
Collection

Decision  
making made  
simple.

Intelligent  
traffic pattern  
analysis

# Soft-WSN: Software-Defined WSN management system for IoT

- ▶ Component Based Approach:
  - ▶ Device management
    - ▶ Sensor Management : Sensors can be used depending on application.
    - ▶ Delay Management : Delay for sensing can be set dynamically.
    - ▶ Active Sleep Management:
  - ▶ Topology Management
    - ▶ Node Specific management : Forwarding Logic of a sensor can be modified.
  - ▶ Network Specific Management
    - ▶ Forward all the traffic of a node.
    - ▶ Drop all the traffic of a node.

# Software-Defined Sensor network Architecture

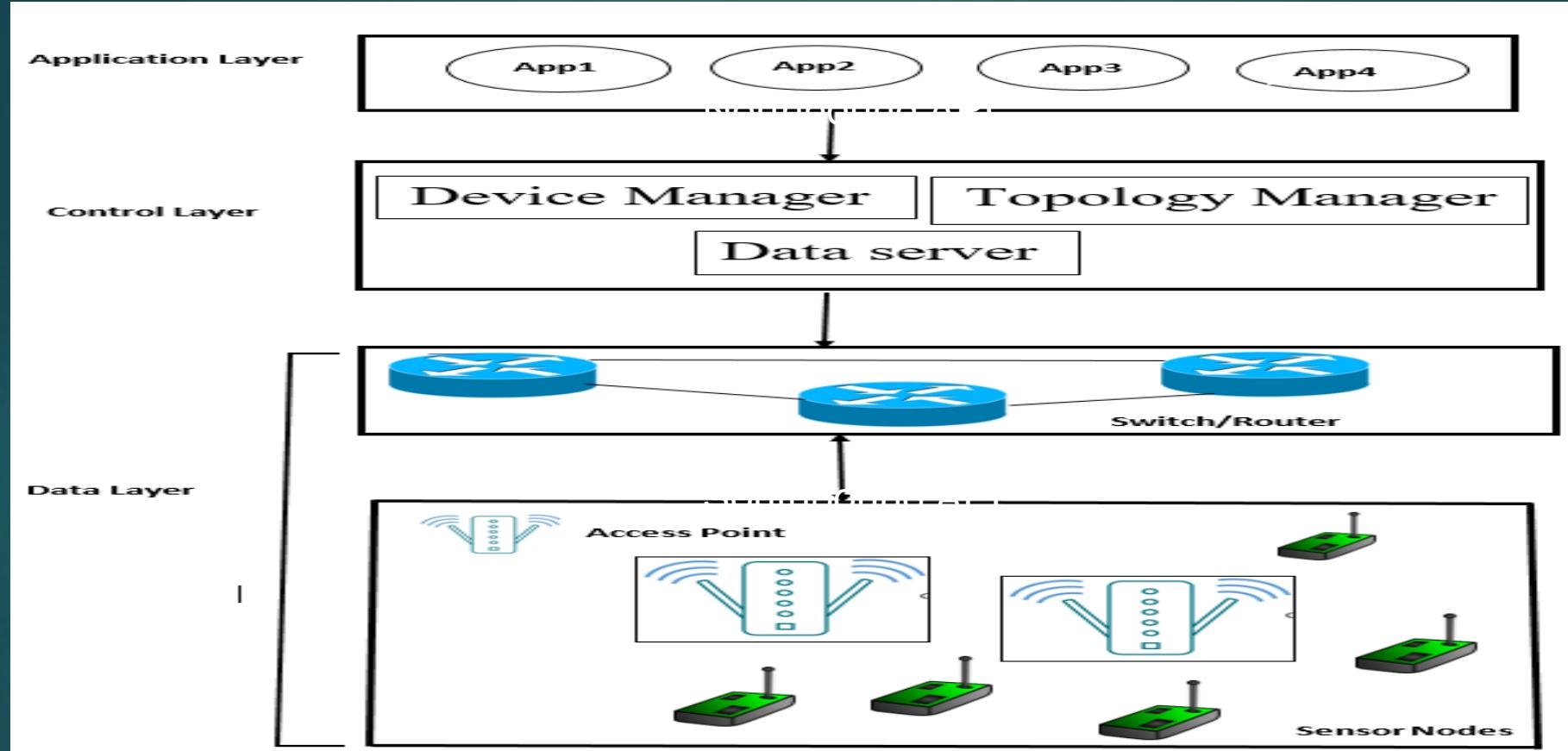
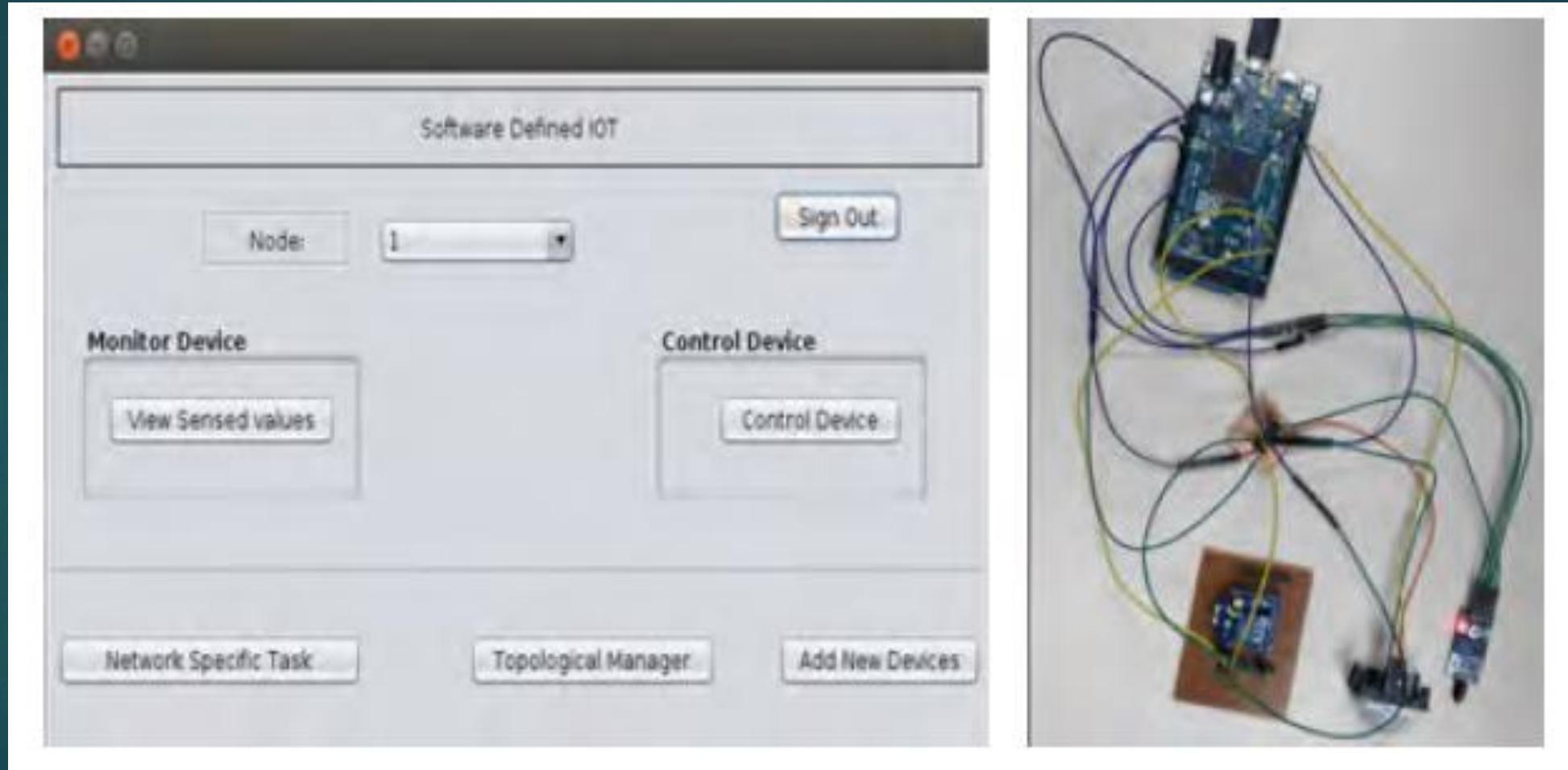


Fig: SDN-WSN Architecture

# Schematic diagram of Developed Application



# Results Achieved

1. Packet delivery ratio is larger in case of Soft-WSN due to intelligent routing by controller as compared to traditional WSN.
2. Energy Consumption is reduced in case of Soft-WSN due to central coordination as compared to traditional WSN. Due to which the lifetime of network also increases.
3. Message Overhead is reduced in case of Soft-WSN.

# Sensor Open flow: Enabling SDN enabled wireless Network.

- WSN was thought to be of as application specific field.
- Policy changes because of changing business needs in WSN is a manual process.
- But if the applications are configurable then WSN will be versatile in nature.
- WSN is hard to manage because of distributed nature.
- The whole idea is to make data plane programmable by manipulating the flow table on each sensor node.
- Controller perform intelligent routing and QoS.

# Some Ideas to integrate SDN with IOT

- MQTT protocol is used in the various IOT devices.
- To simulate data from MQTT protocol Mosquitto is required to be installed on mininet wifi/raspberry pi which help communication between the IOT devices.
- IOT devices will connect to the SDN controller on mininet wifi using mosquito broker.
- MQTT running on mininet wifi is sending the distance information to the smartphone.

# Different Attacks in SDN

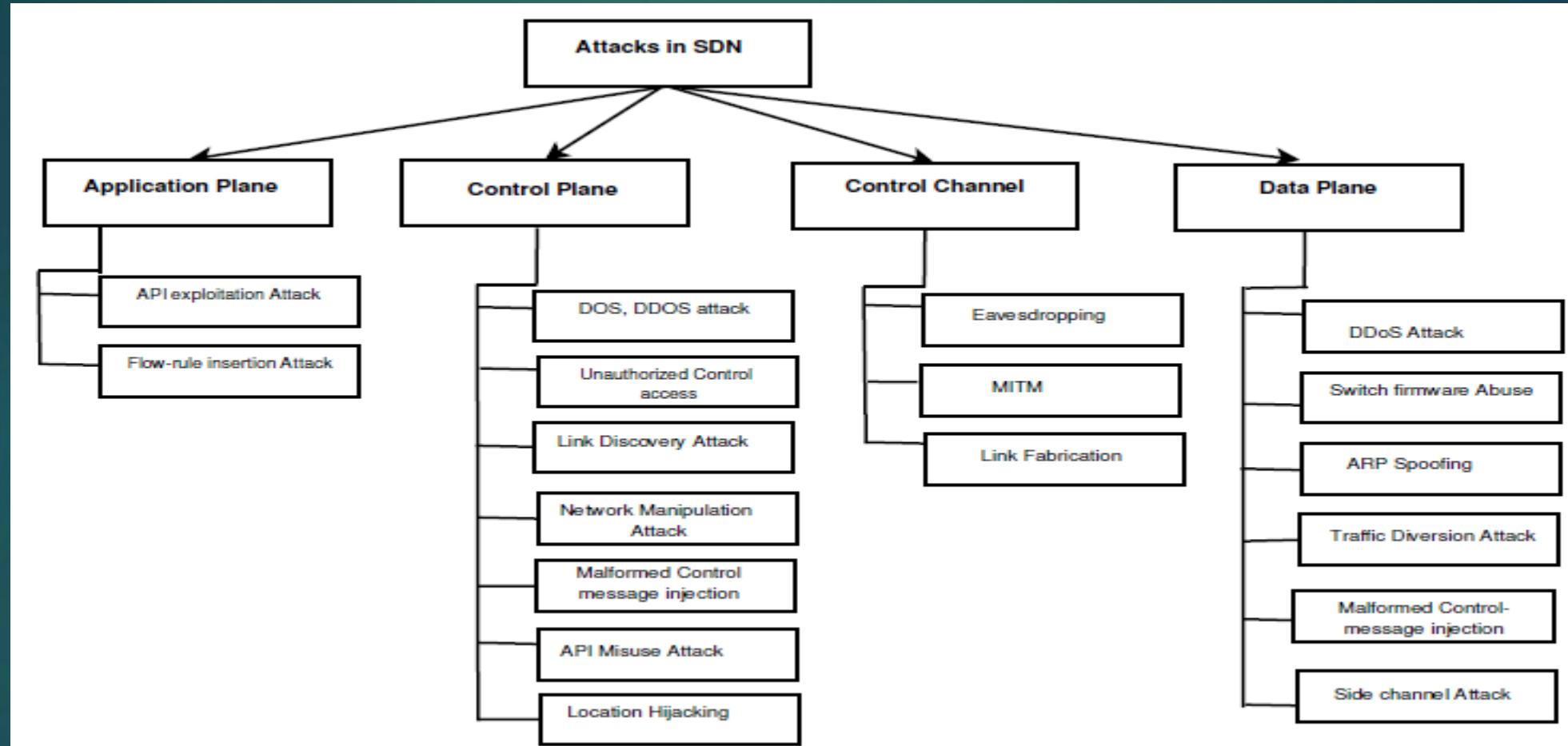


Fig : Attack Taxonomy in SDN

# Different Attacks in SDN

There are various type of attacks possible in SDN. Some of them are:

1. Attacks at Data Plane Layer:
  - a) Denial of Service (DoS) attack: Attacker attempts to make a network resource unavailable to its intended users.
  - b) Spoofing & Tampering: Attacker successfully enter new flows by spoofing either the Northbound API or Southbound API [3].
  - c) Traffic Diversion: This attack occurs to the network element at the data plane. This attack compromises a network element to divert the traffic [4].

# Different Attacks in SDN

## 2. Attacks at Controller Layer:

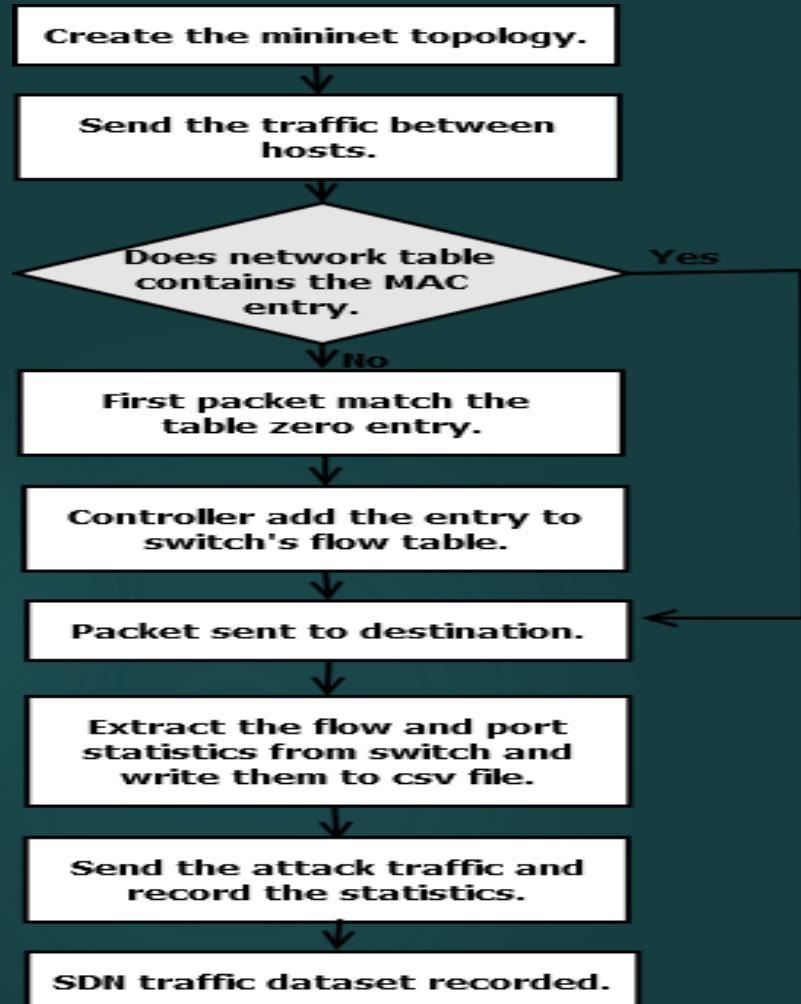
- a) Spoofing & Tampering attack: Attacker impersonate as controller and tamper with the flow table entries of different hosts [5].
- b) Distributed Denial of Service (DDoS) attack: Attackers perform resource consumption attacks on the controller to bog it down and cannot cater to the benign traffic.

# Different Attacks in SDN

## 3. Attacks at Control-Link Channel

- a) Man-in-the-Middle Attack (MITM): Attacker maliciously steal the information exchange between two genuine users and eavesdrop the communication between them [6]. One of the way to perform MITM attack is through ARP poisoning.
- b) Replay attacks: By accessing the timestamp field of the message, it may lead to attack in which benign traffic is maliciously delayed and is known as replay attack.

# DDOS attack detection in SDN



1. Implemented an application to record the various statistics of traffic running in the network.
2. Application collects the various flow statistics and port statistics from various switches.
3. Based on the collected statistics, other parameters are calculated ( Number of Flows, Number of Packet\_ins, Packet Rate, Average packet per flow, Average Byte per flow)
4. All are compared with a predefined threshold limit. If the parameters exceeds the limit, an attack is detected and thus the record is annotated as Attack else normal.

Fig : Flowchart of dataset Creation.

# DDOS attack detection in SDN

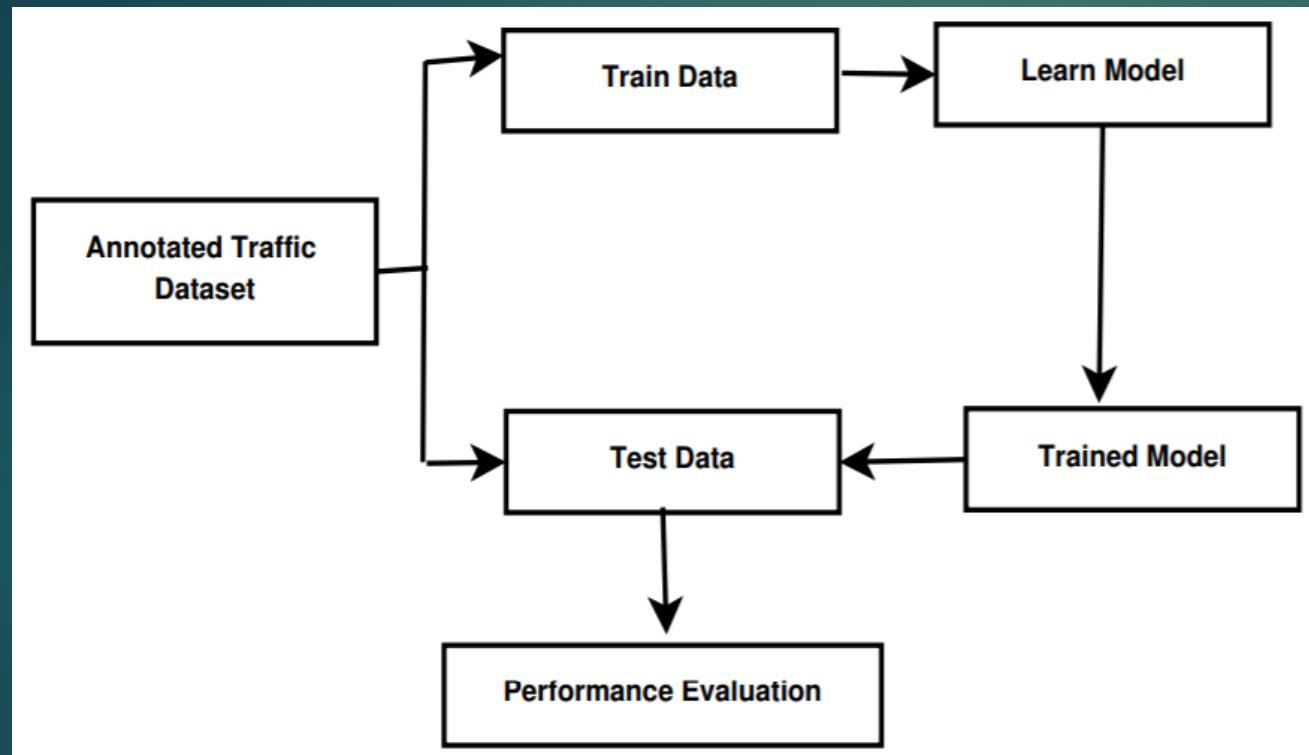


Fig : Machine Learning Algorithm applied to the dataset Created.

Dataset is created with 1,04,345 rows.

After the dataset is recorded, the various machine learning and deep learning[10] algorithms are trained and classification of traffic is done.

# Taxonomy of Topology Attacks

Topology tampering [18] can be possible by following attacks:

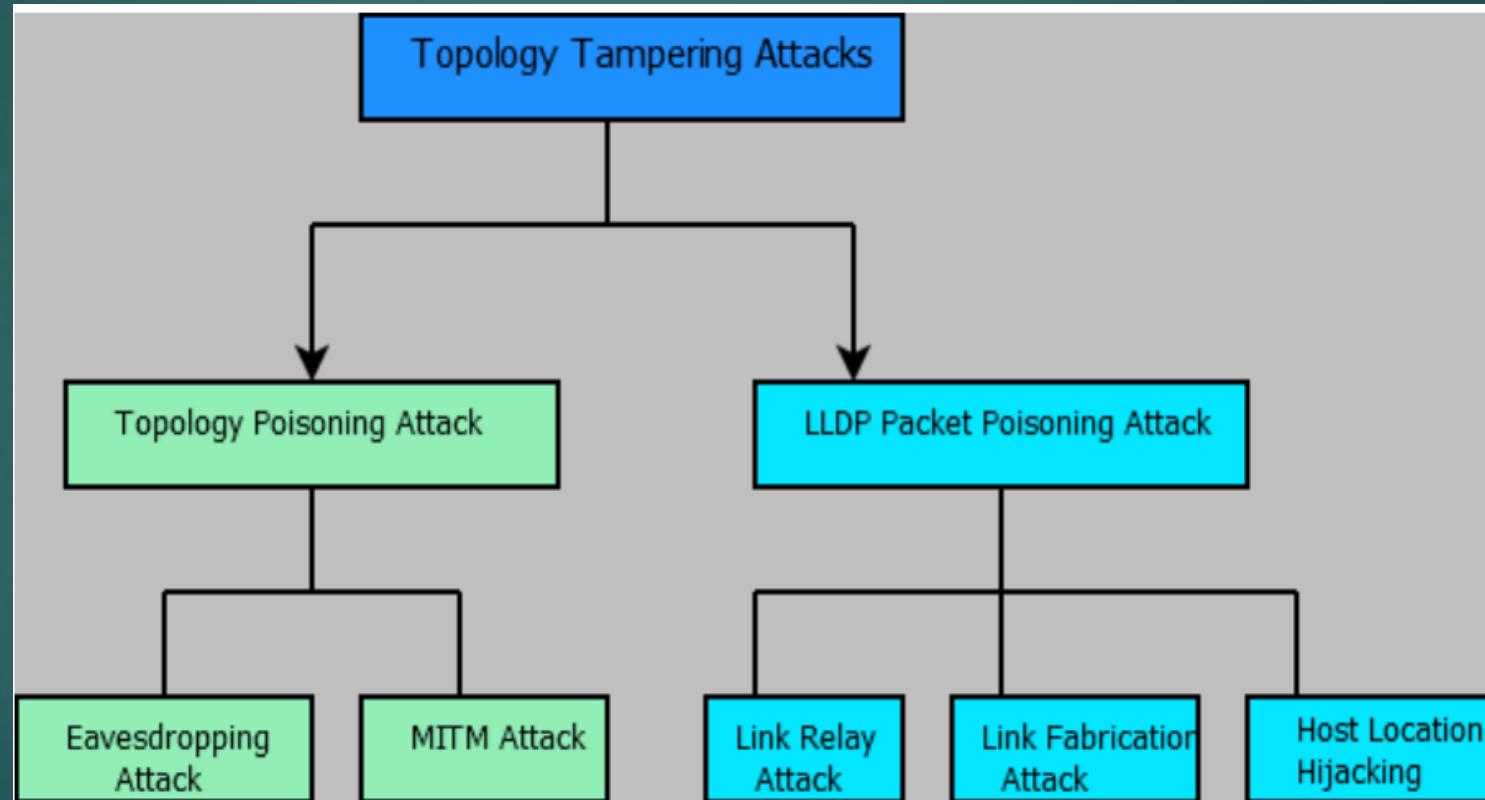
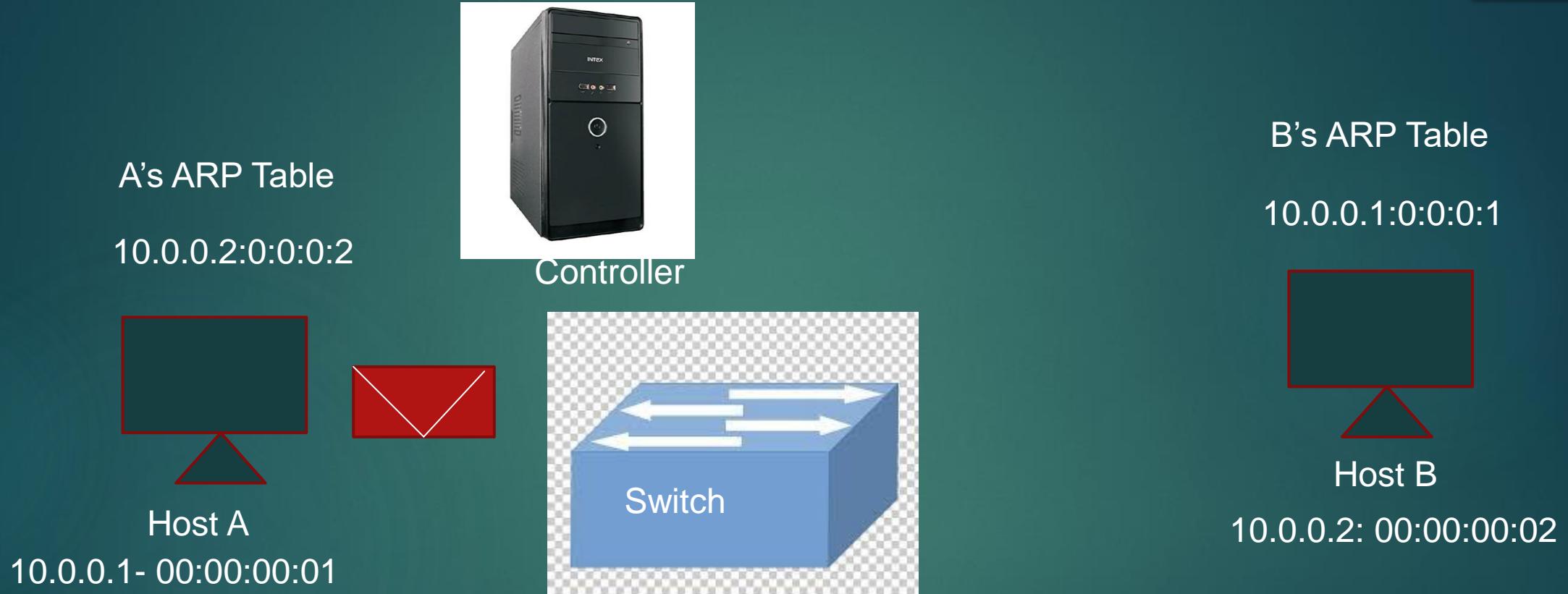


Fig : Topology attacks taxonomy.

# Topology Discovery Process



**Fig: Concept of Topology poisoning attack.**

- ARP request is broadcasted to all the remaining hosts in the topology.
- Destination host update the Source MAC in its ARP table and reply with ARP reply.
- When the Host A get the reply packet, it also updates its ARP table with Host B Mac address.
- In this way the ARP table of both the hosts get updated [17] and normal communication take place.

# Topology Poisoning Attack

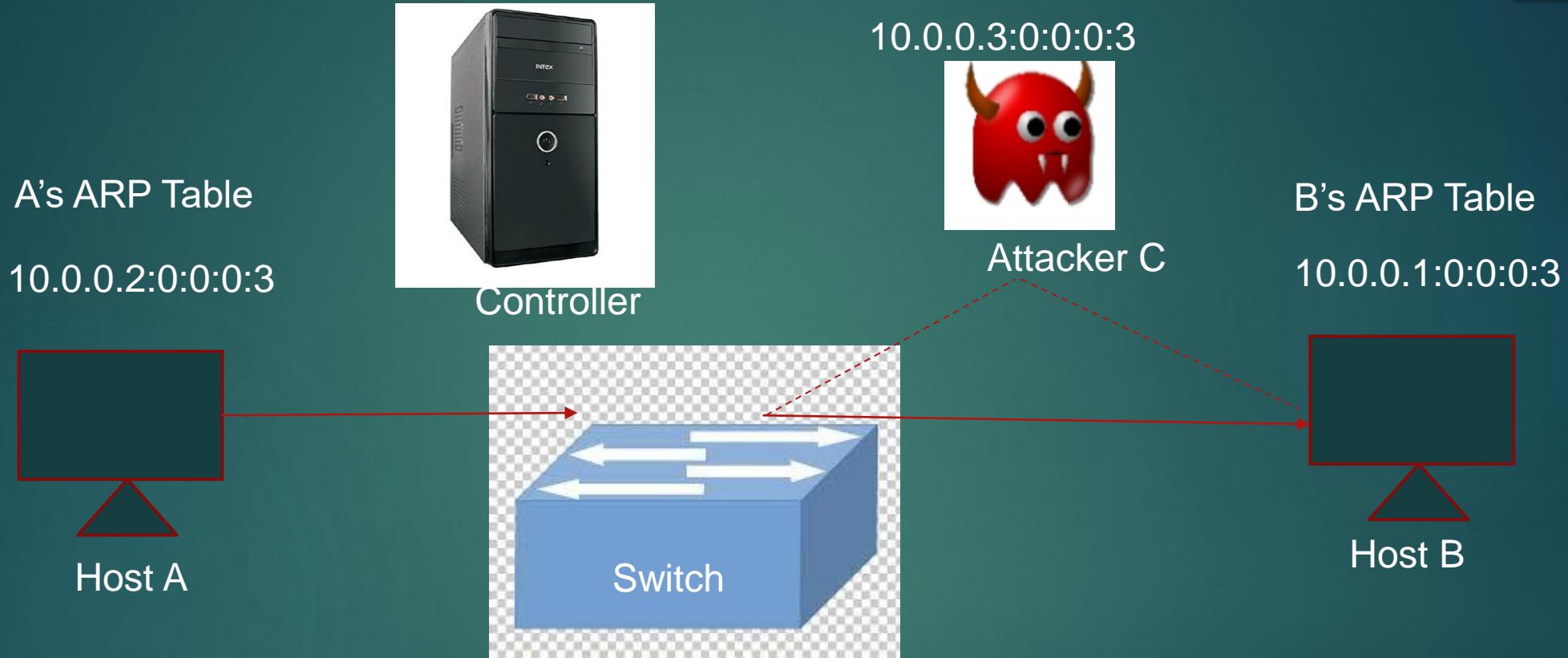


Fig: Concept of Topology poisoning, Flood attack.

- Attacker initiate the ARP request as "who is 10.0.0.1 tell 10.0.0.2" with some random MAC address.
- Attacker keeps on sending such requests from multiple MAC addresses and thus overflow the ARP table of genuine host with number of such requests.
- This way ARP table is flooded with random MAC addresses. Now if any genuine host want to communicate with Host A, it is not fulfilled as the ARP table cannot accommodate more entries.

# Dataset Description

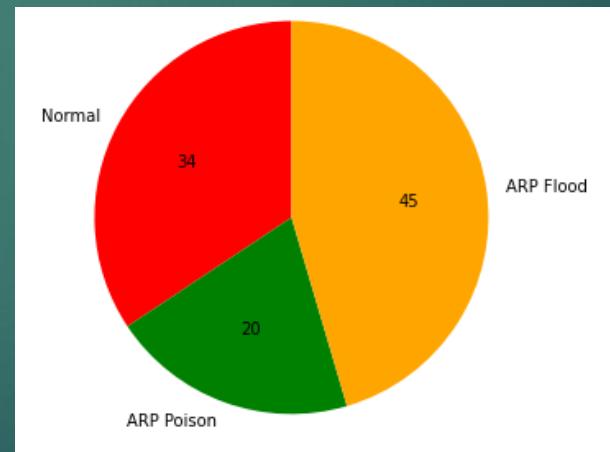
29

**Table 3: Features Used in the created Dataset.**

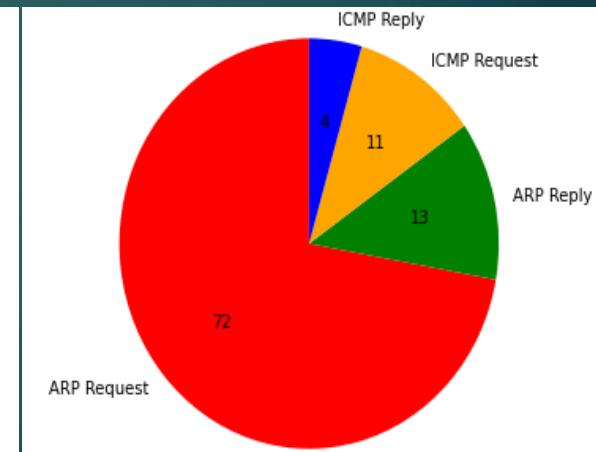
S.No.	Features Used
1	Source MAC Address at ethernet
2	Source MAC address at ARP header
3	Sender IP Address in ARP request.
4	Target IP Address in ARP request.
5	Protocol Code
6	Destination MAC Address at ARP
7	Destination MAC Address at Ethernet
8	Time to live (TTL)
9	Switch-ID.
10	Ping statistics.
11	in_port, out_port.
12	Operation Code.
13	Round trip time.
14	Packet loss
15	Number of Packet _ in messages

**Table 4: Message wise Traffic Analysis of the dataset**

Traffic class	Benign	ARP Poison	ARP Flooding
ARP request	30570	13345	76186
ARP reply	5315	1535	138
ICMP request	4570	235	57
ICMP reply	1893	57	39



**Fig : Traffic Analysis in the complete Dataset**



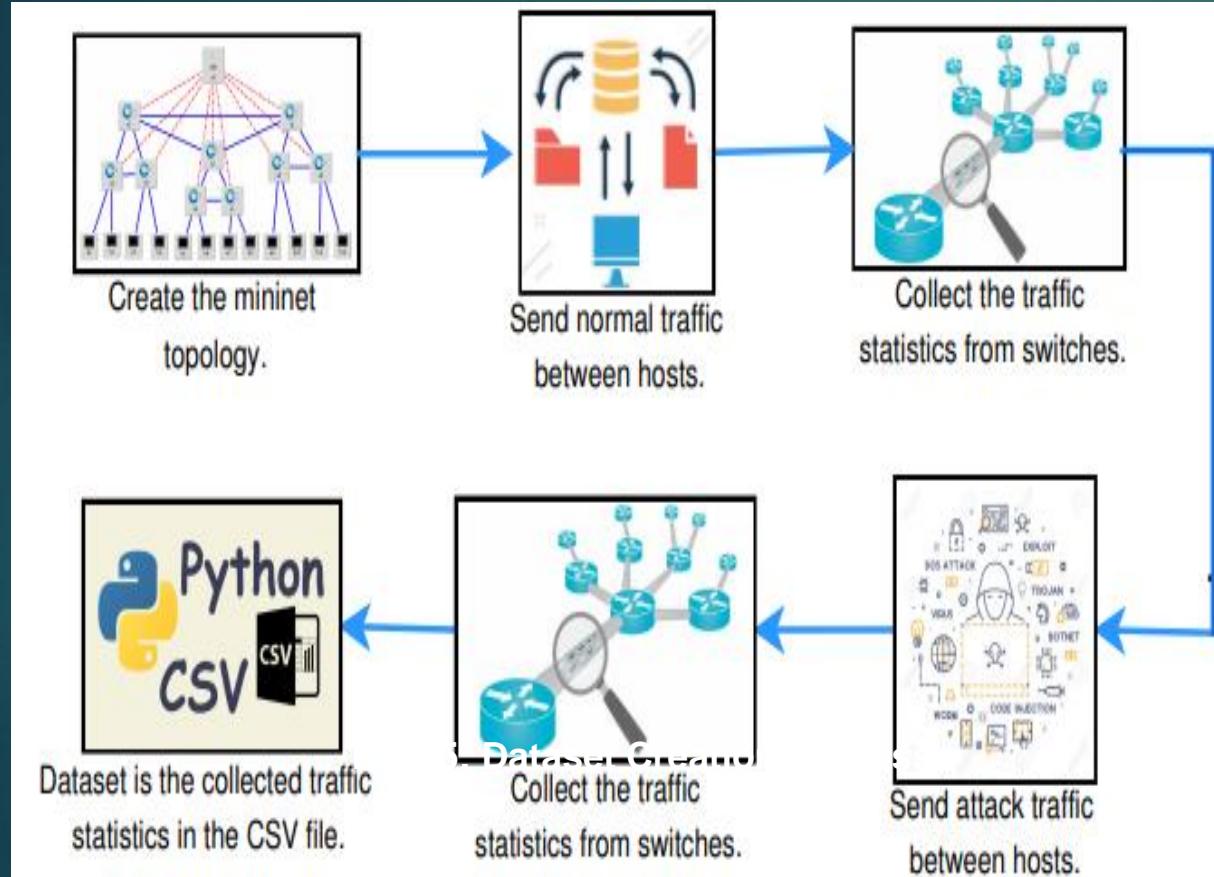
**Fig : Message wise Traffic Analysis of ARP Poison Traffic.**

# Dataset Snapshot

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	switch	switch	in_port	outport	src_mac_addr(eth)	src_mac_addr(air)	dst_mac_addr(eth)	dst_mac_addr(air)	src_ip(arp)	dst_ip(arp)	op_code(arp)	packet_in	Protocol	Pkt loss	rtt	ttl	Label
2	5	5	1	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1707	0	0	0	64	0		
3	2	2	3	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1708	0	0	0	64	0		
4	5	4	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1709	0	0	0	64	0		
5	4	3	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1710	0	0	0	64	0		
6	3	1	1	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1711	0	0	0	64	0		
7	1	6	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1712	0	0	0	64	0		
8	6	10	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1713	0	0	0	64	0		
9	10	8	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1714	0	0	0	64	0		
10	7	7	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1715	0	0	0	64	0		
11	11	9	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1716	0	0	0	64	0		
12	9	12	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1717	0	0	0	64	0		
13	5	11	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1718	0	0	0	64	0		
14	2	13	4	4.29E+09	00:00:00:00:00:07	00:00:00:00:00:00:ff:ff:ff:ff:ff:ff	00:00:00:00:00:10.0.0.7	10.0.0.12	1	1719	0	0	0	64	0		
15	3	7	3	4 00:00:00:00:00:0c	00:00:00:00:00:00:00:00:00:00:07	00:00:00:00:00:10.0.0.12	10.0.0.7	2	1720	0	0	0	64	0			
16	4	6	1	4 00:00:00:00:00:0c	00:00:00:00:00:00:00:00:00:07	00:00:00:00:10.0.0.12	10.0.0.7	2	1721	0	0	0	64	0			
17	1	1	2	1 00:00:00:00:00:0c	00:00:00:00:00:00:00:00:00:07	00:00:00:00:10.0.0.12	10.0.0.7	2	1722	0	0	0	64	0			
18	10	2	4	3 00:00:00:00:00:0c	00:00:00:00:00:00:00:00:00:07	00:00:00:00:10.0.0.12	10.0.0.7	2	1725	0	0	0	64	0			
19	6	5	4	1 00:00:00:00:00:0c	00:00:00:00:00:00:00:00:00:07	00:00:00:00:10.0.0.12	10.0.0.7	2	1727	0	0	0	64	0			
20	13	7	3	4 00:00:00:00:00:0c	00:00:00:00:00:00:00:00:00:07	00:00:00:00:10.0.0.12	10.0.0.7	1	2180	0	0	0	64	0			
21	11	6	1	4 00:00:00:00:00:0c	00:00:00:00:00:00:00:00:00:07	00:00:00:00:10.0.0.12	10.0.0.7	1	2181	0	0	0	64	0			
22	12	1	2	1 00:00:00:00:00:0c	00:00:00:00:00:00:00:00:00:07	00:00:00:00:10.0.0.12	10.0.0.7	1	2182	0	0	0	64	0			
23	9	2	4	3 00:00:00:00:00:0c	00:00:00:00:00:00:00:00:00:07	00:00:00:00:10.0.0.12	10.0.0.7	1	2183	0	0	0	64	0			

Fig: Dataset for ARP Poison and ARP Flood attack with 15 features.

# ARP Poison Attack Detection

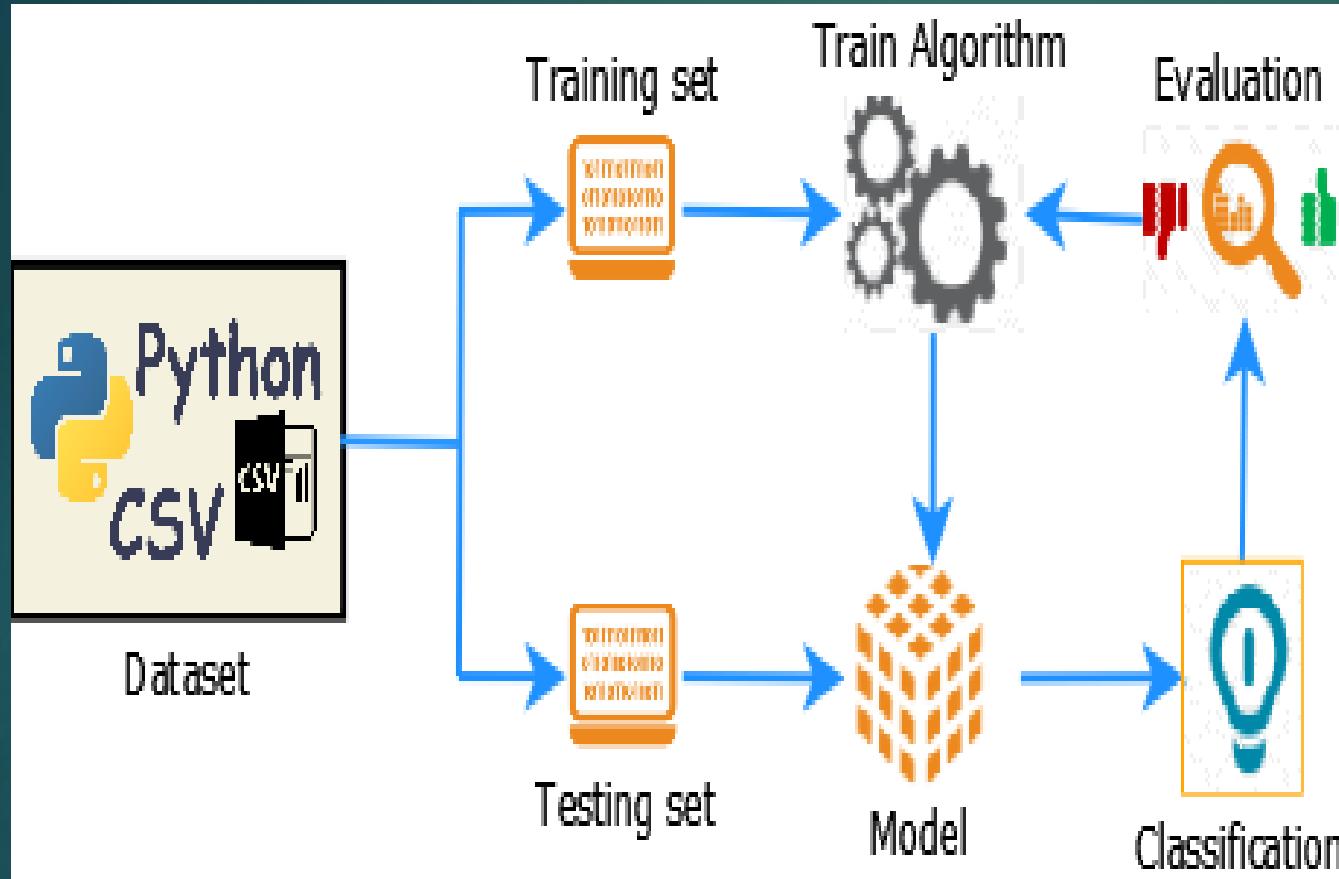


► Fig: Dataset creation for ARP based attacks in SDN.

Steps to create Dataset creation:

- ❖ A python application is created to extract the different features of the traffic at ARP and IP layer.
- ❖ ARP and IP related features are written into respective CSV files.
- ❖ Normal traffic and attack traffic is run and the features are extracted.
- ❖ Both the files are combined based on the common Date Time field and Dataset is created.

# ARP Poison Attack Detection



- A Dataset is created with 1,34,000 records in a CSV file.
- Different Machine Learning [12] and Deep Learning algorithms are applied for traffic classification.

Fig: Traffic classification using Machine Learning

# Blockchain Introduction

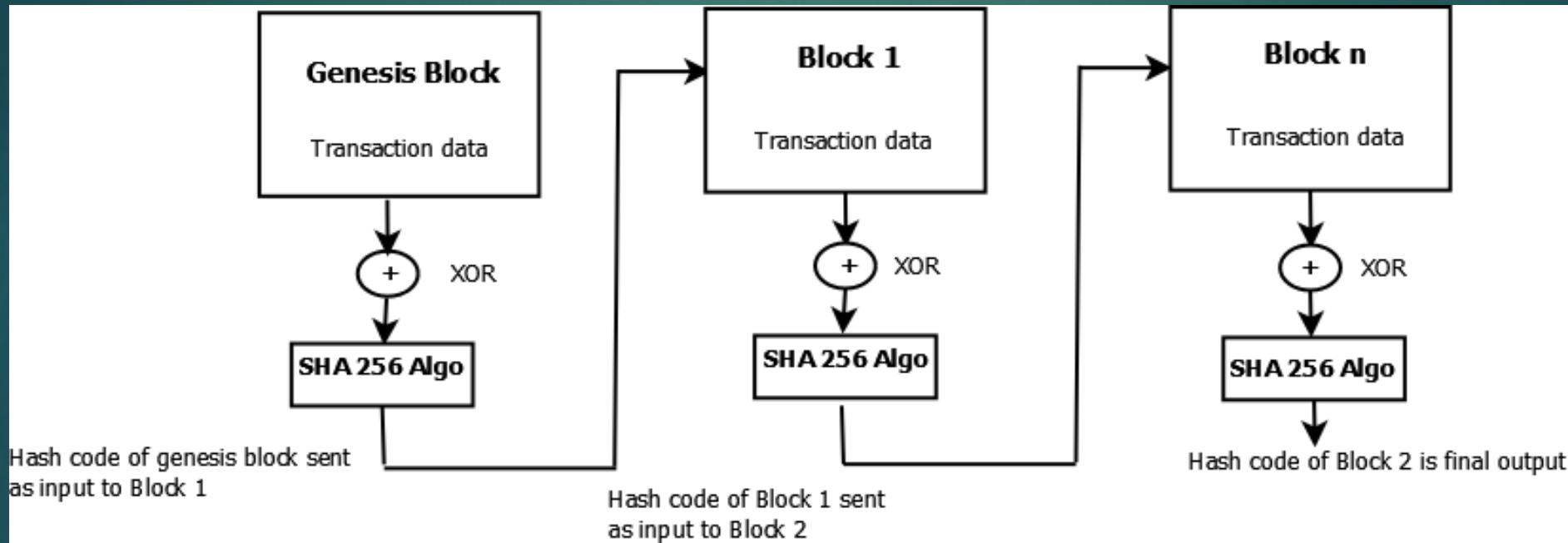


Fig: Diagram depicting the Concept of Blockchain

# Blockchain Principles

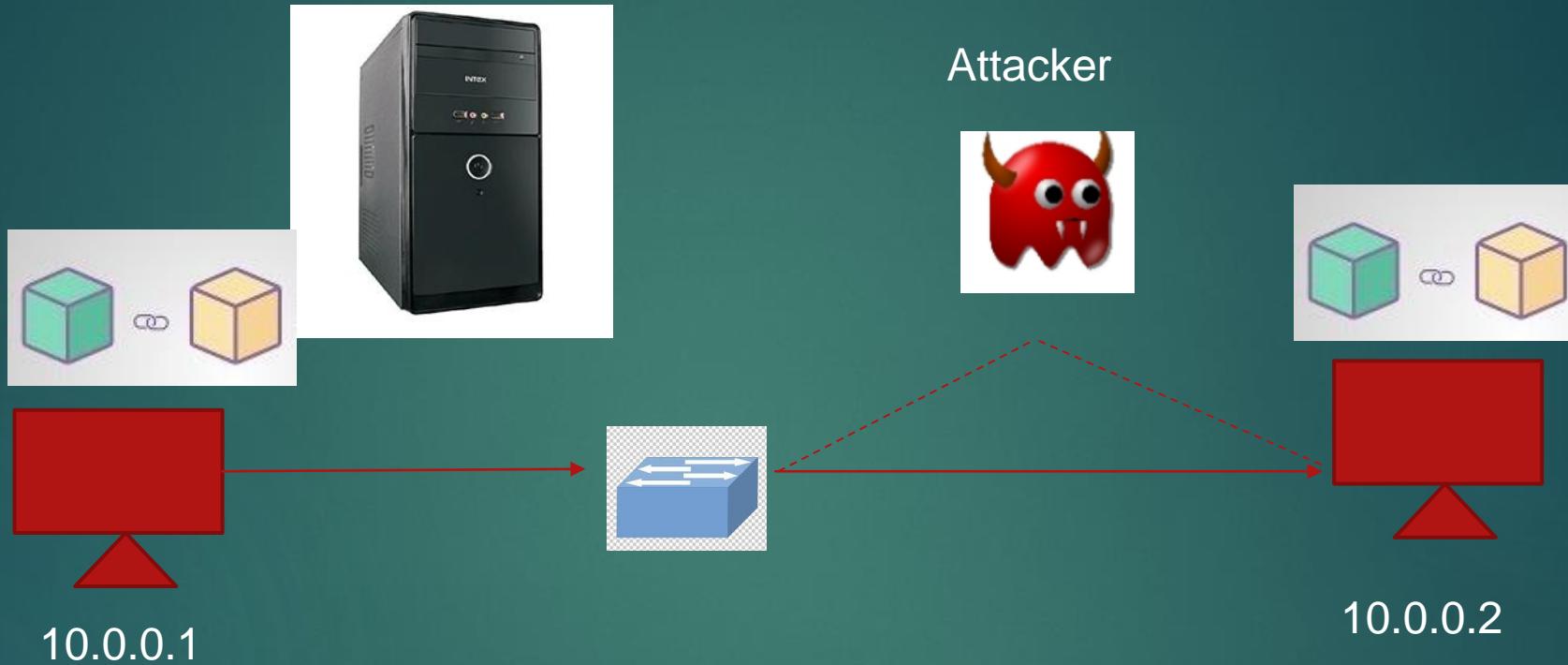
- ▶ Trust: Network elements work upon updation of table entries together but each updating the entries together rather than each one separately.
- ▶ Integrity: To update or read the block data, one needs access rights which makes the blockchain an important security feature.
- ▶ Resiliency: In Blockchain every block has its importance since the information is stored in all the blocks.
- ▶ Privacy: Blockchain permanently stores the data of older generated blocks to maintain records of all the transactions. This way the new blocks can be checked for their consistency and authenticity by looking into the previous records.

# ARP Poisoning Attack



1. Attacker initiate the ARP request as "who is 10.0.0.1 tell 10.0.0.2" with MAC address as attacker Mac address and also "who is 10.0.0.2 tell 10.0.0.1" with MAC address of attacker.
2. Host A and B update their ARP table with 10.0.0.2 with MAC address of attacker and Host B with 10.0.0.1 with MAC address of attacker.
3. This way ARP table is poisoned. Now any communication between Host A and B is going through attacker C.

# Blockchain Perspective



**Fig: Blockchain for ARP Poisoning attack.**

1. Before updating the ARP tables this arp request block is sent to every host in the network.
2. Host 1 will reject the block as its IP/MAC pairing is wrongly reflected in the ARP request, so the block will not be validated and hence the transaction got rejected and thus ARP poisoning will not take place.

# Proposed Architecture

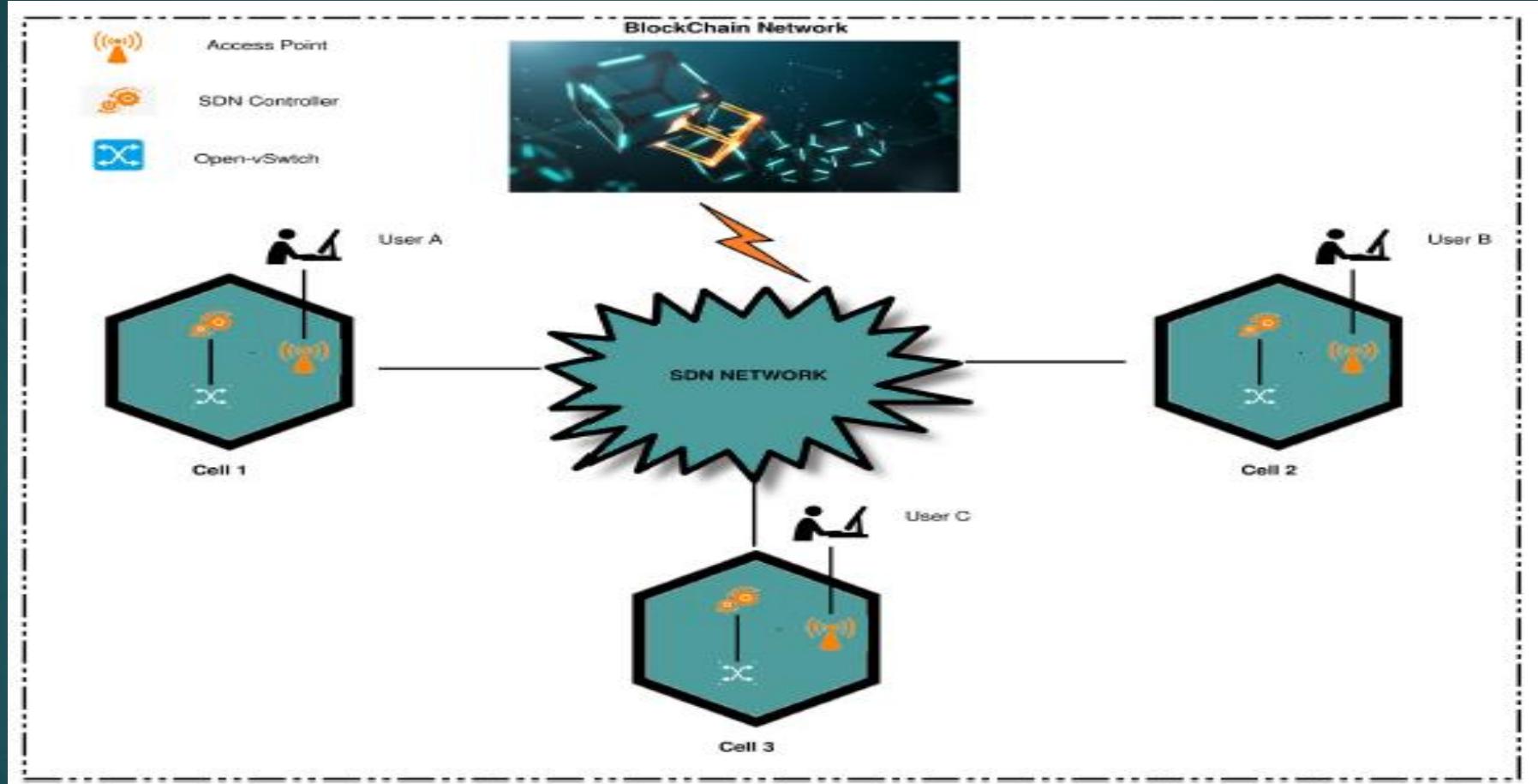


Fig: Blockchain Architecture for SDN.

# Conclusion

- ❖ SDN network has redefined the networking domain by making it programmable and the various network devices can be programmed remotely supporting the dynamic configuration of the network.
- ❖ The programming capability of the network can be employed to solve the various issues which arise in traditional network, IOT network.
- ❖ In the past, different Statistical and Cryptographic techniques have been used to detect the attacks in network.
- ❖ Different statistical techniques which are applied are found to be less competitive in comparison with Machine Learning approach.
- ❖ Blockchain can be integrated with SDN to make SDN more secure

# References

1. Anand, N., Sarath Babu, and B. S. Manoj. "On detecting compromised controller in software defined networks." *Computer Networks* 137 (2018): 107-118.
2. Maimó, L. F., Gómez, A. L. P., Clemente, F. J. G., & Pérez, M. G. A Self-Adaptive Deep Learning-Based System for Anomaly Detection in 5G Networks.
3. Latah, Majd, and Levent Toker. "Towards an Efficient Anomaly-Based Intrusion Detection for Software-Defined Networks." *arXiv preprint arXiv:1803.06762* (2018).
4. Aldwairi, Tamer, Dilina Perera, and Mark A. Novotny. "An evaluation of the performance of Restricted Boltzmann Machines as a model for anomaly network intrusion detection." *Computer Networks* 144 (2018): 111-119.
5. AlEroud, Ahmed, and Izzat Alsmadi. "Identifying cyber-attacks on software defined networks: An inference-based intrusion detection approach." *Journal of Network and Computer Applications* 80 (2017): 152-164.

# References

6. Liu, Hongyu, et al. "CNN and RNN based payload classification methods for attack detection." *Knowledge-Based Systems* 163 (2019): 332-341.
7. Dabbagh, Mehiar, et al. "Software-defined networking security: pros and cons." *IEEE Communications Magazine* 53.6 (2015): 73-79.
8. Tang, Tuan A., et al. "Deep learning approach for network intrusion detection in software defined networking." *Wireless Networks and Mobile Communications (WINCOM), 2016 International Conference on*. IEEE, 2016.
9. Mohammadi, Reza, Reza Javidan, and Mauro Conti. "Slicots: An sdn-based lightweight countermeasure for tcp syn flooding attacks." *IEEE Transactions on Network and Service Management* 14.2 (2017): 487-497.
10. Yoon, Changhoon, et al. "Flow wars: Systemizing the attack surface and defenses in software-defined networks." *IEEE/ACM Transactions on Networking* 6 (2017): 3514-3530.

# References

11. Maninderpal Singh, Gagangeet Singh Singh Aujla, Amritpal Singh, Neeraj Kumar, and Sahil Garg. Deep learning based blockchain framework for secure software defined industrial networks. *IEEE Transactions on Industrial Informatics*, 2020a.
12. Nehra, A., Tripathi, M., & Gaur, M. S. FICUR: Employing SDN programmability to secure ARP. In 2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 1-8). IEEE.
13. Sebbar, A., Zkik, K., Boulmalf, M., & El Kettani, M. D. E. C. (2019). New context-based node acceptance CBNA framework for MitM detection in SDN Architecture. *Procedia Computer Science*, 160, 825-830.137.
14. Alharbi, T. (2020). Deployment of blockchain technology in software defined networks: A survey. *IEEE Access*, 8, 9146-9156.
15. Wenjuan Li, Weizhi Meng, Zhiqiang Liu, and Man-Ho Au. Towards blockchain-based software-defined networking: security challenges and solutions. *IEICE Transactions on Information and Systems*, 103(2):196–203, 2020.

# References

16. M. Brooks, B. Yang, A man-in-the-middle attack against opendaylight sdn controller, in: Proceedings of the 4<sup>th</sup> Annual ACM Conference on Research in Information Technology, 2015, pp. 45–49.
17. S. Hong, L. Xu, H. Wang, G. Gu, Poisoning network visibility in software-defined networks: New attacks and countermeasures., in: NDSS, Vol. 15, 2015, pp. 8–11.
18. S. Y. Nam, D. Kim, J. Kim, Enhanced arp: preventing arp poisoning-based man-in-the-middle attacks, IEEE communications letters 14 (2) (2010) 187–189.
19. Zakaria Abou El Houda, Abdelhakim Senhaji Hafid, and Lyes Khoukhi. Cochain-sc: An intra-and inter-domain ddos mitigation scheme based on blockchain using sdn and smart contract. IEEE Access, 7:98893–98907, 2019.
20. Durbadal Chattaraj, Sourav Saha, Basudeb Bera, and Ashok Kumar Das. On the design of blockchain-based access control scheme for software defined networks. In IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pages 237–242. IEEE, 2020.



THANKS

Any Queries??

# De-Fence

Security System using LoRa-based  
Hop-to-Hop communication



# Problem Statement and Inspiration



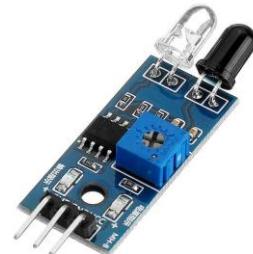
- Limited networking capability in remote areas
- Areas with environment hostile to long range wireless transmission, interference is also involved
- High Power requirements of traditional security systems,
- Current system of human-based manual updates is impractical and prone to blind spots
- Existing infrastructure is fixed and leaves little room for flexibility

# Hardware Used



## Arduino UNO

Arduino Uno is a microcontroller board based on the ATmega328P



## IR Sensor

Electronic device that measures and detects infrared radiation in its surrounding environment, most commonly used in motion-based detection



## Buzzer

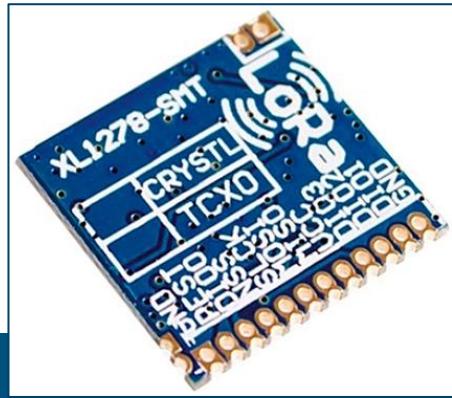
Simple device which can generate beeps and tones, whose working principle is piezoelectric effect



## Jumper Wires

Small cables used to connect other components to network cabling

# Hardware Used



## LoRa XL1278 Module

- Non-cellular, secure, programmable, low-power RF modules, providing long-range IoT data connectivity to sensors and actuators.
- Its **range** is quite high, and can penetrate multiple obstacles.
- Less infrastructure required, making network much cheaper and faster to implement.
- LoRa modules can be recognized accurately even in a low signal-to-noise ratio (SNR) environment. They can be distinguished and identified effectively.

# Components of IoT

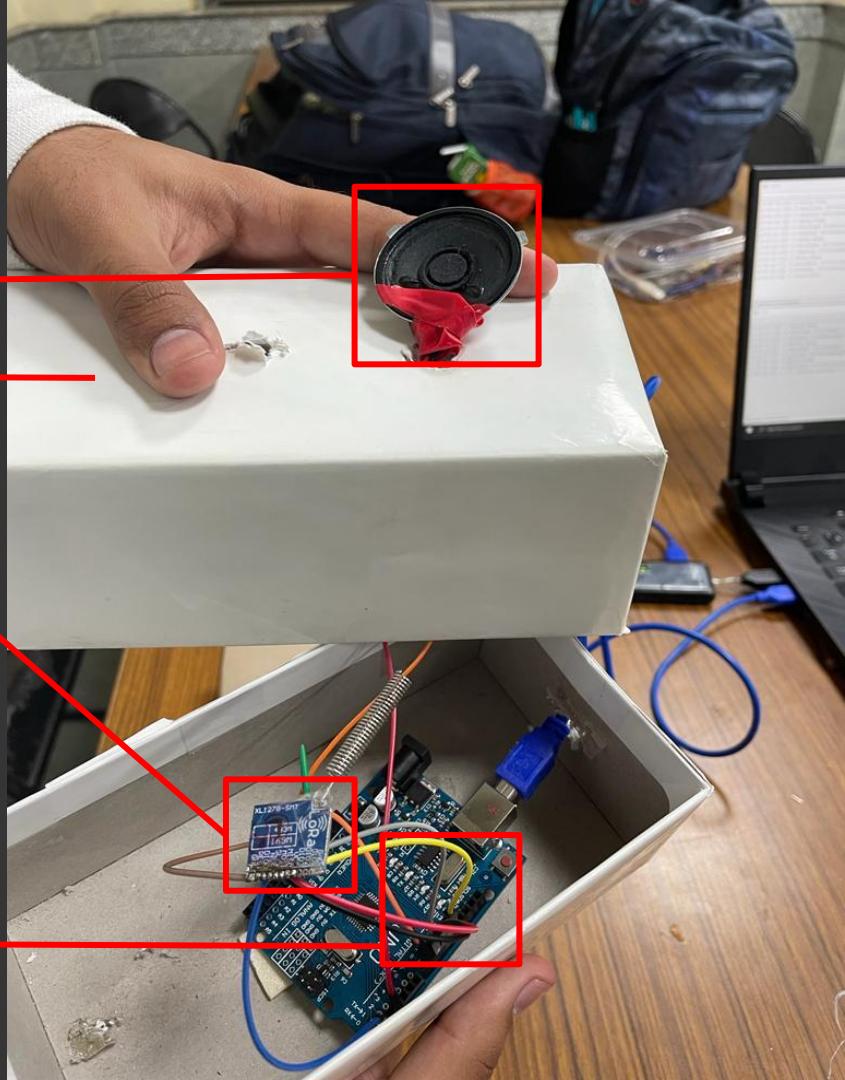
Node  
(Thing)

IR Motion  
(Sensor)

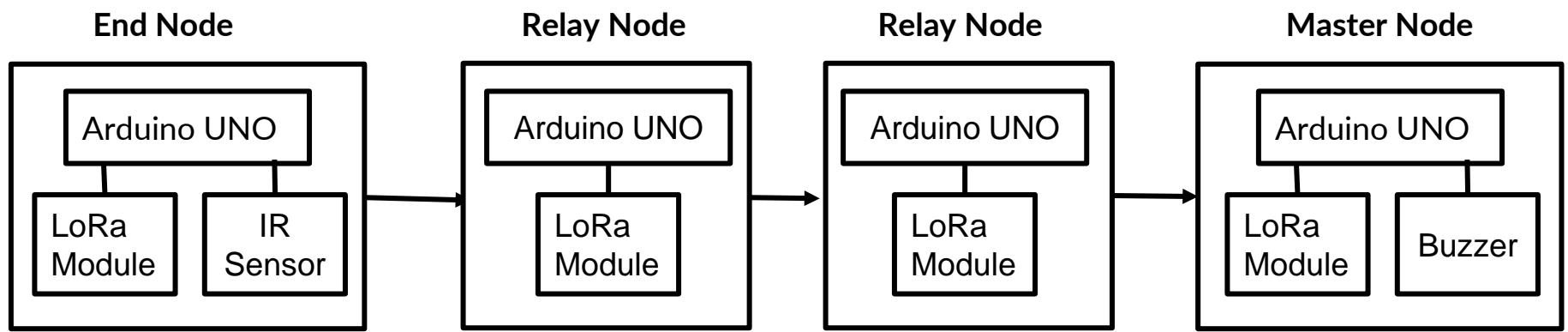
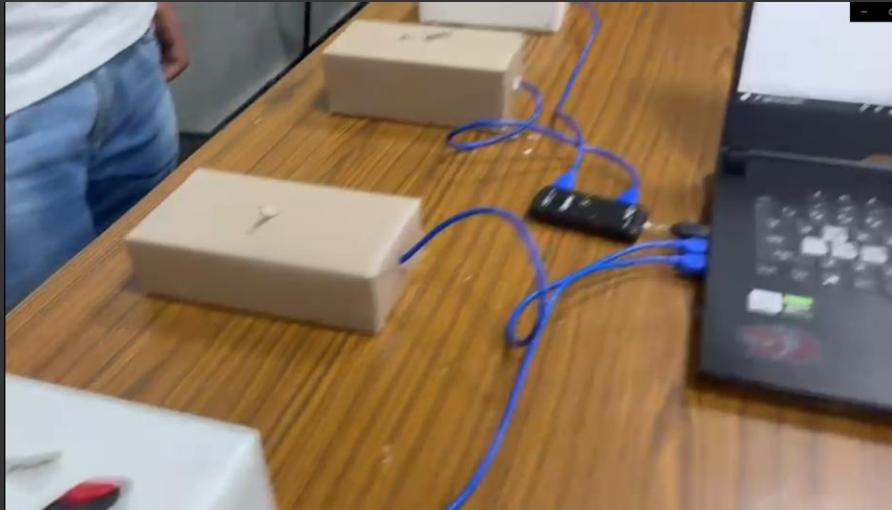
Buzzer  
(Actuator)

LoRa module  
(Communicator)

Arduino  
UNO  
(Controller)



# Schematic Diagram



# Features

## Scalability



As the area to be covered increases, more nodes can be added to the network, in order to ensure a smooth expansion.

## Reliability



The network does not have a single bottleneck and can keep working even if a few of the nodes fail.

## Security



It avoids the risk of cyber threats, especially from the internet, and ensures confidentiality using end-to-end encryption.

# Target Audience

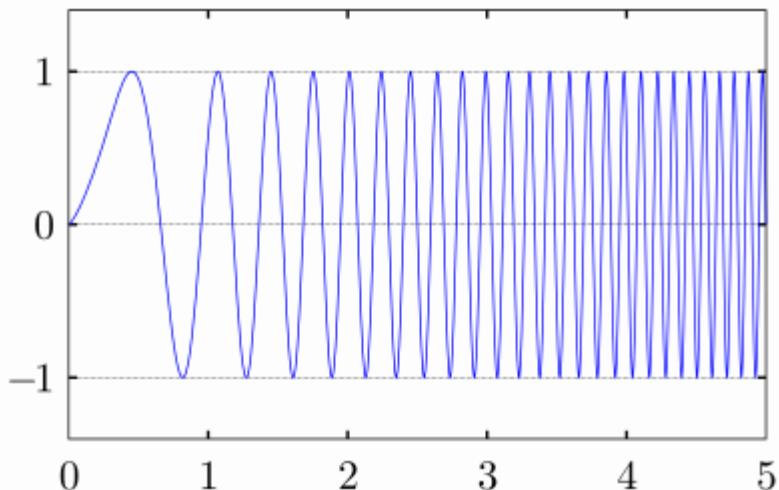


- Dense and widespread security coverage under situations where constant and widespread internet connection is not available, or concentrated electric coverage is not feasible, for example, large compounds or sites in distant locations.
- The stakeholders include agencies/bodies that offer security management systems in remote areas and high altitude water pipeline defect detection. It is suitable for an off-the-grid surveillance and intruder-detection setup.
- This structure is highly versatile and flexible, and it is highly secure against attacks. We can also increase the speed of communication by choosing the optimal path, as well as special pathways for encrypted transmission.
- This would allow the network to be quickly established and easily maintained as portable individual sensors can be easily registered or removed from the network.

# LoRa Technology



- LoRa, from Long Range, and LoRaWAN together define a long range, low power, low bitrate networking protocol.
- LoRa is the proprietary physical radio modulation technique, derived from Chirp Spread Spectrum technology.
- LoRaWAN defines the software communication protocol and system architecture. The continued development of the LoRaWAN protocol is managed by the open, non-profit LoRa Alliance, of which SemTech, the company currently in possession of the patent for LoRa, is a founding member. The latest LoRaWAN version is 1.0.4, released in October 2020.
- The LoRa alliance is a non profit association created to support LoRaWAN protocol, as well as to ensure interoperability of all LoRaWAN technologies.



- Chirp spread spectrum (CSS) is a spread spectrum technique that uses wideband linear frequency modulated chirp pulses to encode information.
- A chirp is a sinusoidal signal whose frequency increases or decreases over time.
- Chirp modulation uses the entirety of its bandwidth, so it is robust against channel noise. They are also resistant to multipath fading making them ideal for use in locations with multiple obstacles. It is also resistant to the Doppler effect. However, unlike other spread spectrum techniques, CSS does not include special elements to distinguish itself from the noise.
- Originally developed to compete with ultra-wideband, it is instead now recommended for personal mobile networks.

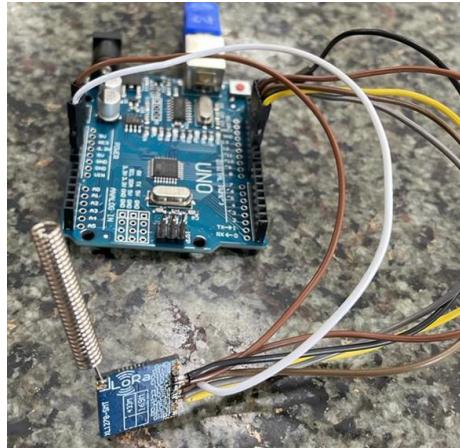
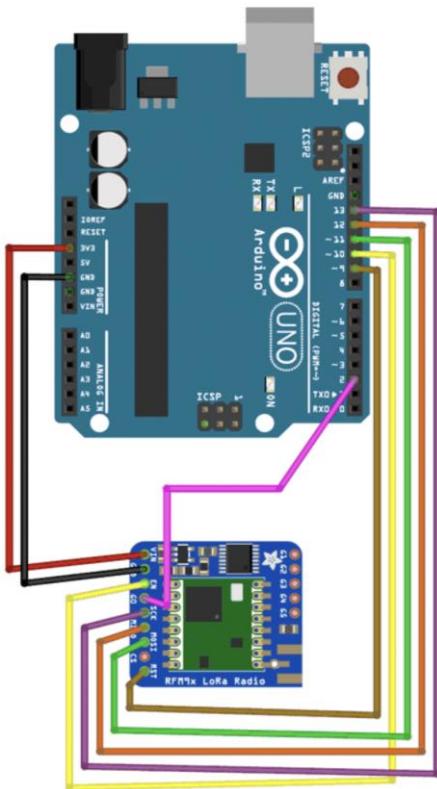


- There is a large variety in LoRa enabling solutions provided by Semtech.
- The LoRa Core portfolio represents the essential capability of Semtech's LoRa devices including long range, low power and cost effective end-to-end communication.
- LoRa Edge is an ultra-low power platform that integrates a long range LoRa transceiver, multi-constellation scanner and passive Wi-Fi AP MAC address scanner targeting GNSS asset management applications.
- LoRa 2.4GHz offers ultra-long range communication in the 2.4GHz band with lowest power and highest reliability connectivity.
- We have used a version of the Semtech SX1278 (LoRa Core), the XL1278, which has simplified circuitry.

# Interfacing components with the Arduino Uno

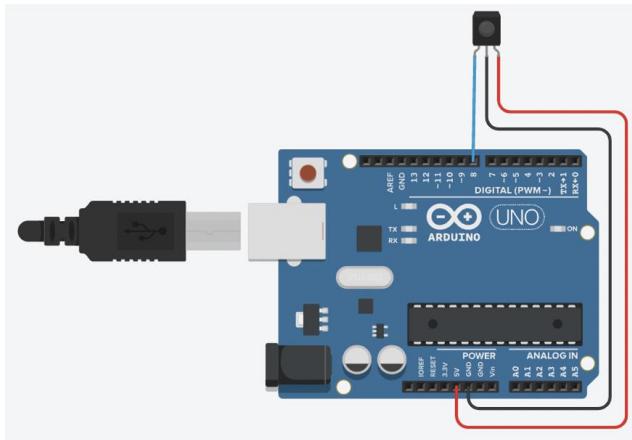
- LoRa XL1278-SMT Module
  - IR sensor
  - Buzzer
-

# Interfacing LoRa XL1278-SMT



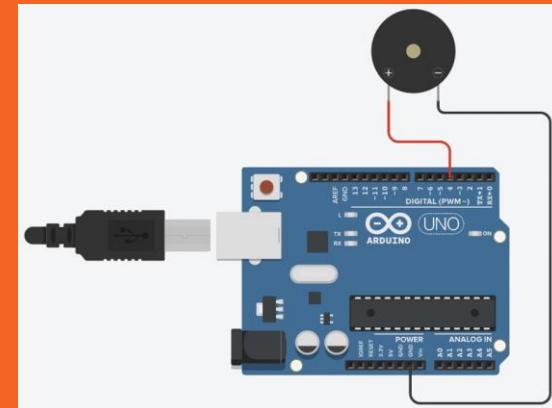
LoRa XL1278 Module	Arduino UNO Board
VCC	3.3V
GND	GND
NSS	D10
DIO0	D2
SLCK	D13
MISO	D12
MOSI	D11
REST	D9

# Interfacing IR sensor



IR Sensor	Arduino Uno
VCC	VCC
GND	GND
OUT	D8

# Interfacing the Buzzer



Buzzer	Arduino Uno
+	D4
-	GND

# Our Prototype

- Path discovery in the ad-hoc network
    - Flooding the network to determine a valid path between the end node and master node
    - Retracing the path to let the end node know about the path
  - Sending the payload
- 1 End node to detect objects using sensors and initiate the data transfer
  - 2 Relay nodes to relay the data from the end node to the master node
  - 1 Master node to process the received information and sound the alarm
-

# Packet Description

[Type] \* [MessageID] \* [Destination] \* [Source] \* [Path] \*  
[Payload] \*

The whole message can be of maximum 256 bytes.

1. **Type** - Flood (F), Retrace (R), Sending payload (P)
2. **MessageID**
3. **Destination node ID**
4. **Source node ID**
5. **Currently traversed(flood)/remaining(retrace and payload) path**
6. **Payload/message**

# End Node

- IR sensor detects a new object in the surroundings
  - Creates a message for the master node
  - Adds self node ID to the path
  - Initiates path discovery by flooding the message
- Retrace message is received
  - Checks if the message is intended for this end node
  - Encrypts the payload
  - Removes the self node from the path
  - Creates the message with encrypted payload and sends

The source and destination IDs are changed as and when required.

# Relay Node

- Flood message is received
  - Change the source node ID in the packet
  - Add the self node ID to the path
  - Flood the message again
- Retrace/Payload message is received
  - Checks if the message is intended for this relay node
  - Removes the self ID from the path
  - Sends the message to the next node in the path

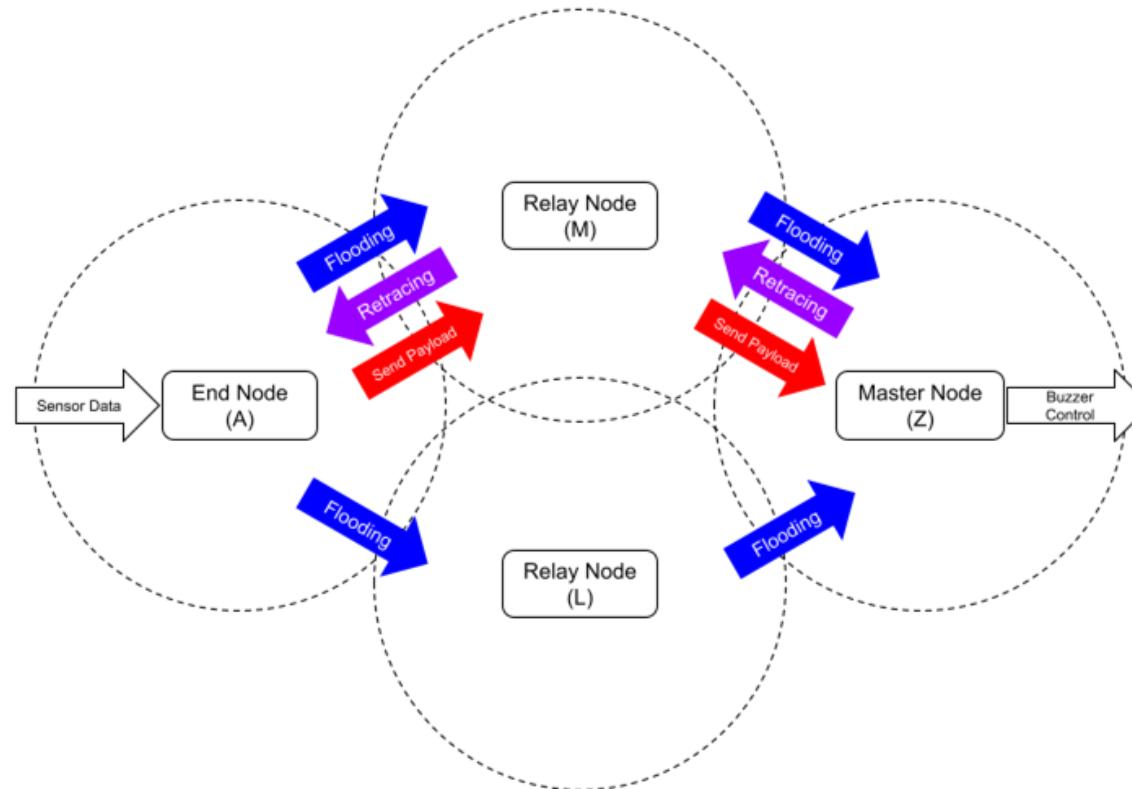
The source and destination IDs are changed as and when required.

# Master Node

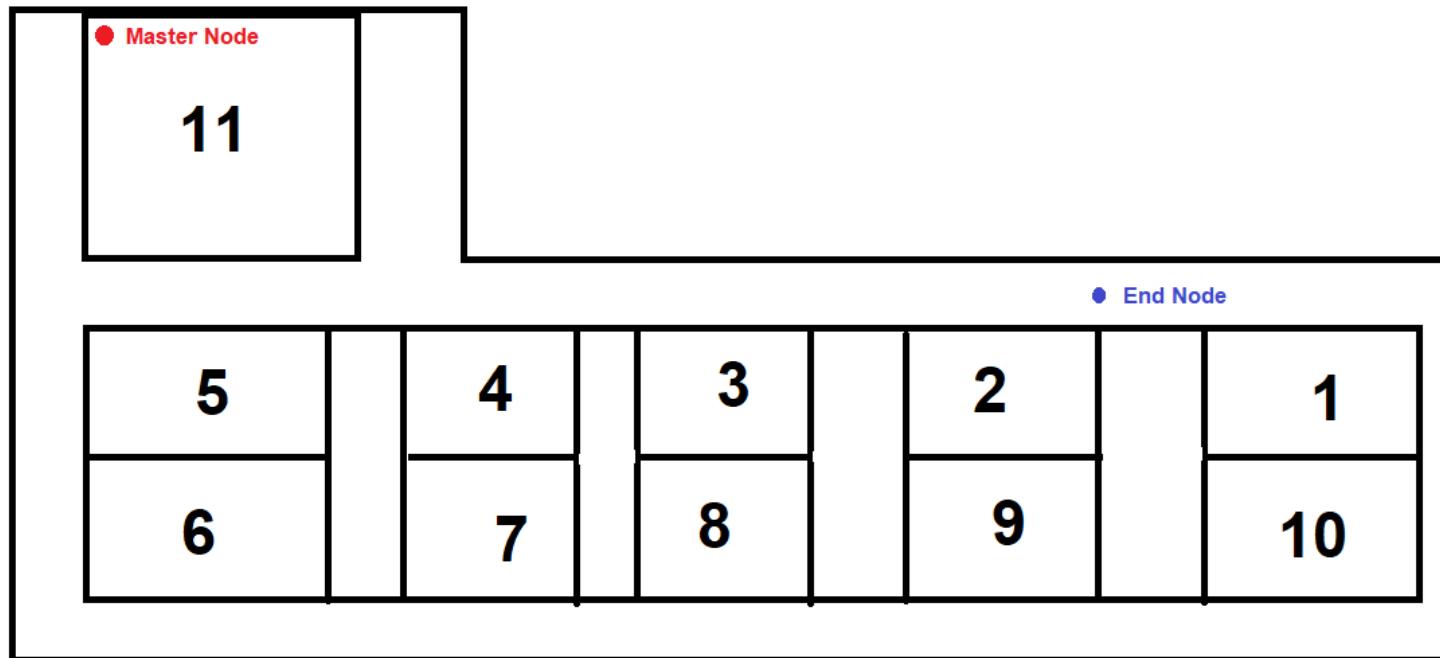
- Flood message is received
  - Add the self node ID to the path
  - Add the path as a payload
  - Send a retrace message
- Payload message is received
  - Checks if the message is intended for this master node
  - Decrypts the payload
  - Sirens the buzzer if “MOTION DETECTED” payload is received

The source and destination IDs are changed as and when required.

# Workflow



# Stress Test Results



# Demonstration

The image shows a Windows desktop with four serial monitor windows open, demonstrating communication between an Arduino Uno and a computer. The top-left window (COM5) shows a node receiving motion detection data and sending it to a master node. The top-right window (COM4) shows the master node receiving the data and sending it to a relay node. The bottom-left window (COM3) shows the relay node receiving the data and sending a response back to the master node. The bottom-right window shows the master node receiving the relay's response and displaying the decrypted payload.

**Top Left Window (COM5):**

```
01:18:17.936 -> End Node
01:18:29.181 -> Sending: F*1*Z*A*A**
01:18:33.258 -> Received: F*1*Z*M*AM**
01:18:34.323 -> Received: R*1*M*Z*AM*AMZ*
01:19:01.115 ->
01:19:01.115 -> Received: R*1*A*M*A*AMZ*
01:19:06.076 -> Encrypting the payload i.e. MOTION DETECTED
01:19:06.122 -> Encrypted the payload to PRWLRQ GHWHFWHG
01:19:06.169 -> Sending: P*1*M*A*MZ*PRWLRQ GHWHFWHG*
01:19:10.267 -> Received: P*1*Z*M*Z*PRWLRQ GHWHFWHG*
```

**Top Right Window (COM4):**

```
01:18:24.317 -> Master Node
01:18:29.198 -> Received: F*1*Z*A*A**
01:18:29.198 -> Received: F*1*Z*A*A**
01:18:33.274 -> Received: F*1*Z*M*AM**
01:18:34.247 -> Sending: R*1*M*Z*AM*AMZ*
01:18:34.339 -> Master Node
01:18:38.362 -> Received: R*1*A*M*A*AMZ*
01:19:06.169 -> Received: P*1*M*A*MZ*PRWLRQ GHWHFWHG*
01:19:17.829 ->
01:19:17.829 -> Received: P*1*Z*M*Z*PRWLRQ GHWHFWHG*
01:19:17.829 -> *** Payload received ***
01:19:17.874 -> Decrypting the payload i.e. PRWLRQ GHWHFWHG
01:19:17.920 -> Decrypted payload: MOTION DETECTED
```

**Bottom Left Window (COM3):**

```
01:18:21.004 -> Relay Node
01:18:29.228 -> Received: F*1*Z*A*A**
01:18:33.211 -> Sending: F*1*Z*M*AM**
01:18:33.304 -> Relay Node
01:18:34.323 -> Received: R*1*M*Z*AM*AMZ*
01:18:38.301 -> Sending: R*1*A*M*A*AMZ*
01:18:38.394 -> Relay Node
01:19:06.169 -> Received: P*1*M*A*MZ*PRWLRQ GHWHFWHG*
01:19:10.174 -> Sending: P*1*Z*M*Z*PRWLRQ GHWHFWHG*
01:19:10.267 -> Relay Node
```

**Bottom Right Window (COM4):**

```
files (.h) found in C:\Users\anav2\Documents\Arduino\libraries\pitches
files (.h) found in C:\Users\anav2\Documents\Arduino\libraries\pitches
```

**Windows Taskbar:**

- Type here to search
- File Explorer
- Google Chrome
- Microsoft Edge
- PowerShell
- Task View
- File Explorer
- File Explorer
- File Explorer

**System Tray:**

- 29°C Haze
- ENG
- 01-04-2022
- 6

# Future Scope for Improvement & Upgradation



- There is still scope for additions, and upgradations in the project.
- We currently use a basic, inexpensive IR sensor, which does not match capabilities of other industrial grade sensors. Better, stronger and more varied Sensors can be used.
- A better controller can be used to allow more Sensors to be used at a certain time, and more complex encryption algorithms can be run without losing precious processing time. A better antenna can also be used to improve range.
- A more complex packet based streaming protocol can be created to allow transmission of packages greater than the limit allowed on a single LoRa packet (256 Bytes).
- More robust and secure packaging can be done to decrease maintenance and increase node security.

—

# Thank you

# 6LoWPAN

Internet of Things

# Overview

- ***6LoWPAN*** is an acronym of **IPv6 over Low power Wireless Personal Area Networks**.
- A simple **low throughput** wireless network comprising typically **low cost and low power** devices.
- Common **topologies** include – star, mesh, and combinations of star and mesh.
- The Phy and MAC layers *conform* to IEEE 802.15.4-2003 standard.

## 6LoWPAN Characteristics

- Small packet size (127Byte).
- 16-bit short or IEEE 64-bit extended media access control addresses.
- Low bandwidth. (250kbps).
- Low power, typically battery operated .
- Relatively low cost.

# Protocol and Architecture

- High number of proprietary or semi-closed solutions: Zigbee, Z-Wave, Xmesh, SmartMesh/TSMP, ... at many layers (physical, MAC, L3) and most chip vendor claim to be compatible with their own standard.
- Many non-interoperable “solutions” addressing specific problems.
  - Different **Architectures**,
  - Different **Protocols**

# *Which protocol and architecture ?*

- The architecture and protocol MUST have a specific properties.
- Based on open standards: for interoperability, cost reduction and innovation ...
- Flexibility in many dimensions:
  - Support a wide range of media
  - Support a wide range of devices
- Always favor global than local optimum. ↗
- Highly secure
- Plug & Play
- Scalable

# *IP: The perfect fit !*

- Based on **open standards**: for interoperability, cost reduction and innovation
- **Flexibility** in many dimensions:
  - Support a wide range of media
  - Support a wide range of devices
- Always favor global than local optimum:
  - Highly secure
  - Plug & Play
  - Scalable
- Open standard: The Internet Engineering Task Force.
- **Flexibility** in many dimensions:
  - Serial, SDH, FR, ATM, Ethernet, Wireless, Optical ...
  - From cell phone to high speed routers
- **Always favor global than local optimum:**  
“IP if good enough for everything: from email to video to realtime protocols”
- A very secure and well proven
- **Billions of connected devices**

# IP to the Sensors

- New services and applications
  - M2M, remote management.
- New Markets
  - Process Control for factories.
  - Control and Automation for home, building, cities.
- Larger Core Market
  - Open standards to the sensor
    - Lower cost
    - More connected devices and new applications
    - A wider Internet
- Shaping the future
  - Internet of things.

# Why IPv6?

- **Advantage**

- More suitable for higher density Statelessness mandated.
- No NAT necessary.
- Possibility of adding innovative techniques such as location aware addressing.

- **Defect**

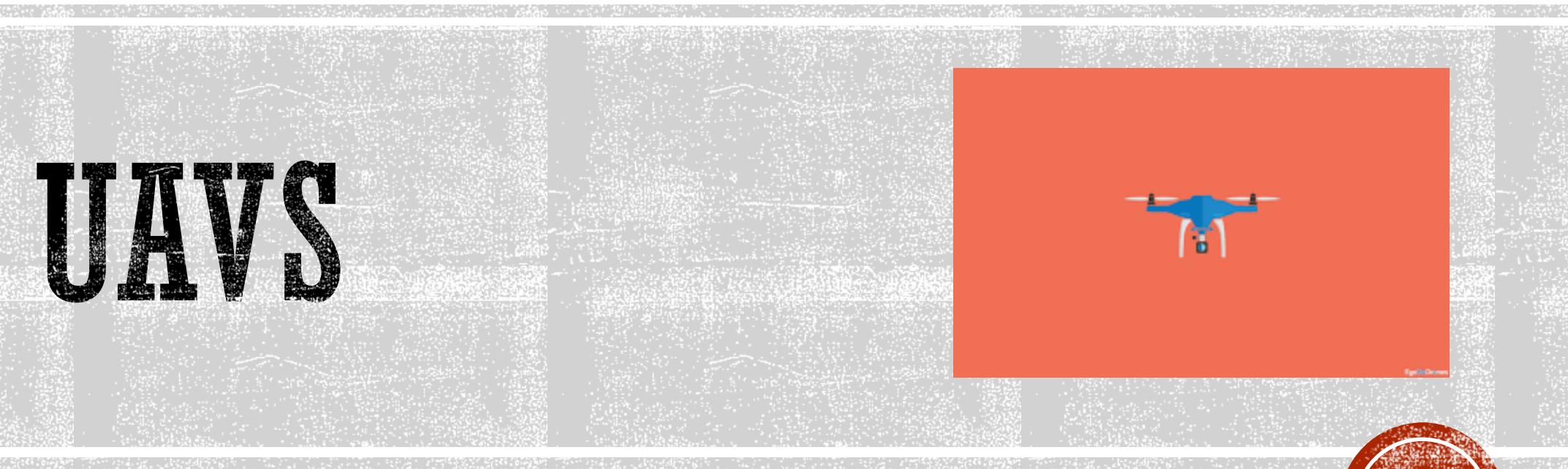
- Larger address width(128bit).
- Complying to IPv6 node requirements.

# Key Factors for IPv6 over 802.15.4

- Header
  - Standard IPv6 header is 40 bytes [RFC 2460]
  - Entire 802.15.4 MTU is 127 bytes [IEEE ]
  - Often data payload is small
- Fragmentation
  - Interoperability means that applications need not know the constraints of physical links that might carry their packets
  - IP packets may be large, compared to 802.15.4 max frame size
  - IPv6 requires all links support 1280 byte packets [RFC 2460]
- Allow link-layer mesh routing under IP topology
  - 802.15.4 subnets may utilize multiple radio hops per IP hop
  - Similar to LAN switching within IP routing domain in Ethernet
- Allow IP routing over a mesh of 802.15.4 nodes
  - Options and capabilities already well-defined
  - Various protocols to establish routing tables
- Energy calculations and 6LoWPAN impact

# Conclusion

- 6LoWPAN turns IEEE 802.15.4 into the next IP-enabled link
- Provides open-systems based interoperability among low-power devices over IEEE 802.15.4
- Provides interoperability between low-power devices and existing IP devices, using standard routing techniques
- Paves the way for further standardization of communication functions among low-power IEEE 802.15.4 devices
- Great ability to work within the resource constraints of low-power, low-memory, low-bandwidth devices like WSN

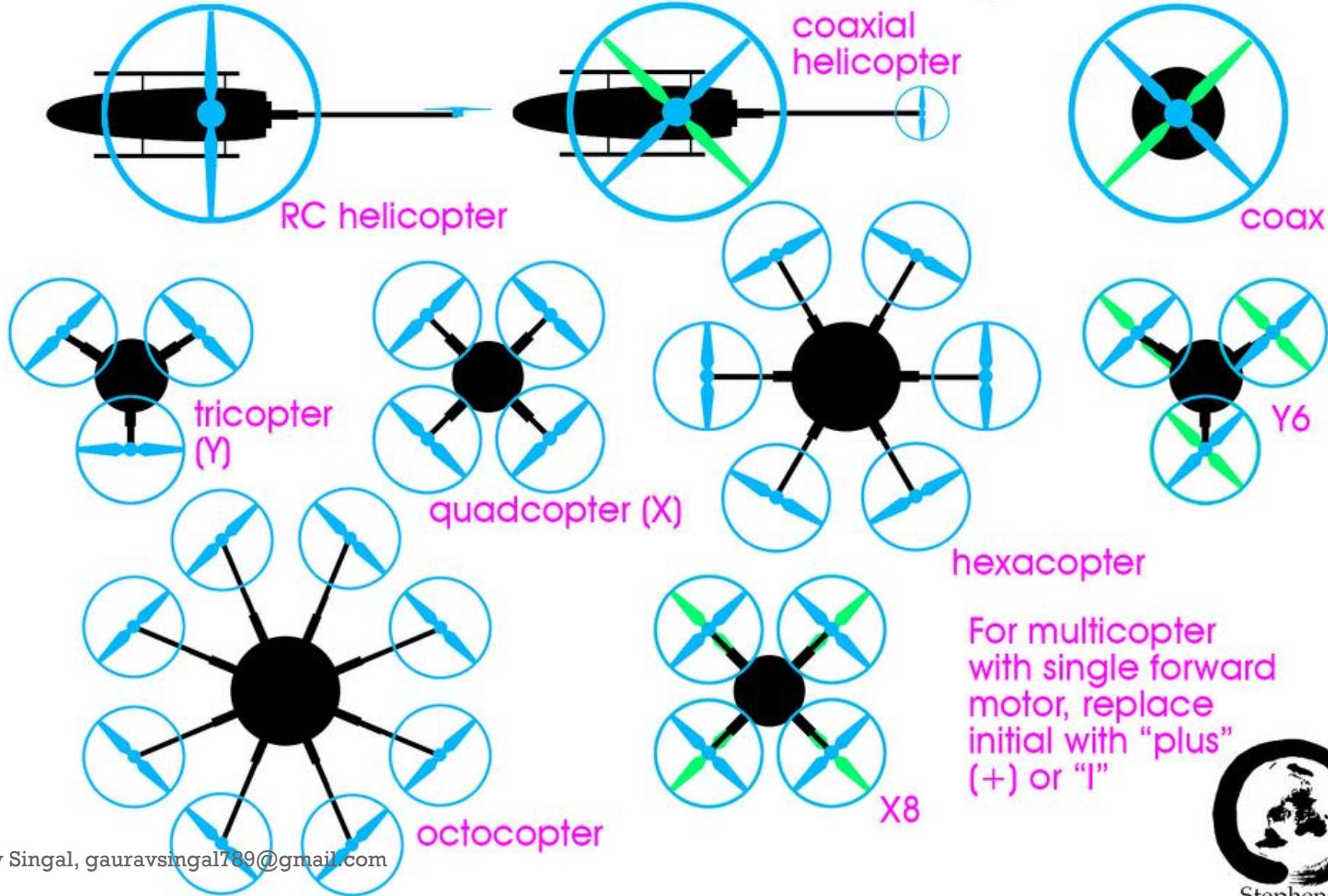


**BENNETT**  
**UNIVERSITY**  
A TIMES GROUP INITIATIVE

# INTRODUCTION TO DRONE

- A drone is an unmanned aircraft.
- Drones are more formally known as unmanned aerial vehicles (UAVs) or unmanned aircraft systems (UAS).
- A drone is a flying robot.
- The aircrafts may be remotely controlled or can fly autonomously through software-controlled flight plans in their embedded systems working in conjunction with onboard sensors and GPS.

# Drone Typology



# WHY HAVE WE CHOSEN QUADCOPTER???

Mainly we choose quadcopter due to it's mechanical simplicity even though it has a few disadvantages like stability and efficiency.

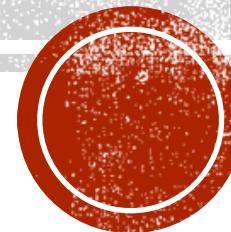
5



## The Tech Behind a Record-breaking Drone Show at PyeongChang 2018

By Intel

# QUADCOPTER



# INTRODUCTION TO QUADCOPTER

- The quadcopter is the **first and most popular drone** you can find in the market.
- This is an option that **uses 4 rotors** organized in a **square pattern**.
- The rotors are arranged onto **each individual corner** of the quadcopter's body.
- It allows for an even amount of support to get the drone **off the ground** and to make it **turn** in different directions.
- Quadcopter is an **aerial vehicle** which is operated to fly independently.
- It is a small representation of UAV. It is classified as **rotorcraft**.
- Quadcopters are **cheaper** and more **durable** than conventional helicopters.

# ROTORQUIZ#1

***Aircrafts have commonly been named after what???***



# INSECTS...

## History

- **Gipsy Moth**, one of the most common aircrafts in UK in 1929
- 1935 - Put radio controls on a **de Havilland Tiger Moth**, a successor to the Gipsy Moth
- **The Queen Bee (DH.82B)** was one of the first returnable and reusable UAV - used as practice targets.

# USES (PICTURES/VIDEO/GIF)

- Effective for Surveillance and Security.
- Research Purposes (robotics, flight control, etc.).
- Military, law enforcement and community agencies.



11



## Asking the Right Questions: Applications of Drone Technology

### Mrinal Pai | TEDxSIBMBengaluru



- Commercial use and aerial photography
  - Augmented reality games
- For delivering food and Medicine to affected places
  - Geographical Calculations





## TU DELFT – AMBULANCE DRONE

# ROTORQUIZ#2

Which was the first country to build drones???

- A little hint:



# **ISRAEL...**

- The first country to manufacture drones was Israel.
- Israel Aerospace Industries has production facilities about 24 countries around the world.

# ROTORQUIZ#3

**Which country are the highest users of drones ???**

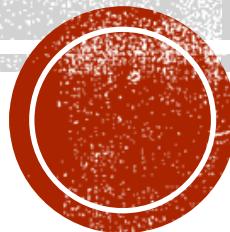
# AMERICA

- Over 181000 Americans have registered their drones with the Federal Aviation Administration of America.
- Stores report selling over 400000 drones.
- There are many unregistered American drones.
- Failing to register an American drone can lead to a fine of \$27500.

# APPLICATIONS



# **COMPONENTS OF QUADCOPTER**

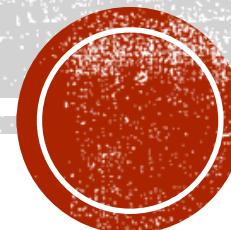


# DRONE COMPONENTS

- Frame
- Motors
- Electronic speed Controller (ESC)
- Flight Controller
- Communication
- Propellers
- Battery



# HOW TO CHOOSE A FRAME



# Purpose Decides the frame type.

# **TYPES OF DRONE**

- **Aerial**
  - Used for cinematography.
  - High Definition Camera.
  - Strong and solid frame.
- **Racing**
  - Used for competing with others.
  - Light weight for high speed.
- **Freestyle**
  - It typically involves you are flying as hobby.
  - Light weight.

# MATERIAL

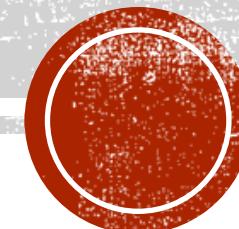
- Carbon
- Aluminium
- Fiberglass

# OTHER FACTORS

- Weight
- Size
- Layout
- Stiffness
- Price
- Review



# HOW TO CHOOSE A MOTOR



# IMPORTANT POINTS

- Estimate the weight of the drone.
- Identify the frame.
- You will get an idea what motor and propeller to use.
- Most of them will suggest you while buying the frame.

# THRUST TO WEIGHT RATIO

- It is important that your motors can produce around 50% more thrust than the total weight of your drone.
- Extra thrust will help you in wind and during aggressive flight maneuvers
- For example if your drone weighs around 600g then your motors need to produce 1.2 kg of thrust.

# KV RATING

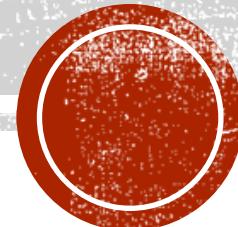
- kV rating tells us at what rpm a motor will spin at full throttle, at a certain voltage.

$$\text{RPM} = \text{KV} * \text{voltage}$$

- If you need high performance drone then you need a high kV rating.
- If you need last longer and more stable then you need to power with least amount, so low rpm are ideal.

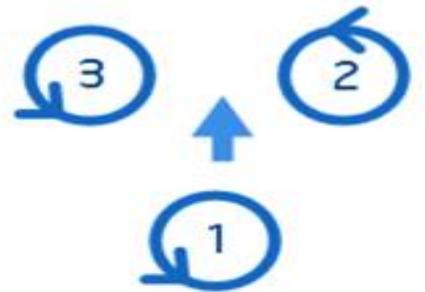


# HOW TO CHOOSE PROPELLER

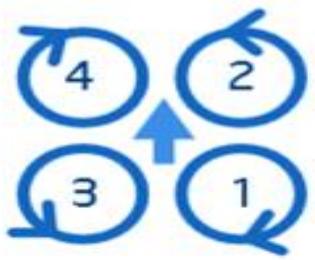


# BASIC INFO

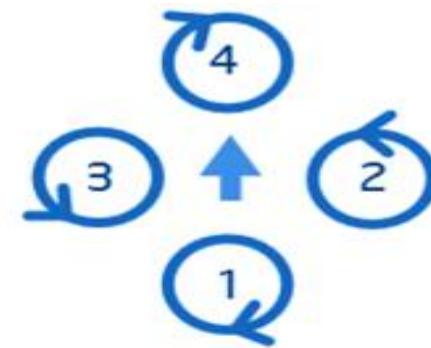
- Propellers are used to generate thrust and torque to keep your drone flying and maneuver.
- The faster they spin more thrust they produce.
- Torque is produced when they move up or down.
- To balance the torque all the propeller do not rotate in same direction.
- The length of propeller is measured from one tip to another tip.
- Pitch is the distance travelled by the propeller in a single rotation.
- Generally, the frame will give you an idea.



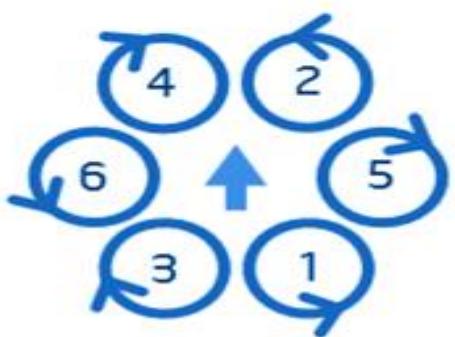
TriCopter



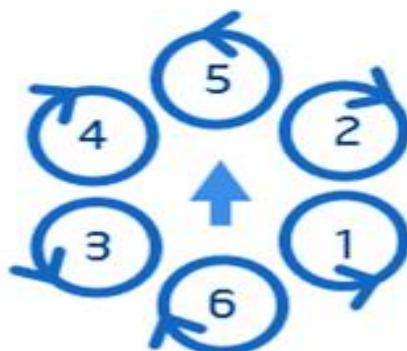
QuadCopter-X



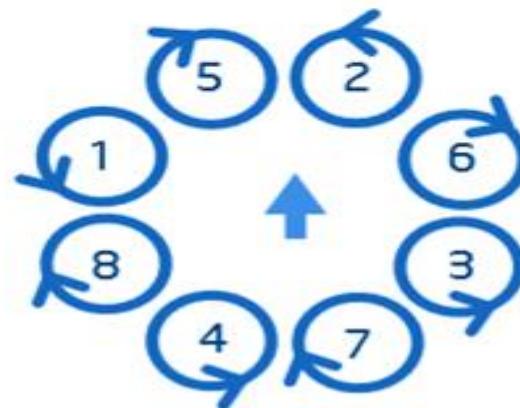
QuadCopter-Plus



Hexa-X



Hexa-Plus



Octo-X

# **IMPACT OF SIZE AND PITCH**

- Propeller with smaller size will be easy to stop and speedup.
- Big size propeller will take time.
- Low pitch propeller will consume less power from the system and this will have positive impact on steadiness. But it will provide low speed.
- High pitch propeller will draw more power from the system and this can have negative impact on steadiness. But it will provide high speed.

# MATERIAL

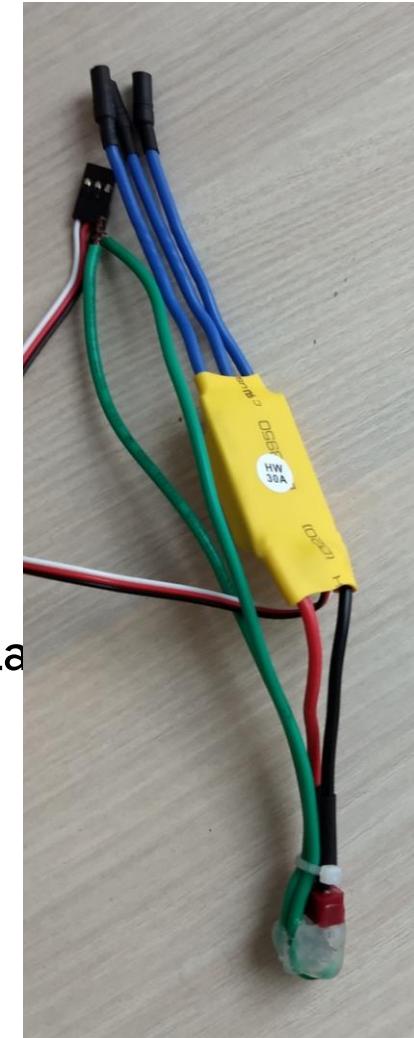
- It consist of materials like carbon, wood, plastic etc.
- Carbon and wood propeller are bit hard and provided smooth flight.
- Plastic propeller are more durable and reliable.

# OTHER FACTORS

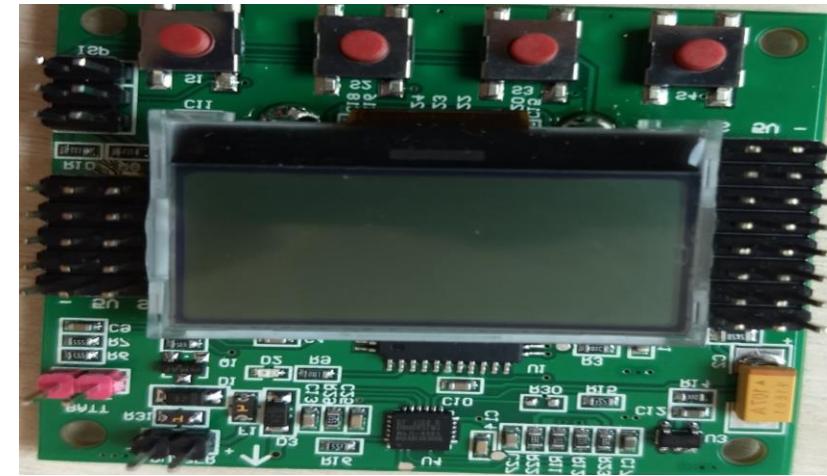
- Shape
  - it impacts on thrust.
- Number of blades
  - Generally dual blade propellers are used

# ESC

- ESC stand for electronic speed control.
- They are used to control motor speed.
- ESC receives throttle signal from flight controller and drives the motor at that speed.
- It tells us the amount of ampere it provide to the motor.
- ESC also tells us which battery it supports.



# FLIGHT CONTROLLER



- Flight controller is a circuit board with sensors that detect the orientation change.
- It receives the user commands.
- You need to choose flight controller according to the purpose of your drone.
- Few sensors will be attached to flight controller and some of them you need to buy.

# BATTERY

- LIPO battery is used for flying drone.
- Battery depends on size of drone and type of motors you use.
- Check the battery discharge rate
- Check the review of your battery.
- Buy the charger for your battery accordingly.
- Take great care of your battery as it is very sensitive and costly.

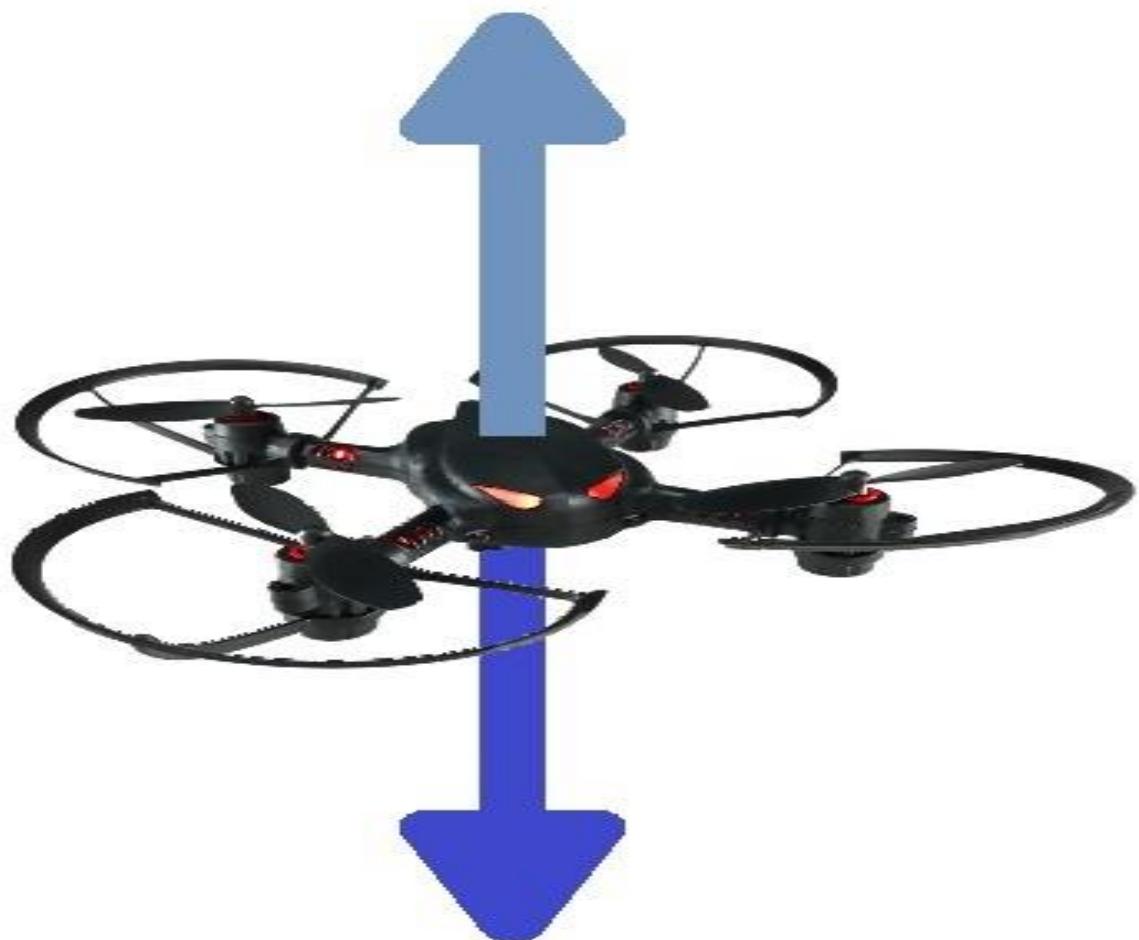
# DIFFERENT TYPES OF SENSOR

- Accelerometer
  - It is used for measuring the linear acceleration in three axes(X,Y,Z).it is very important as it tells us the direction up and down.
- Gyroscope
  - It is the most important sensor as it tells about the orientation of the drone, and hence the correction required to make it horizontal.
- Barometer
  - It is used to measure air pressure and also we can measure altitude.

# DETECTION OF ORIENTATION OF DRONE

- Throttle
  - Controls the drone vertical and down motion.
- Yaw
  - It is left and right rotation of drone.
- Pitch
  - Forward and backward movement.
- Roll
  - It controls the side to side tilt.

**+ THROTTLE**

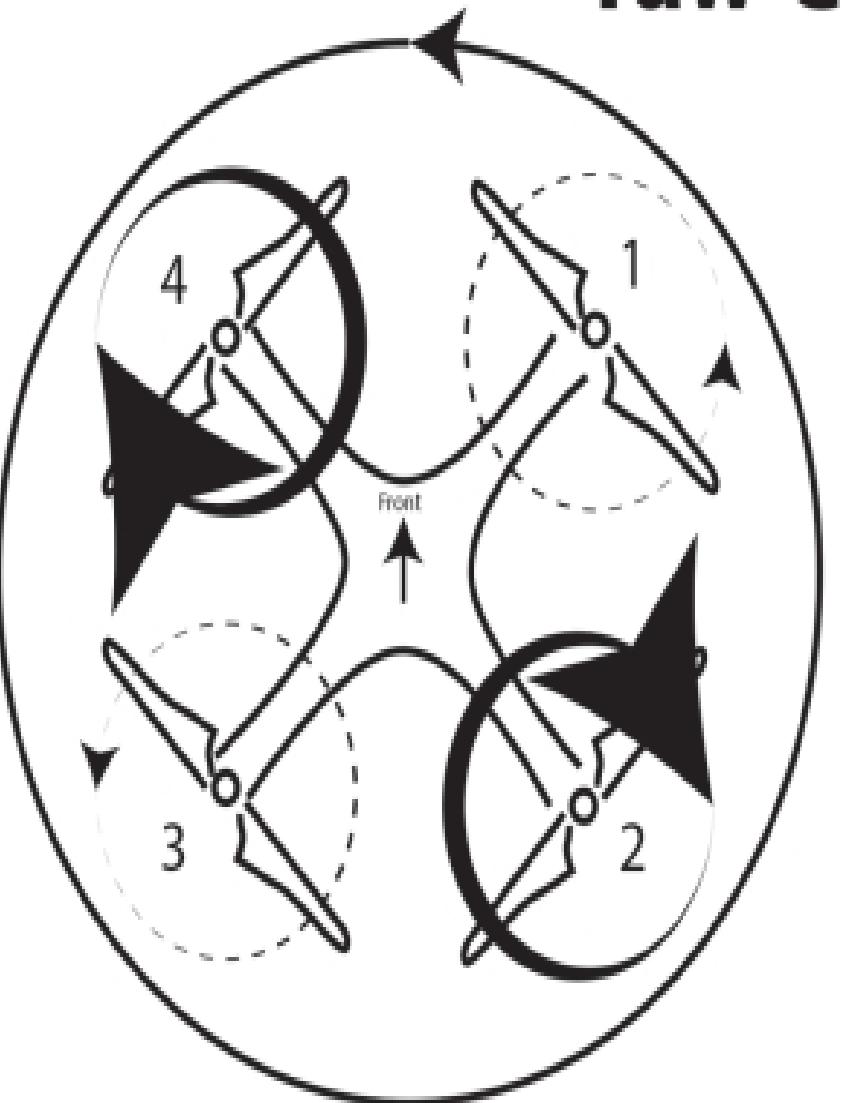


**- THROTTLE**

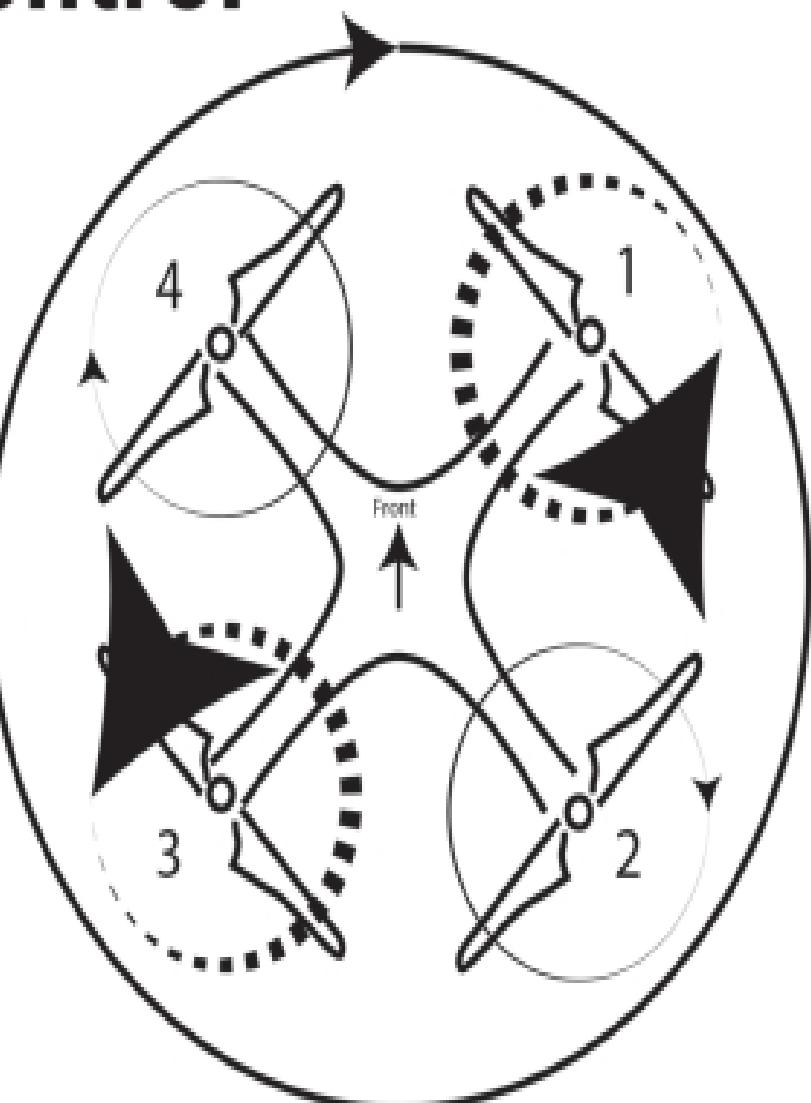
## **THROTTLE**

Controls the vertical motion.

# Yaw Control



Left Turn



Right Turn

**YAW**

It controls left and right rotation.

Pitch Up



(move backward)

Pitch Down

(move forward)

## PITCH

Forward and backward movement.

Roll Left

Roll Right



(move left)

(move right)

## ROLL

it controls the side to side tilt.

46

# COMMUNICATION

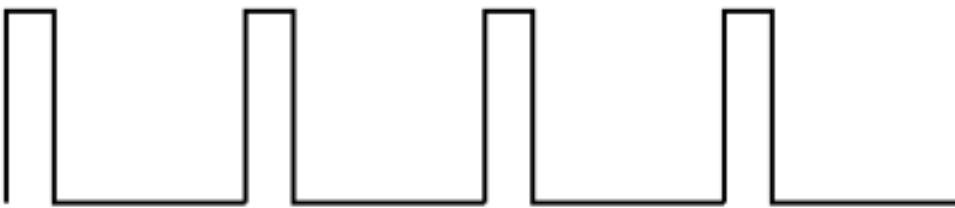
# DIFFERENT MODE OF COMMUNICATION

- Radio Control
  - It involves RC transmitter and receiver and need minimum four channels like pitch, roll, yaw and throttle.
- WiFi
  - It is being implemented through smartphone, computer or wifi router and range is limited.
- Infrared
  - This method is not suggested as there are lot of interference while flying and it is not reliable.
- Bluetooth
- Radio Frequency

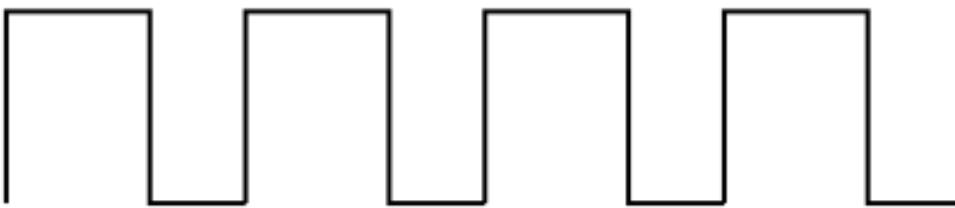
# PWM SIGNALS

- PWM stands for Pulse Width Modulation
- A square wave is created by ON and OFF pattern by alternating between 0 and 5V.
- The duration of ON time is called pulse width.
- A call to `analogWrite()` is on a scale of 0-255, hence duty cycle of 100% is 255, 50% is 127 and so on.

Duty Cycle: 20%



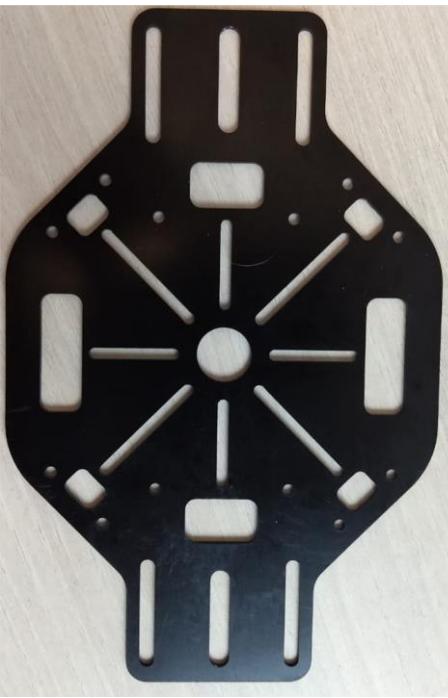
Duty Cycle: 60%



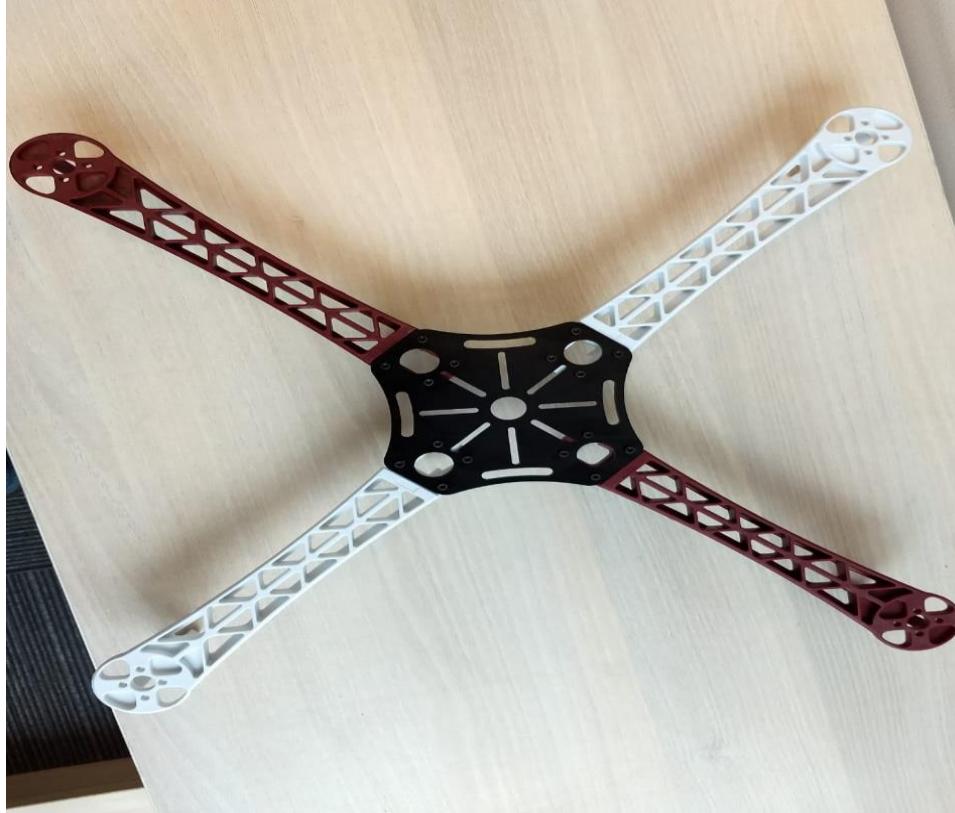
49

# ASSEMBLING A DRONE!!

We start initially with these parts at hand..



Assembling the frame components and putting those pieces together we get..



After having the basic structure ready, we are ready to mount the esc and motor.



# **CAUTION**

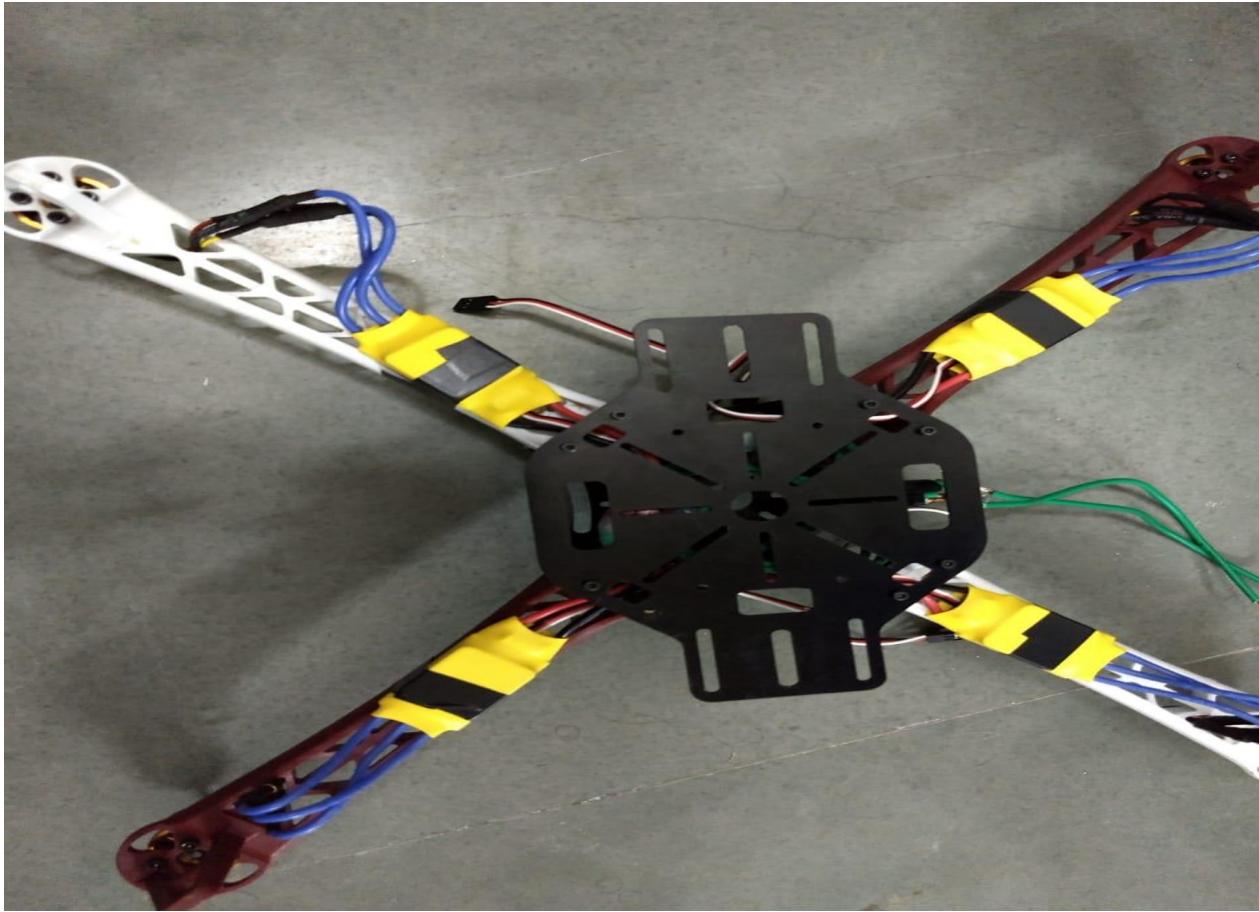
**It must be ensured that all screws are tightened well. As spinning of motors along equipped with propellers might generate sufficient torque to separate the motor from the frame.**

# **FIXING THE ESCS**

- Three bullet connectors from the motor are to be connected to the other three bullet connectors on ESC.
- It must be noted that to conserve angular momentum in drones two propellers spin clockwise and other two spin anti-clockwise.
- Opposite propellers always have the same spin.

- Hence while connecting ESCs we have to make sure that the positive terminal of the motor is connected to the positive terminal of ESC, and the negative to the negative, for a clockwise configuration.
- The positive and negative wires of motor are swapped in case of anti-clockwise configuration.
- The middle wire from the ESCs always connects to the middle wire of the motor, in both clockwise and anti-clockwise configuration.

# NOW WE HAVE THIS..



# PROCESSING UNIT OF OUR QUADCOPTER!!



Flight controller processes all the data received from the sensor and keeps the drone horizontal by providing error-correction mechanism.

# KNOW YOUR FLIGHT CONTROLLER!

- We have used a KK2.1.5 flight controller board.
- This board comes with an LCD screen that enable us to configure settings such as PID and calibration without connecting to a computer.
- The board also has self-level mode which can be helpful for beginners to fly the quadcopter.

# OTHER FLIGHT CONTROLLERS

- KK2 is a simple flight controller board with minimal features.
- Other flight controller such as ardupilot APM comes with GPS, telemetry options, etc.
- KK2 is simple to use and gives a stable flight.

# SETTING UP THE CIRCUIT

- Circuit building is a crucial part for implementing a quadcopter.
- The right side of the flight controller board has to be connected the ESCs. Notice the thin (black, red, white) wires from the ESCs.
- The white wire is the signal pin and has to be connected in the line marked with 's' on the flight controller board. The second and third pin are to be connected with red and black respectively. Innermost pin is the signal pin.
- ESCs are connected in order. Therefore ESC of motor 1(clockwise spinning) goes in the first set of three pins. The second ESC of motor 2(spinning anti-clockwise) goes to second set of pins and so on.



# RECEIVER



- Receiver receives the values of aileron, elevator, throttle, rudder from the transmitter and passes it on to flight controller.
- Connect the first set of three pins on the receiver to the first three pin set on the left side of flight controller, keeping in mind the signal pin.
- Do the same for all the four channels on the receiver.

# CONNECTING THE BATTERY!



- We use a 11.1V battery capable of dissipating current at high rates.
- All the red wires of the ESCs are connected and grouped and similarly all the black wires of the ESCs are connected and grouped.
- These grouped wires are then connected to a T-connector.
- This T connector will go to the battery when Quadcopter is to be switched on.
- Make sure the positive terminal connects to the positive of the battery and negative to the negative.

# CALIBRATING THE MOTORS

- When running the ESCs for the first time we need to specify at what input signal it should run at highest speed and at what input signal at the lowest speed. This process is called calibration.
- Our flight controller board has an inbuilt calibration mode, for calibrating all the ESCs together.

# CONT..

- To calibrate the motors switch on the transmitter and put the throttle stick at extreme upper end.
- Now hold the first and last buttons (red) on the flight controller which you connect the battery.
- The ESC should beep.
- When the beeps stop bring the throttle stick down, and the ESC should beep again.
- Once this is done our ESCs are calibrated.
- Bring the throttle stick up to see the motors running.

# **CAUTION!!**

**Always remove the propellers before calibrating.**

# ACCELEROMETER CALIBRATION

- Our quadcopter needs to be told what is the horizontal level that it should maintain, this information should sync with the gyroscope sensor.
- To calibrate just put the drone on levelled surface and switch it on.
- Now Go to Menu-> ACC calibration.
- It will ask for confirmation and then you can proceed.
- It will be calibrated automatically.

# **PUTTING ON PROPELLERS**

- There are two types of propellers in the set, one is for clockwise and one is for anti-clockwise spinning motor.
- A clockwise spinning propeller has a bulge on its top right side.
- Put similar propellers on the diagonally opposite side.

# FLYING QUADCOPTER

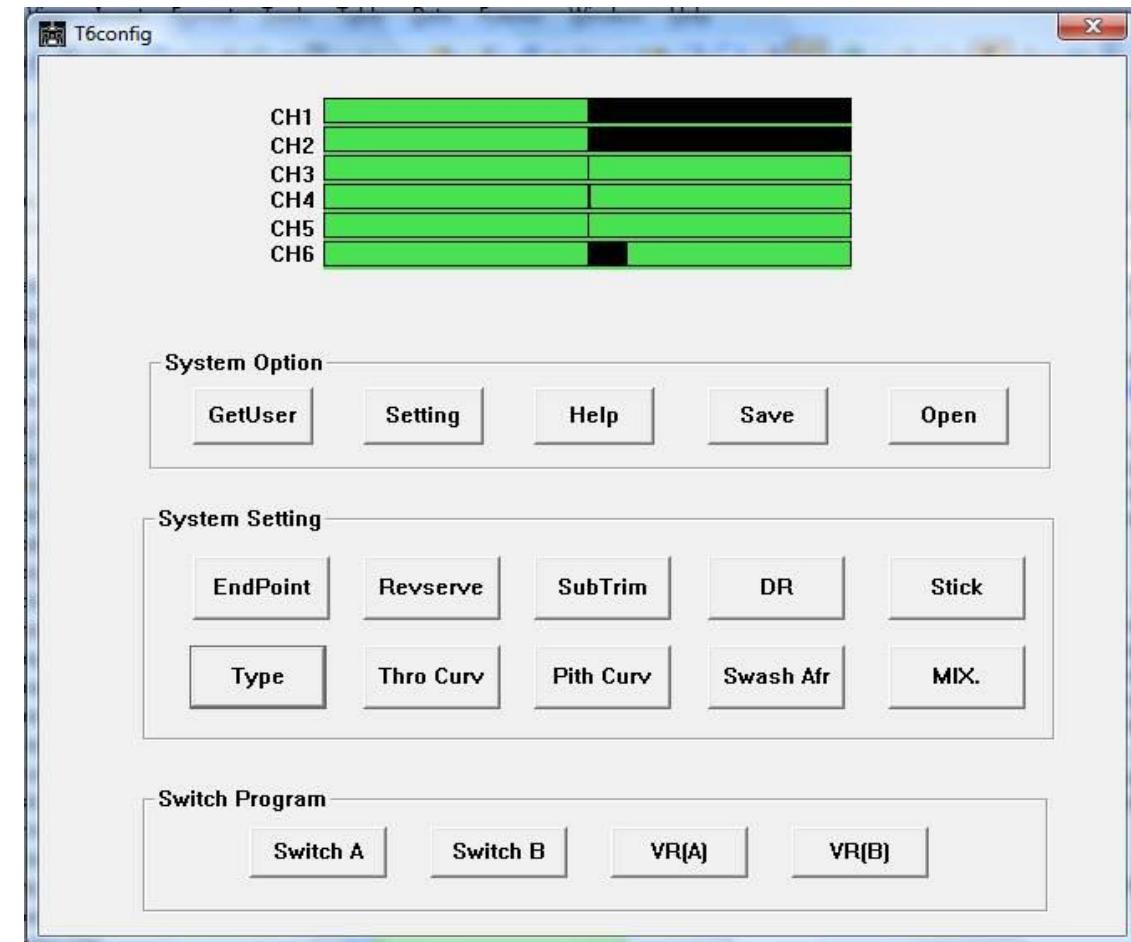
- Always fly your quadcopter outdoors.
- The propellers attached are dangerous.
- Always check if the quadcopter is configured properly.

# ARMING THE QUADCOPTER!

- Arming is the procedure which enables the motors to spin, and indicates that the user is ready to fly.
- Always arm the quadcopter once you are on a safe distance from it.
- To arm the quadcopter pull the first stick of the transmitter to the extreme low and to the extreme left.
- A red light on the flight controller board indicated that the quadcopter is armed.
- The flight controller board also has ARMED written on center of its screen.

# TRANSMITTER CONFIGURATION

- The FS-CT6B transmitter comes with a CD carrying the configuration program and driver.
- Install the drivers.
- The configuration program (T6Config) can be used to adjust the transmitter values and reverse channels.
- The modes (or) layout of transmitter channels can also be changed.
- It is useful at times to reverse the pitch to get better intuitive control.



# PID TUNING

- P is Proportion, I is Integral and D is Derivative.
- PID tuning is very important as it determines how responsive the drone is to the controls.
- Responsiveness decides the experience of the flight.
- The need for responsiveness also depends on the purpose of flight.
- A drone racer (person) would need more responsiveness while a photographer would need less responsiveness.

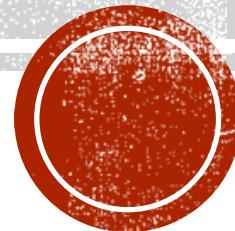
# PID TUNING

- Increase p gain if the response to command is too slow, but if on changing the command, the Quadcopter oscillates with respect to that command then decrease p gain.
- For example in case of yaw, if the Quadcopter oscillates around the axis decrease p gain, else if the response in moving only about the axis slow, decrease it.
- While PID tuning consider moving the Quadcopter only in the movement of command being tuned.

# PID TUNING

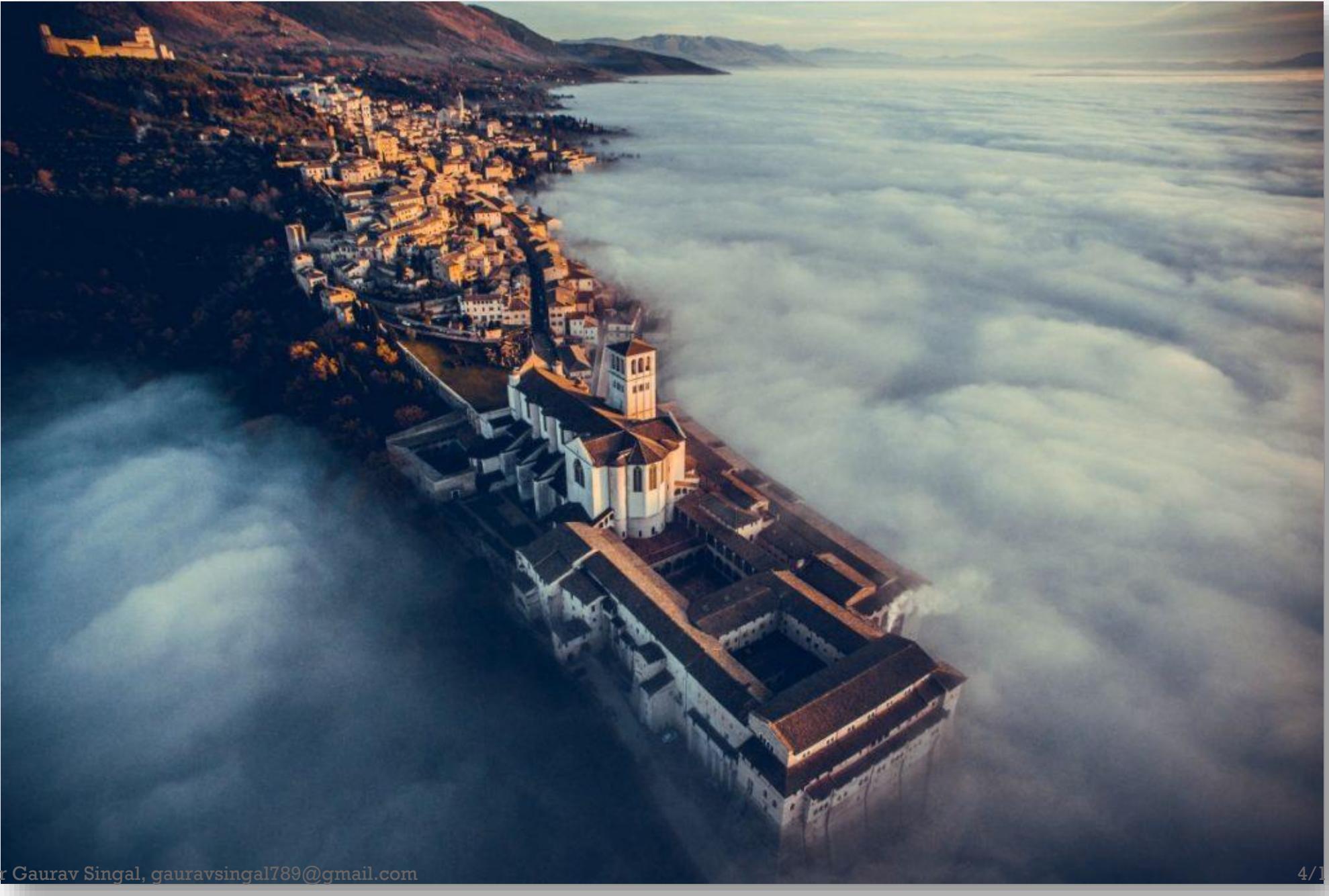


# APPLICATIONS



# AERIAL IMAGING

- UAVs are capable of capturing images and video at thrilling angles, creating effects that are appealing to people.
- Images and videos captured by drones are post-processed.
- Post-processing includes stitching of images so as to make them look as if they were taken from uniform altitude, pitch, roll, and yaw.
- Videos captured with drones are usually noisy, due to the vibration. However post-processing makes them better.

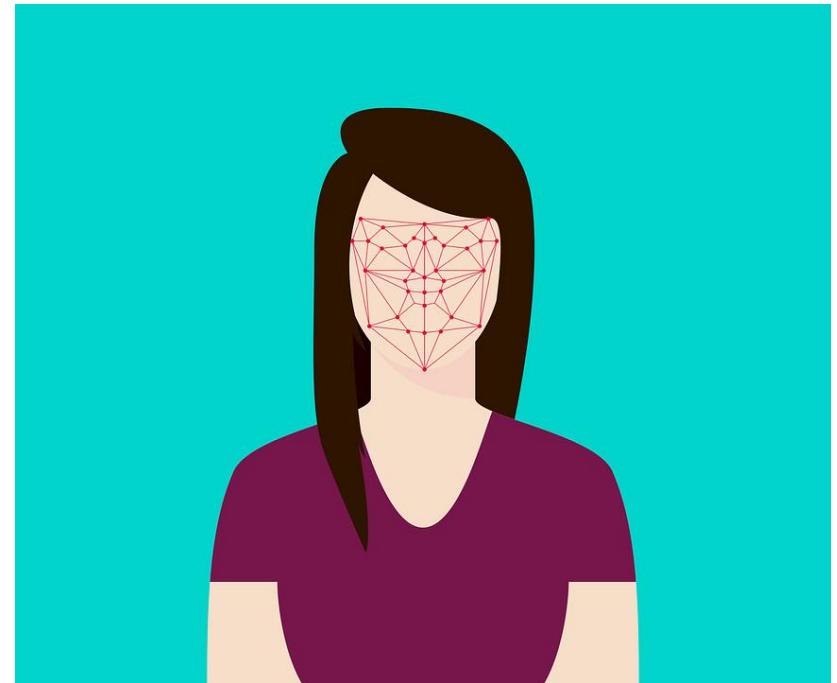


# **SURVEILLANCE**

- Recently, drones have become quite popular in surveillance.
- Drones can often survey subjects that are not accessible to otherwise and can offer FPV(First-Person View).
- First-Person View is the ability to view from different visual perspectives.
- Drones combined with computer vision, object recognition and face recognition can essentially bring surveillance to another level.

# WHY FACIAL RECOGNITION WITH DRONE?

- Facial recognition with drones can help in identifying subjects on whom the surveillance is being done.
- It can help in identifying photos of people carrying out robberies, by matching the frames in video to photos in crime records.
- It enables us to identify the person so that he can be tracked by the drone.

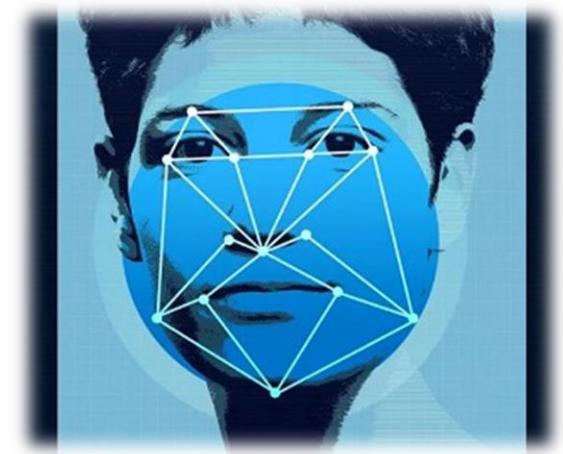


# ABOUT FACIAL RECOGNITION

- Facial recognition can be divided into face detection and identification.
- Face identification can be done in several ways such as using non deep learning algorithms like eigenfaces and LBPH.
- However, deep learning based methods give extremely higher accuracy and even comparable to human vision on several datasets.

# METHOD

- Our network architecture for face recognition is based on ResNet-34 from the *Deep Residual Learning for Image Recognition* paper by He et al., but with fewer layers and the number of filters reduced by half.
- We use dlib library of python.
- The frame is processed to detect face.
- The face is then pre-processed and fed into the trained model which generates a 128-d embedding.
- If Euclidean distance between these embedding of two different faces is below a threshold then we say it is a match.
- The threshold values in our model is 0.4.



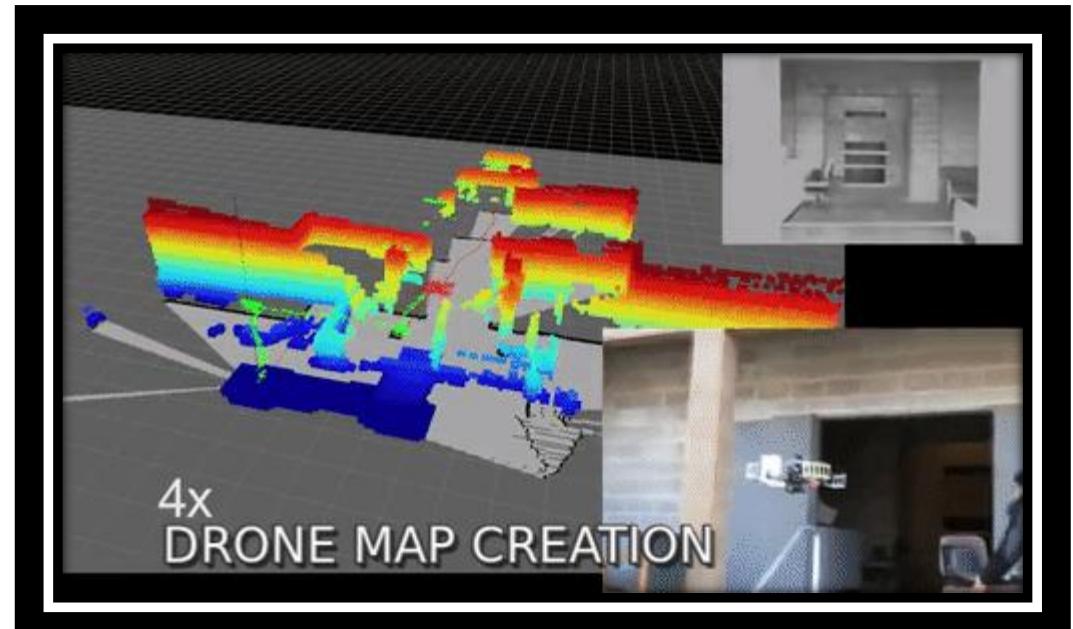
# FARMING

- Drones can prove very useful in farming, specially in India where farming is an essential activity.
- Spraying of pesticides and other harmful chemicals by farmers can affect their health seriously.
- Determining crop health using computer vision techniques.
- Determining crop variations, etc.



# SLAM

- We are currently working on implementation of SLAM at our university.
- SLAM stands for Simultaneous Localization and Mapping.
- Simultaneous localization is the computational problem of updating the map of a place while keeping the current location of drone.
- SLAM can map the entire location, using camera or sensors like LIDAR(Light Detection and Ranging).

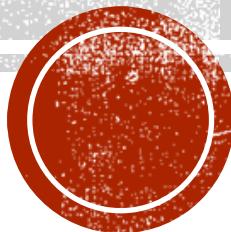


# OTHER APPLICATIONS

- Condition survey, detecting leaks.
- Rescue missions on tough terrains.
- Logistics(Amazon delivery drone).
- Managing security threats in battle grounds.
- Eliminating terrorism.

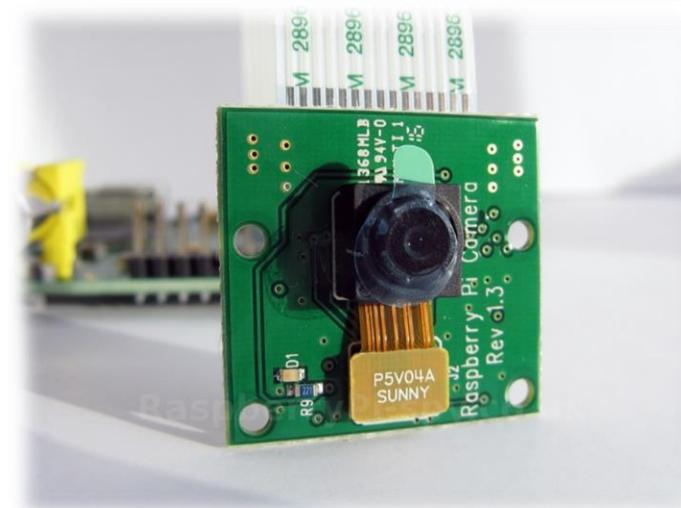


# **IMPLEMENTING APPLICATIONS**



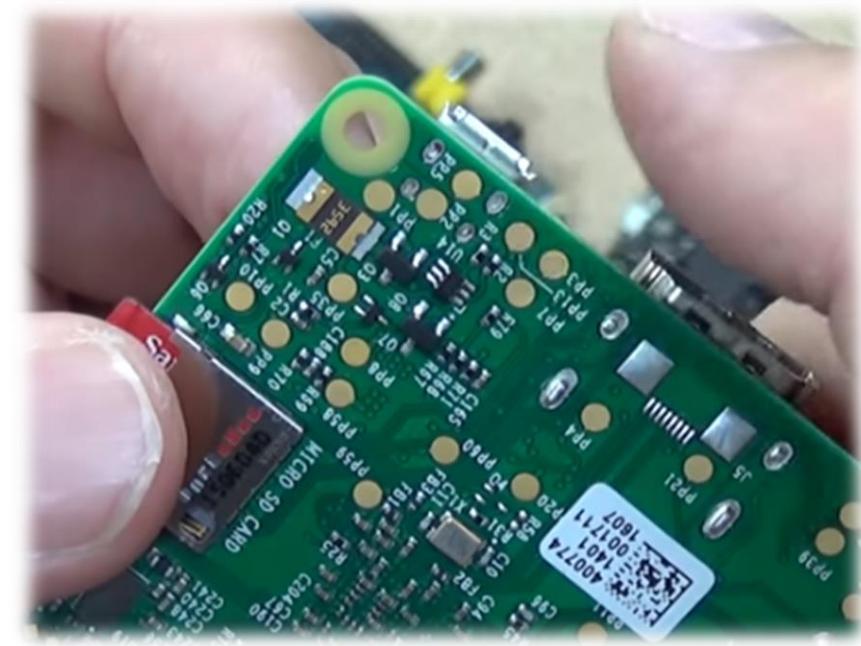
# REQUIREMENTS

- Raspberry Pi.
- Pi Camera.
- DC-DC voltage buck converter.
- Other sensors specific to application.



# POWERING RASPBERRY PI

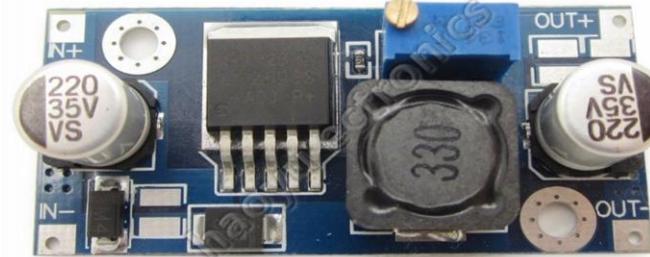
- Raspberry pi can be powered in different ways:
  - Through USB cable
  - Through GPIO pins
  - Supplying 5V directly through pads below USB input.



# THROUGH GPIO PINS

- To power raspberry pi from GPIO pins, the positive and negative terminals from Li-Po go to IN+ and IN- terminals respectively marked on buck converter.
- Fix voltmeter on OUT+ and OUT- and rotate the screw until you get 5V on voltmeter.
- Remove the voltmeter and connect OUT+ to pin 2 and OUT - to pin 6 of raspberry pi.

Dr Gaurav Singal, gauravsingal789@gmail.com



Raspberry Pi 3 GPIO Header

Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I <sup>2</sup> C)	DC Power 5v	04
05	GPIO03 (SCL1 , I <sup>2</sup> C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I <sup>2</sup> C ID EEPROM)	(I <sup>2</sup> C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

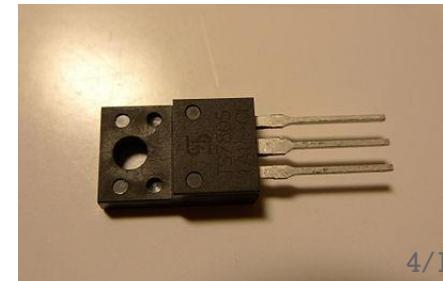
# BUCK CONVERTER

- It is used to step down the voltage, and get constant voltage output which is required for stable functioning of raspberry pi.
- We have used LM2596, which gives a current output of 3A.
- The Li-Po battery used provides an input of 11.1V which is brought down to 5V.



# BUCK CONVERTER V/S LINEAR VOLTAGE REGULATOR

- Linear voltage regulators can be used in place of buck converter.
- Linear voltage regulators are simple to use, and cheaper.
- However, they lose a lot of energy in form of heat.
- Hence it is suggested to use buck converters.

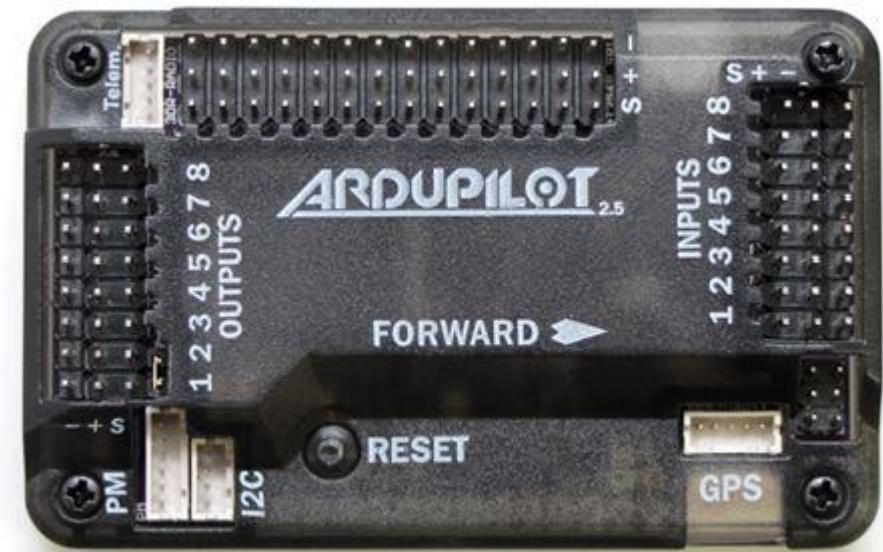


# COMMUNICATING WITH RASPBERRY PI

- VNC server can be used for accessing raspberry pi.
- To use VNC server create a hotspot and connect raspberry pi to it.
- To send data to raspberry pi socket programming can be used.
- As programming can be done in python, sending data using socket programming becomes easier.

# COMMUNICATING WITH FLIGHT CONTROLLER

- Generally flight controller gets input from radio receiver.
- For automating drone the input such as throttle, yaw, pitch, roll can be calculated using a computer vision algorithm and then sent to the drone via raspberry pi.
- Flight controllers like ardupilot APM support serial input.



# COMMUNICATING WITH FLIGHT CONTROLLER

- For Flight controllers such as KK2 which do not have a serial input, the pi output must be converted to PWM signals.
- As raspberry pi does not have 4 hardware PWM pins, we cannot get stable PWM signals.
- This problem can be solved using an intermediate Arduino which converts the serial input from pi to PWM signals and gives stable input to KK2 board.

# ADDITIONS TO DRONE

- **GPS:**
  - Global Positioning System module can be used with flight controller to get GPS coordinates of drone. This enables us to move the drone autonomously, or to plan the flights rather than using a transmitter.
- **Ultrasonic sensor:**
  - Ultrasonic sensors detect distance from respective object by sending a sound wave of specific frequency.
  - Ultrasonic sensors are very useful in avoiding collisions of drone in cases of autonomous flights. They detect obstacles and provide information to move accordingly.

# FINAL STEP

- The last step is to write code specific to the application.
- Raspberry pi can be programmed in python, and PiCamera and GPIO libraries help in accessing camera stream and writing GPIO outputs.

# TIPS AND TRICKS

- Always keep a voltmeter handy, it is useful in several ways:
  - It can be used to check if proper voltage is being applied at specific places.
  - It can be used to check connectivity in connections, helpful in detecting broken wires and faulty jumpers.
- Have an Arduino programmed which can give PWM reading.
- Calibrate your drone at regular intervals, motor and level calibrations determine flight experience.
- Try not draining Li-Po battery completely, dispose them if you see a swelling.
- Do not leave the connections loose, as they can come out easily while a drone is flying.

# DGCA RULES AND REGULATIONS

- DGCA stands for Directorate General of Civil Aviation.
- New policy is to come in effect from 1<sup>st</sup> December 2018.
- *The DGCA has segregated drones into five different categories*
  - Nano : Less than or equal to 250 grams.
  - Micro : From 250 grams to 2kg.
  - Small : From 2kg to 25kg.
  - Medium : From 25kg to 150kg.
  - Large : Greater than 150kg.

# DGCA RULES AND REGULATIONS

- Seven day prior permission is required to hold a drone flying event.
- Exceptions:
  - Drones of size nano and micro do not need seven days prior information.
  - Height should not reach more than 60 m for micro drones.

# DGCA RULES AND REGULATIONS



---

# Thank You!!



# IoT Security

# Agenda

Introduction to Security in IoT Devices

IoT Trends

Security Goals

Security Services

Security Mechanisms

IoT Vulnerabilities

Types of Attacks in IoT environment

IoT attack Detection Mechanisms

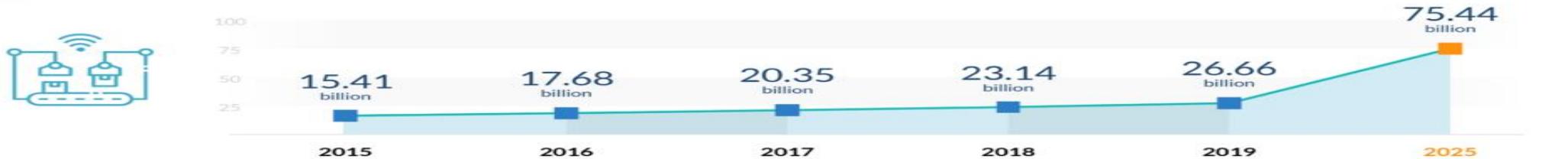
# IoT Trends

## 3 Key IoT Trends You Should Know



### 1 Number of Installed IoT devices around the world

Source: Statista



### 2 Major challenges IoT technology is facing

Sources: Innovation Enterprise, Gartner, Entrepreneur Media, Bitdefender, Brookings Institution

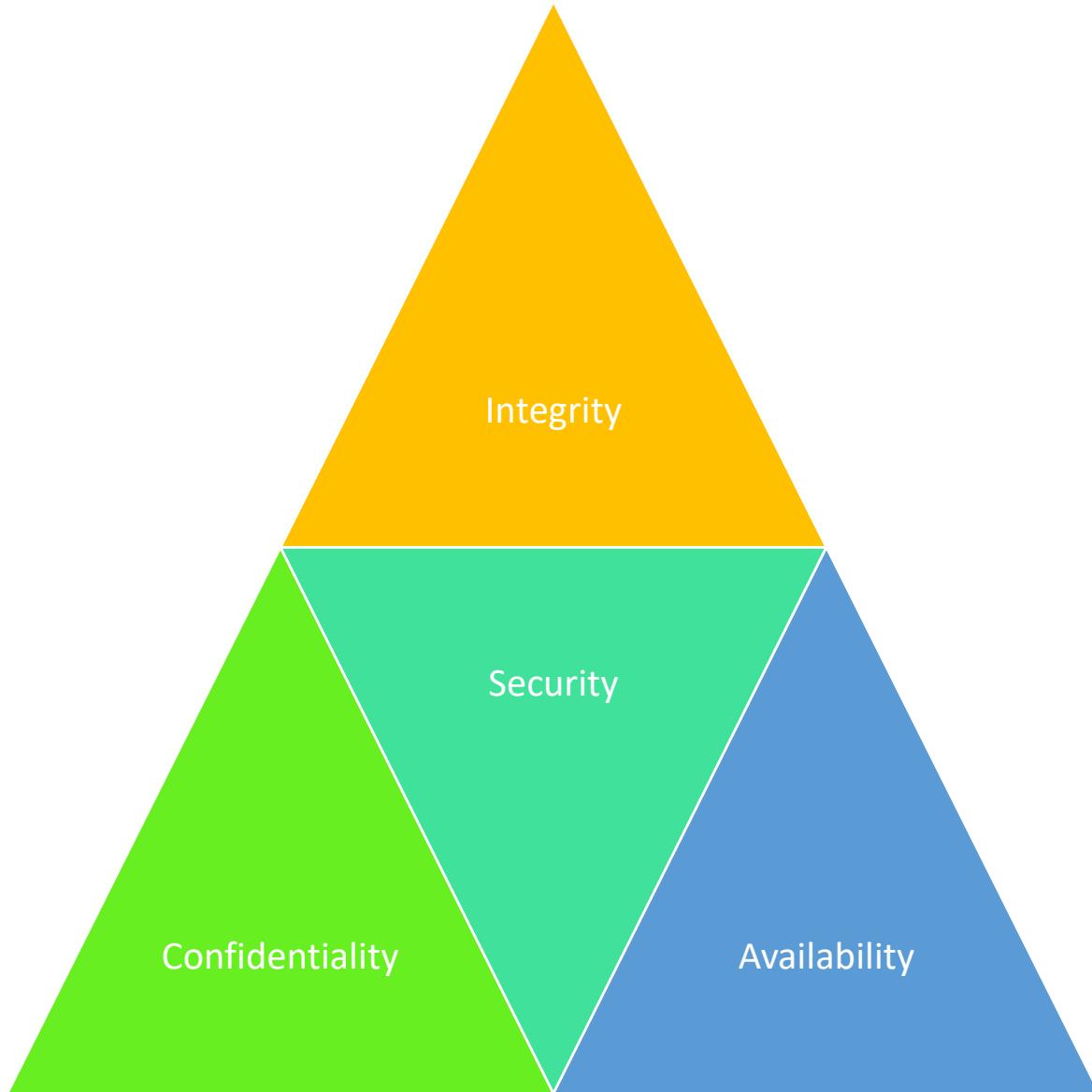


### 3 Perceived, expected, and real benefits of IoT

Sources: Statista, SAS, Data-Smart City Solutions, Tech Republic, Health IT Analytics



# Security Goals



# Security Goals

**Confidentiality:** Confidentiality means protection of data or assets from the unauthorized access. It is applicable not only to the data stored in the system but also to the data in transmission .eg hiding sensitive information in military or in industry from competitors.

**Integrity:** Protection of data from modification of any kind like insertion, deletion and replaying by the attacker i.e only the authorized user can make changes to the data.

**Availability:** It means that the information should be available to the user as and when he needs it. Data is of no use if it is not available on time.

**Accountability:** user is taking responsibility for his or her actions or it is possible to trace the actions to the system or user so that the responsibility for those actions can be established.

# Security Services

**Data Confidentiality:** It is designed to protect data from disclosure attack. Aim is to protect data from traffic analysis and snooping.

**Data Integrity:** It is designed to protect data from unauthorized modification.

**Authentication:** It helps in proving the authenticity of the user/party at the other end of the line. It can be done in two ways:-

- **Peer Entity Authentication:** It Proves the authenticity of the two parties during connection establishment in connection oriented communication
- **Data origin Authentication:** It Proves the authenticity of the source or origin of data in connectionless communication

# Security Services

**Non-repudiation:** This service protects against the repudiation by either of the two parties involved in communication i.e sender and receiver. With the proof of origin, sender cannot deny sending and with the proof of delivery, receiver cannot deny the reception of data.

**Access control:** Here the word access can involve reading, writing, modifying and executing programs and so on and the term access control means protecting the data from unauthorised access.

# Security Mechanisms

Security Mechanisms are used to provide security services. A security service may be provided by one or combination of more than one security mechanism. Various security mechanisms are:-

**Encipherment:** Encipherment means covering or hiding the data. It can be done using cryptography or steganography

**Data Integrity:** It means creating a short checkvalue from the given data using some specific process and appending that checkvalue to the message at the sender side. At the receiver side, receiver also calculates the checkvalue using received data and then compares it with the one received from the sender. If the two checkvalues are same then the data integrity is preserved.

**Digital Signature:** It is a means of verifying the data by the sender and receiver by signing it electronically

Security Mechanisms are used to provide security services. A security service may be provided by one or combination of more than one security mechanism. Various security mechanisms are:-

# Security Mechanisms

**Authentication Exchange:** In authentication exchange, the sender and the receiver exchanges some messages among themselves to prove their identities to each other

**Traffic Padding:** It involves adding some bogus data to the original data traffic to prevent adversary from accomplishing his task of gaining unauthorised access using traffic analysis.

**Routing Control:** It means to select and to change continuously the routes available between the two parties to prevent eavesdropping on a specific route.

**Notarization:** It means appointing a third trusted party between the sender and the receiver to control the communication between them.

**Access Control:** It can use various methods to prove that the particular user has access to the particular data or resources. Some methods of proving access could be passwords or PINs.

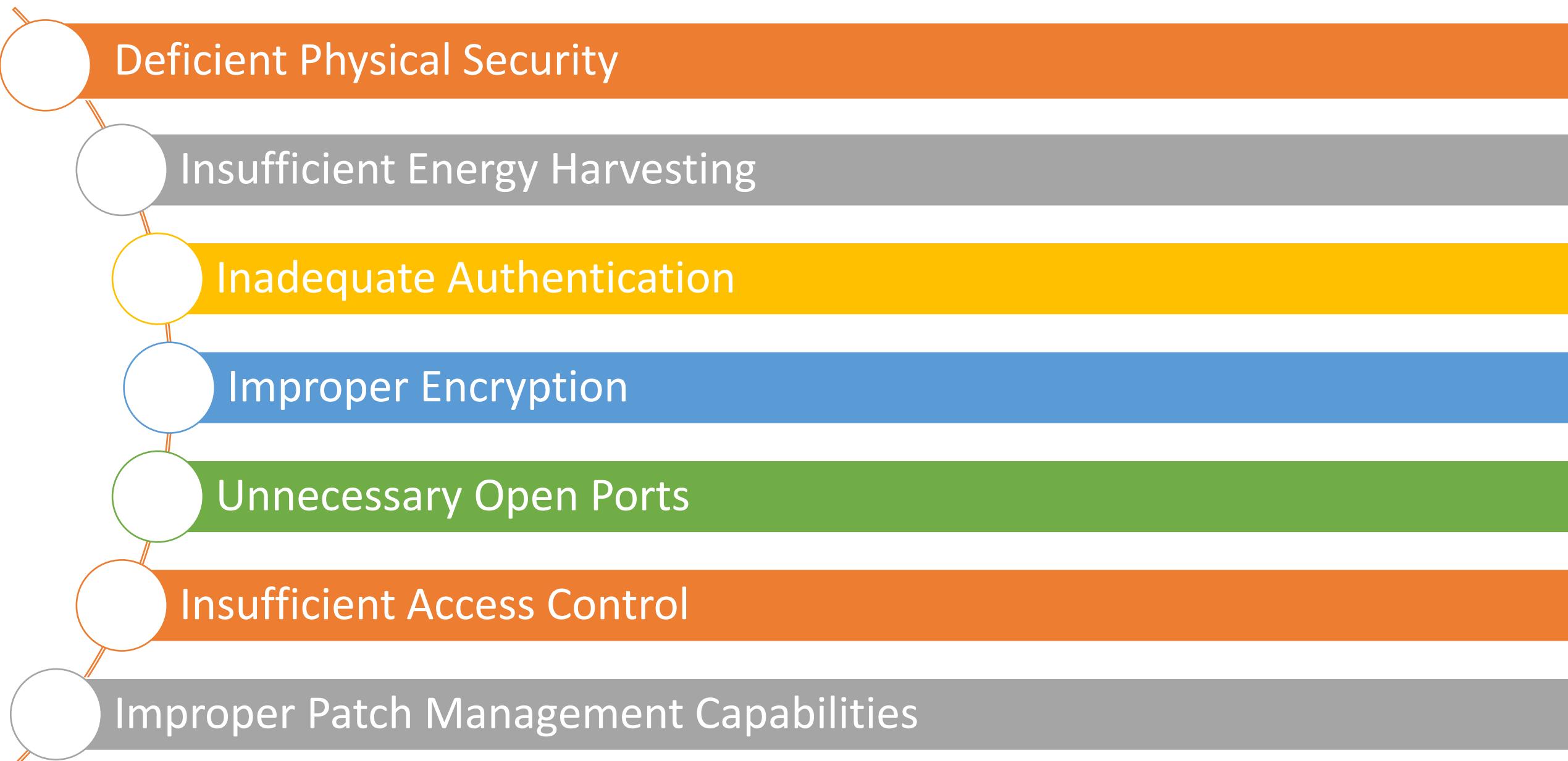
# How do you find **VULNERABILITIES?**



# Vulnerabilities

Vulnerability can be defined as the incompetency of the system which can be exploited by the attacker to attack the system security.

# Common Vulnerabilities



# Deficient Physical Security

Since most of the IoT devices operate independently in an unattended environment, an adversary can very easily get physical access to the system and can take control over them.

**What attacker can do after gaining physical access to the system ?**

Physical Damage to the device.

Unveiling cryptographic scheme used

Gain root passwords

Modify boot parameters

Firmware Replication using malicious node

# Insufficient Energy Harvesting

As IoT devices are characterized to have low energy and also no mechanism to renew it on their own.

## What an attacker can do?

Can drain the battery of the device and make it unavailable to users.

# Inadequate Authentication

Due to the presence of constraints such as low computational power and limited energy, IoT devices cannot implement complex authentication mechanism because of which it is quite easy for an attacker to attack such devices.

## What an attacker can do?

can append any spoofed malicious node.

Authentication keys exchanged may be corrupted.

# Improper Encryption

More complex the encryption algorithm, stronger will be the cryptosystem, but the fact that IoT systems have limited resources make the encryption algorithms less effective, less efficient and less robust. .

## What an attacker can do?

can easily break the cryptosystem and can gain access to the sensitive data.

# Unnecessary Open Ports

Many IoT devices while running vulnerable services may have unnecessary open ports

**What an attacker can do?**

an attacker can connect through open ports and exploit lots of vulnerabilities.

# Insufficient Access Control

A strong credential management system is required to protect data from unauthorised access. Most of the IoT devices in conjunction with their cloud management solutions do not use sufficiently complex passwords. Rather, most of the devices do not ask user to change the default user credentials after installation.

**What an attacker can do?**

an attacker can easily access the device

# Weak Programming Practices

To minimize the attack vectors and to increase the functional capabilities of IoT devices, their operating system and the embedded firmware should be patched properly. But unfortunately most of the manufacturers do not recurrently maintain security patches.

**What an attacker can do?**

Can easily modify firmware

# Improper Patch Management Capabilities

It has been reported by many researchers that various firmware are released with some vulnerabilities such as root users as main access point, backdoors and lack of secure socket layer usage.

**What an attacker can do?**

Can easily modify firmware

Can inject false data

# Insufficient Audit Mechanisms

Since most of the IoT devices do not have thorough logging procedures, making it possible to hide IoT generated malicious activities.

**What an attacker can do?**

Various malicious activities of attacker may go un-noticed

# Attacks in IoT

# Types of Attacks Based on Security Goals

Any activity that threaten any security goal or that compromise the IoT device can be termed as an attack to IoT system. Attacks can be classified broadly on the basis of the security goal that they threaten:

- **Attack against confidentiality:** These types of attacks are mainly done to get unauthorized access to the data or any resource to perform further malicious actions.
- **Attack against integrity:** Attacker can make unauthorized modifications to the data
- **Attack against Availability:** attacker makes the system unavailable to the legitimate users.

# Another Way of Categorising Attacks

## Active Attacks

- may change the data or harm the system.
- threaten the integrity and availability
- are normally easier to detect than to prevent, because an attacker can launch them in a variety of ways.
- Cannot be prevented by using encipherment
- Eg. Firmware modification, False data injection

## Passive Attacks

- The attacker's goal is just to obtain information.
- does not modify data or harm the system.
- threaten confidentiality
- Difficult to detect
- can be prevented by encipherment of the data.
- eg. snooping and traffic analysis are passive attacks.

# Attack Against Confidentiality

Various attacks that threaten the confidentiality are:

- Snooping/Eavesdropping:
- Traffic Analysis
- Dictionary Attack
- Side Channel Attack
- Sybil Attack

# Snooping/Eavesdropping

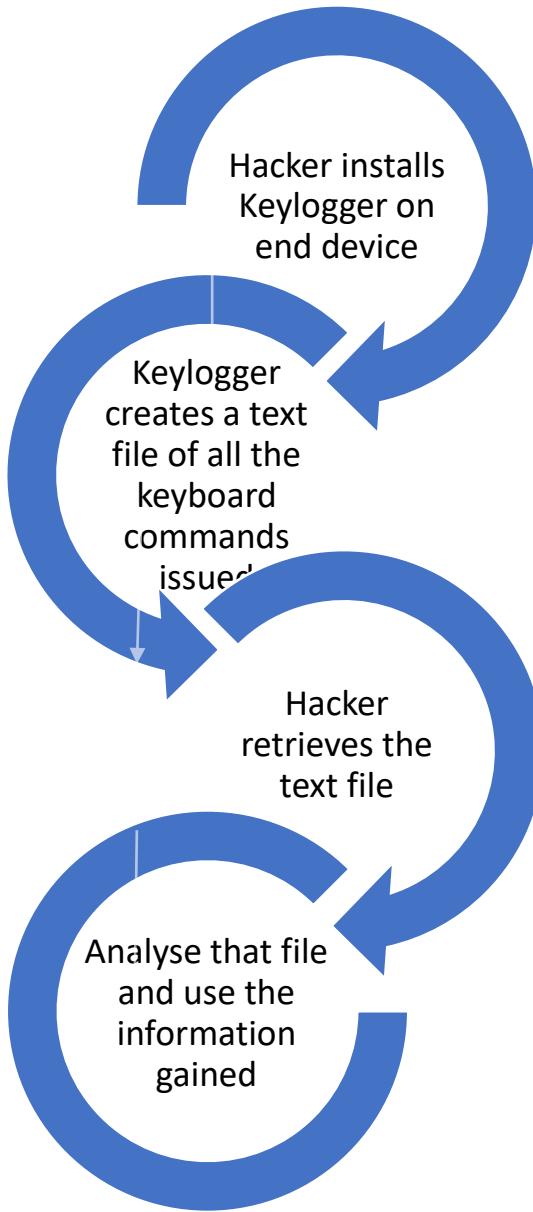
Snooping or Eavesdropping means unauthorized access to or inception of data.

It may include activities from just watching the emails that appear on victim's computer screen, to examining his keystrokes.

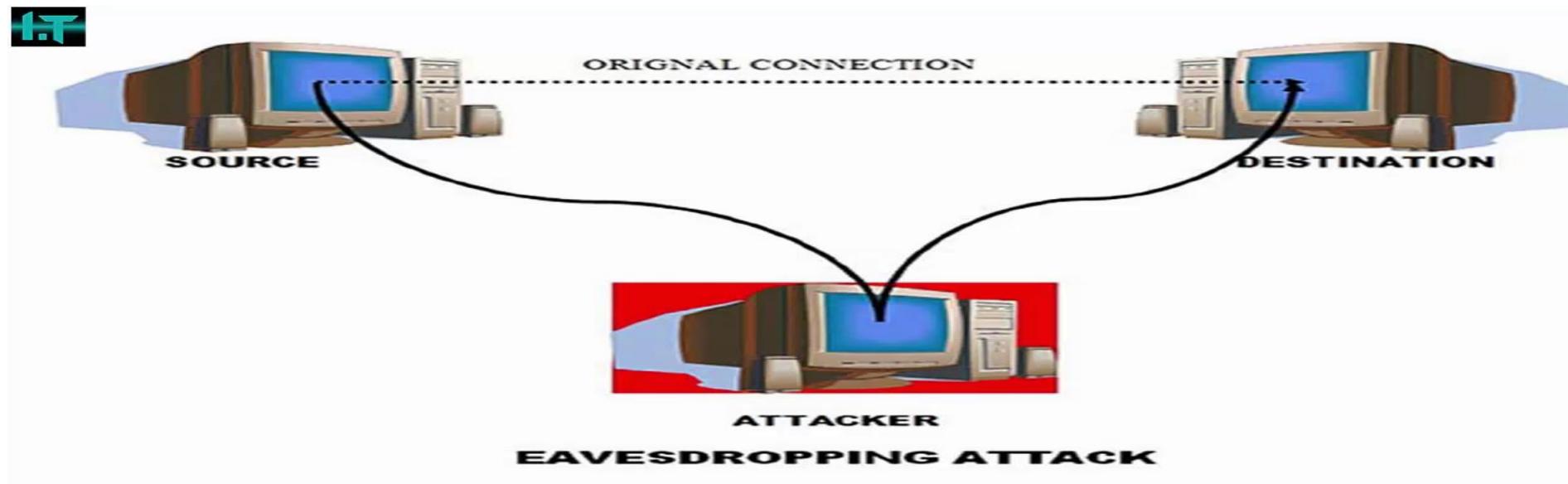
More sophisticated snooping attack may include use of software tools for examining the user's activities remotely or examining the data when the data is in motion.

An example of such tool is **Keylogger**, which is a software used to observe keystrokes including sensitive information such as login credentials and password of the victim.

# Snooping/Eavesdropping



# Snooping/Eavesdropping



# Dictionary Attack

- Dictionary Attack aims to gain illicit access IoT devices by using restricted subset of keyspace.
- It may involve trying millions of possibilities which may be obtained from past security breaches.

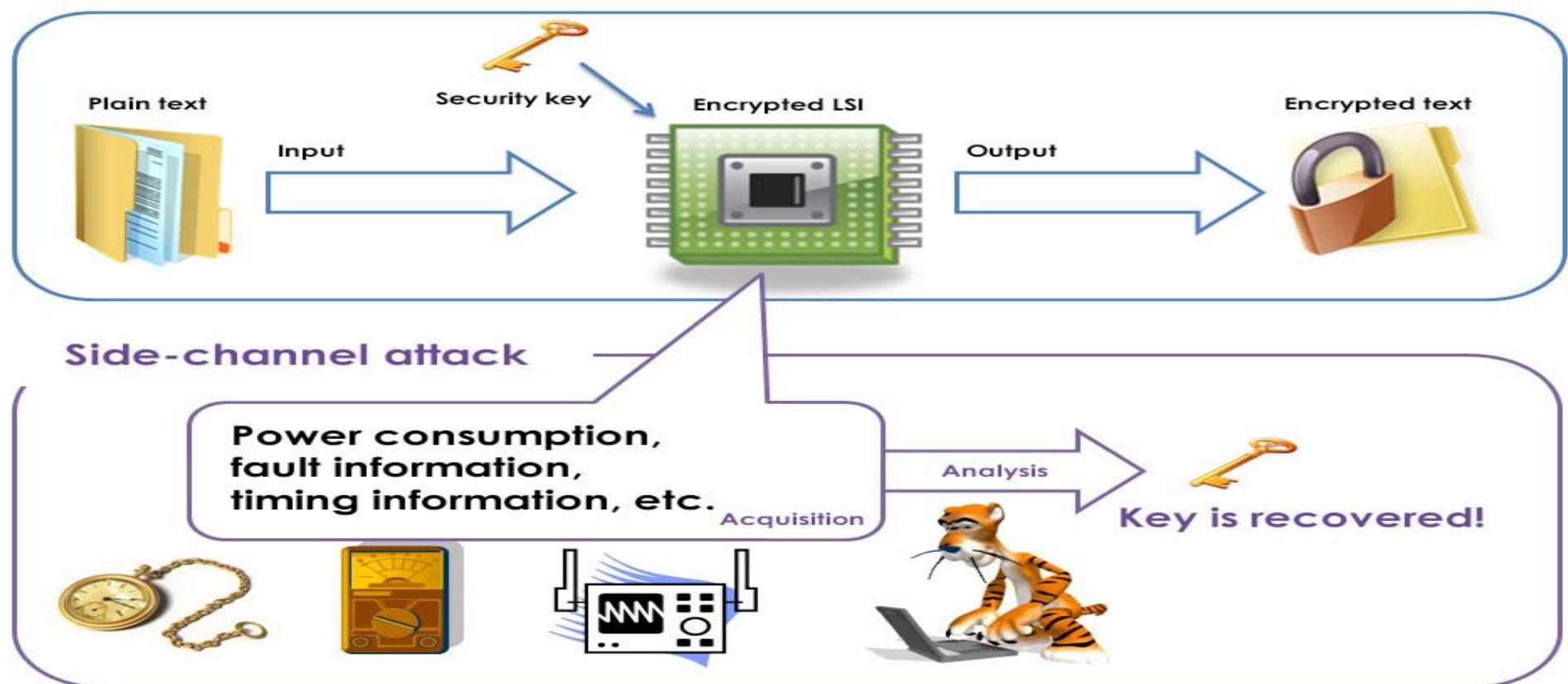
## Dictionary Attack

- Most people use real words as passwords
- Try all dictionary words before trying a brute force attack
- Makes the attack much faster



# Side Channel Attack

- Side channel attack refers to attack strategy where the attacker exploits the extra information related to the way a particular protocol or algorithm is implemented.
- This extra information could be power consumption, timing information, sound, electromagnetic leaks etc.



# Side Channel Attack

**Timing Attack:** In timing attack, an adversary tries to attack the cryptosystem by analysing on how much time does an algorithm take to execute. Each logical operation in computer takes some time to execute which can differ depending on the input provided. By measuring the time accurately for each operation, attacker can use backward approach to determine the input.

**Cache Attack:** Here, attacker can monitor the cache accesses done by the victim in a shared physical system such as virtualized environment or a type of cloud service.

**Power monitoring attack:** This attack can be performed by analysing the power consumption by the hardware during computation.

**Electromagnetic attack:** These attacks are based on leaked electromagnetic radiation, that can directly give plaintext or other information.

# Sybil Attack

In sybil attack a single entity (or device) can create or operate as multiple identities (accounts based on IP addresses, user accounts)in peer to peer network.

To the rest of the world, these different fake identities appears to be genuine identities.

# Attacks Against Data Integrity

False Data Injection Attack

Firmware Modification Attack

Cross Channel Scripting Attack

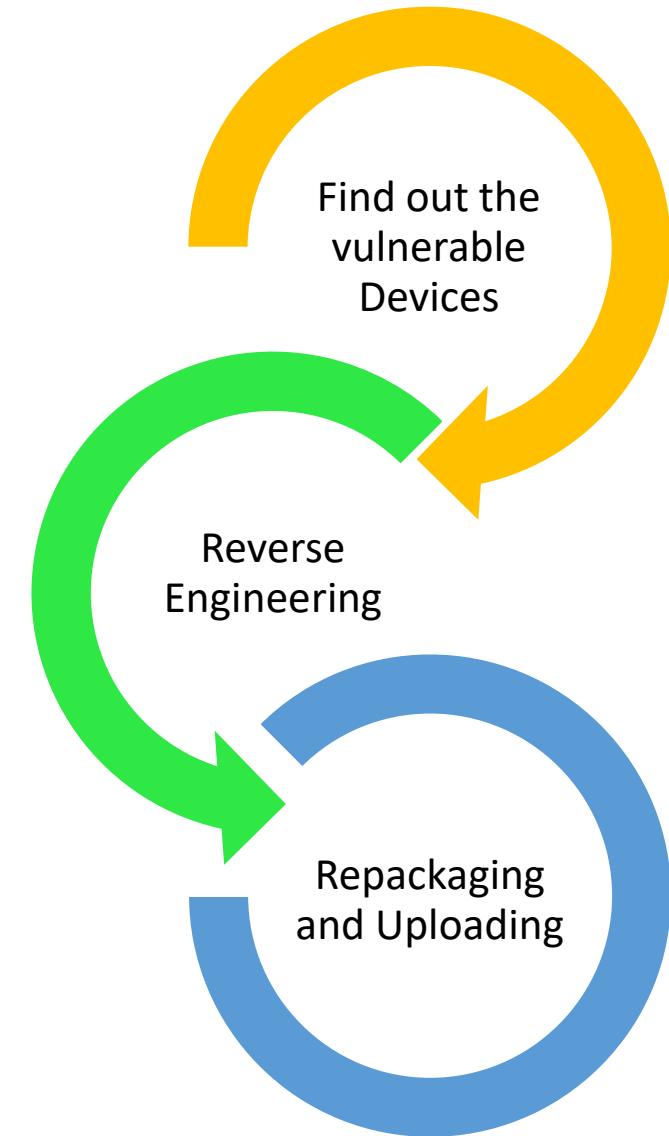
# False Data Injection Attack

In FDI , an attacker injects some malicious data to the IoT sensor due to which various integrity violations may occur.

For example, injecting false data may lead to incorrect estimation of the state of the device employed in some critical environment.

# Firmware Modification Attack

As the name suggests, firmware modification means maliciously changing the firmware of the system to change the functionality of the system.



# Cross Channel Scripting ( CCS Attack)

is a Java Script code injection attack that permits an attacker to execute injected JavaScript in victim's web browser in order to gain access to the sensitive resources like cookies, password, credit card numbers etc.

is an attack on the client-side web browser, but its capabilities are exploited on the web server side.

This script is injected in such a way that it seems to be benign component of the website and finally this script is executed within the domain of the trust of the website.

This is possible only if the web application accepts an input from the user-side into its web pages because an attacker can inject a malicious JavaScript string that will be reflected as a code on the browser of victim

# Attack Against Data Integrity

Various attacks possible against data integrity are:-

- 1. Modification:** The attacker after gaining access to the data can modify it for his own benefits. Eg. an attacker can intercept the transaction request from user to the bank and then modify that request for his benefit
- 2. Masquerading/spoofing:** an attacker impersonates as somebody else. Eg. an attacker somehow manages to get the bank card and PIN and then he pretends that he is that customer in front of the bank.
- 3. Replay:** an attacker gets a copy of the message and then he replays the same message later on. Eg. Bob requests his bank to transfer some money from his account to Eve's account, Eve intercepts that message and then tries to replay it again in the future.

# Attack Against Data Integrity

Various attacks possible against data integrity are:-

4. **Repudiation:** Repudiation means either the sender is denying the fact that he has sent the message or the receiver is denying the fact that he has received the message.
5. **False Data Injection (FDI):** In FDI data integrity can be violated by fusing some legal or corrupted data to the IoT sensor. For example if FDI leads to incorrect estimation of the state of the IoT device it can cause very damaging effects and in extreme case it can even cost human life.
6. **Firmware Modification Attack:** Such types of attacks makes malicious changes to the firmware of the device leading to the functional disruption of the victim device

# Attack Against Availability

Device/node Capture

Routing Attack

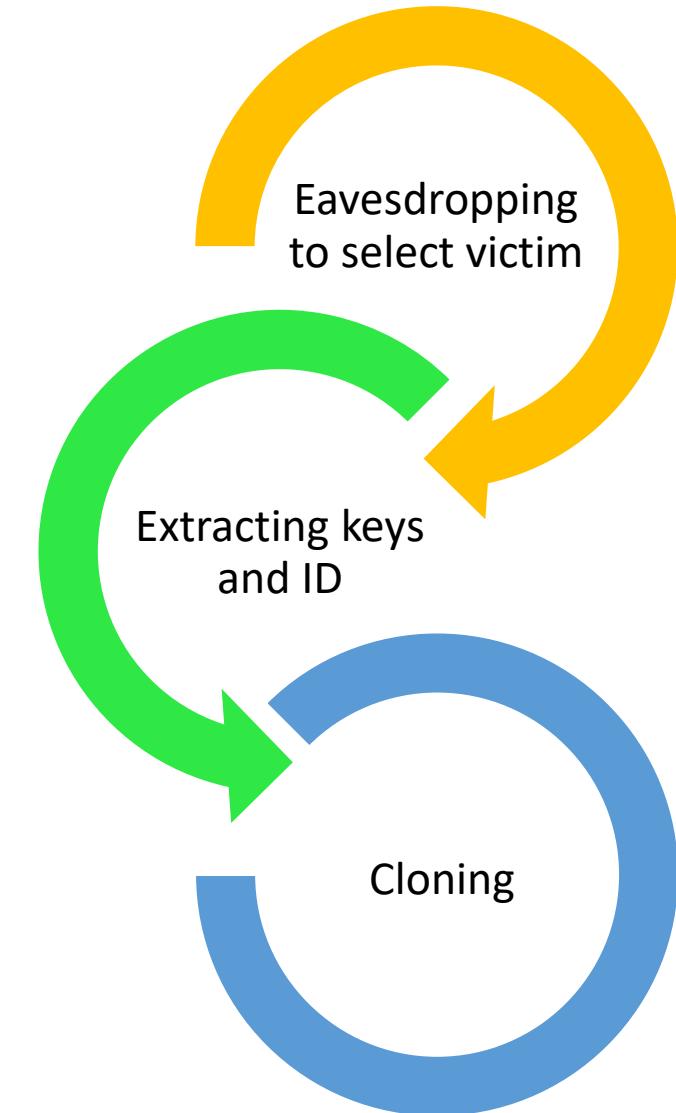
Denial of Service(DoS)

Distributed Denial of Service(DDoS)

Battery Draining Attack

# Node/Device Capture

In node capture attack, an adversary can gain physical access to the device which are usually placed in a hostile environment



# Routing Attack

An adversary degrades the performance of the network by modifying the network topology. Malicious node is given the power to advertise an artificial routing path containing large number of nodes in order to compel them to send packets through such fake paths. Various kinds of Routing attacks are:

- Sinkhole Attack
- Blackhole Attack
- Wormhole Attack
- Rank Attack
- Sybil or clone ID
- HELLO flooding Attack

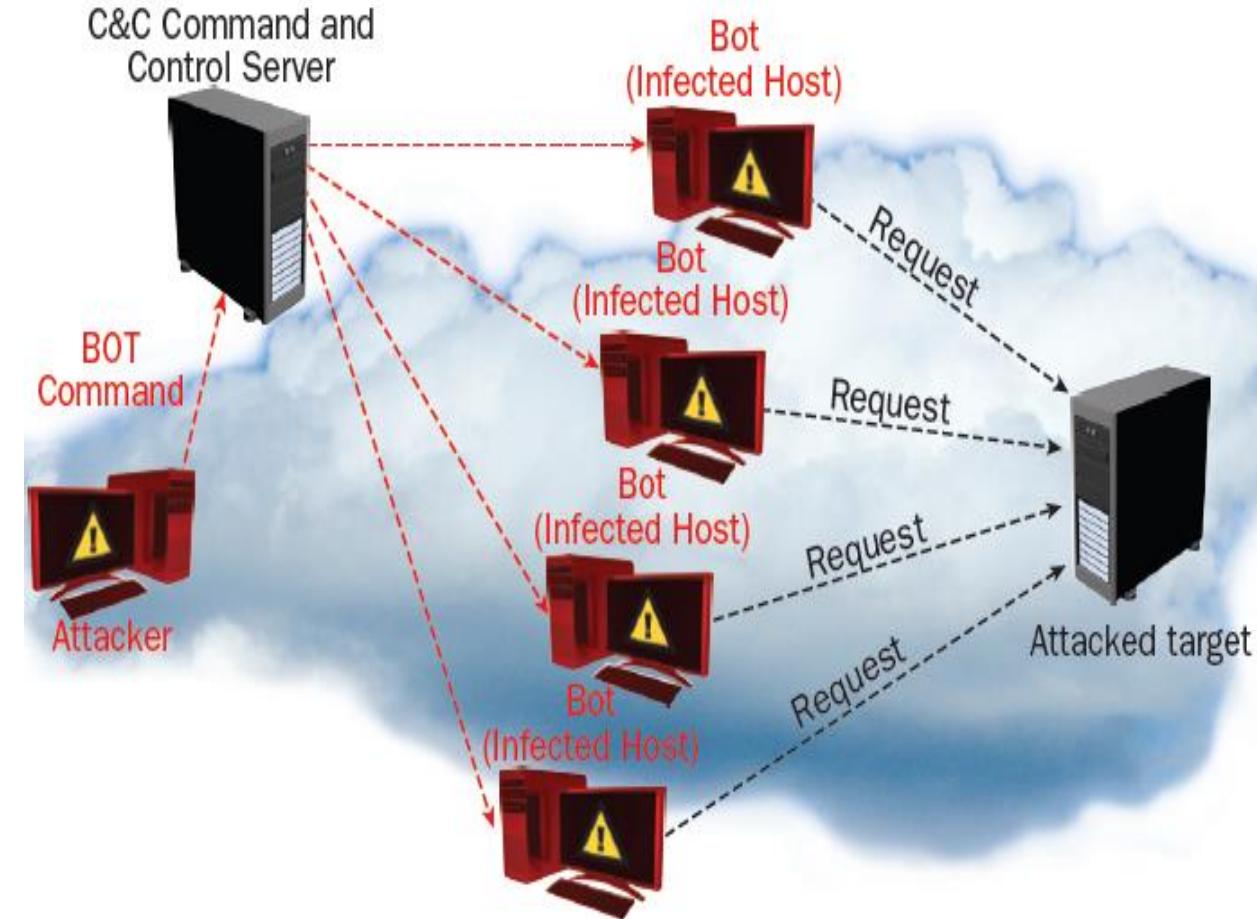
# Denial Of Service Attack

DoS attack is an attack strategy in which an attacker overwhelms the resources of the victim's system to such an extent that the attacked system is not able to provide any service or resources to the legitimate users.

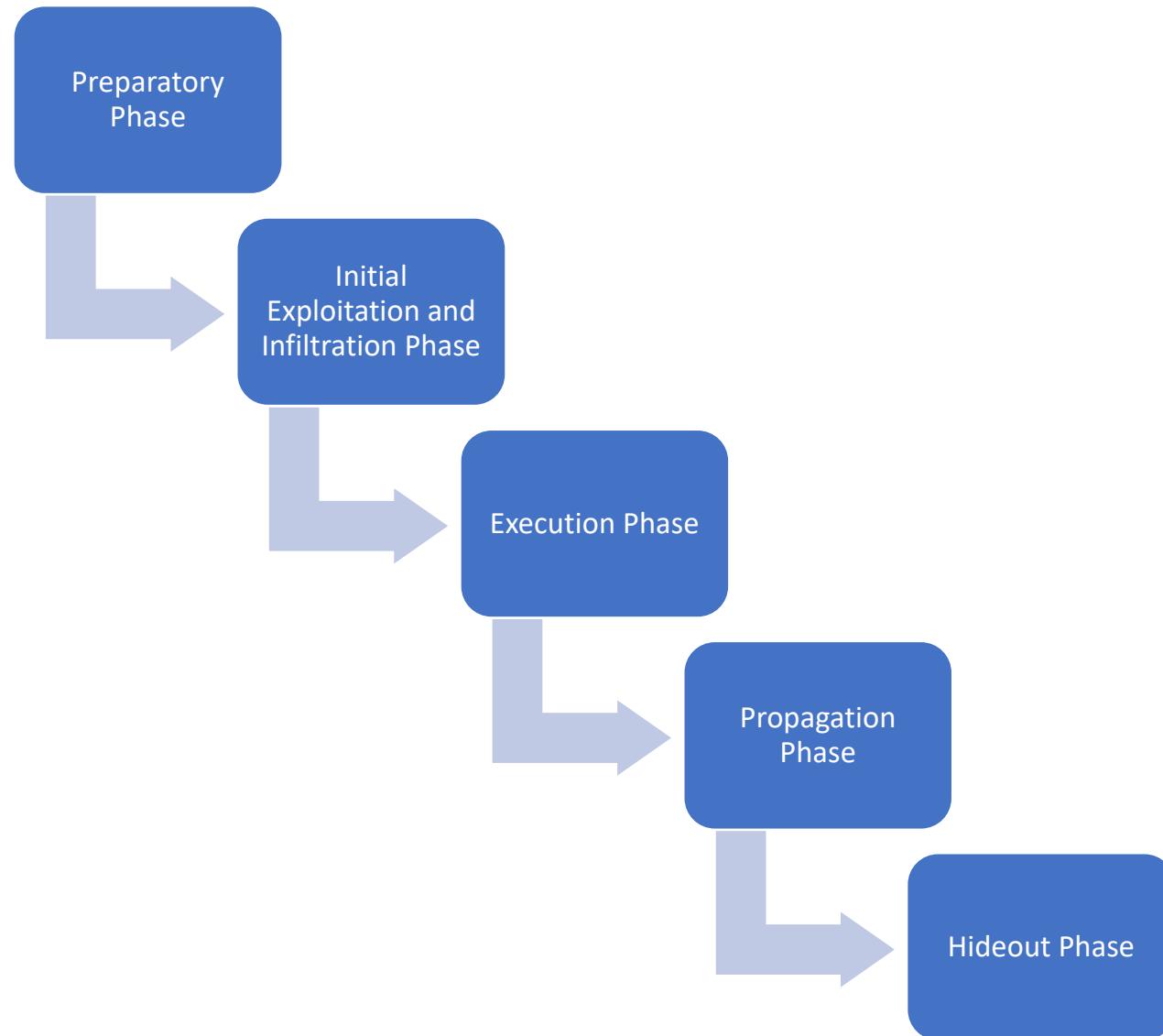
It affects the availability of the system.

# DDoS Attack

In Distributed Denial of Service (DDoS) attack, resources of the victim system are flooded with the multiple requests from the large number of malware infected bots, so that the victim becomes unavailable to the legitimate users.



# DDoS Attack Methodology



# DDoS Attack Methodology

1. **Preparatory Phase:** Attacker examines the target system and check for the devices with vulnerabilities, these vulnerabilities could be use of default credentials, unnecessary open ports, device hardware or software weaknesses etc.
2. **Initial Exploitation and Infiltration Phase:** Malware enters the vulnerable device and then tries to login using Brute force technique
3. **Execution Phase:** Steps taken by malware in execution phase
  - Download the additional payload from the server and delete any other malware present in the system.
  - reconfigure the device to make it a part of Botnet,
  - execute the downloaded malware binary and
  - finally perform the particular malicious task.

Bot communicates with the controller regularly.

# Types of DDoS Attack

**Volumetric/Flooding Attack**

**Amplification Attack**

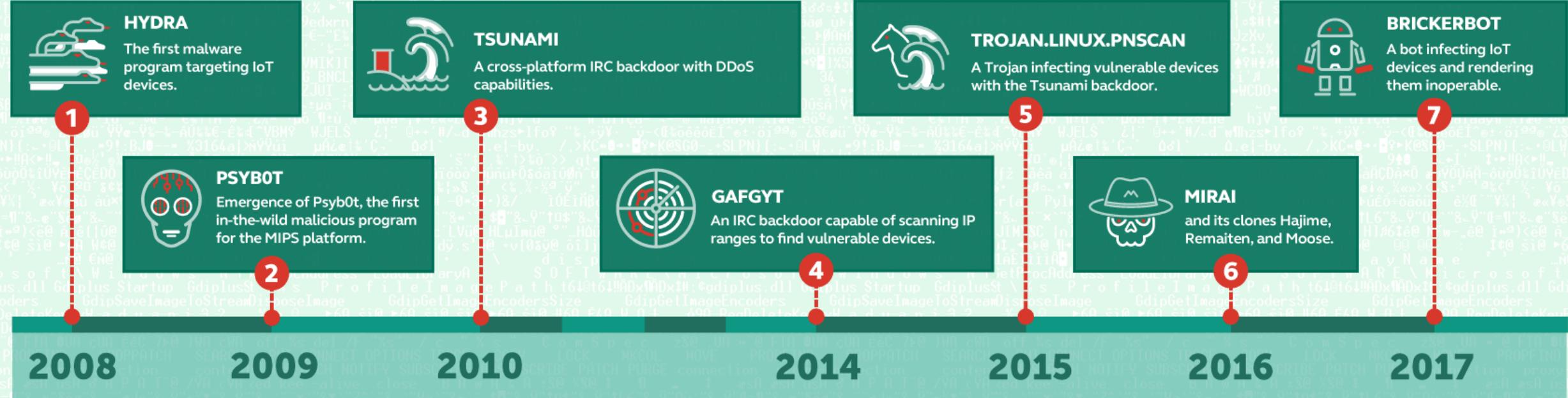
**Protocol Exploit**

**Logical/Software Attack**

# Malware Attack

# IoT devices at risk: malicious programs target the ‘Internet of Things’

Currently, over 6 billion of ‘smart’ devices exist globally. It was when the Mirai botnet emerged in 2016 that the whole world learned how dangerous such devices may become in the hands of cybercriminals. However, the history of malware attacking IoT devices began much earlier.

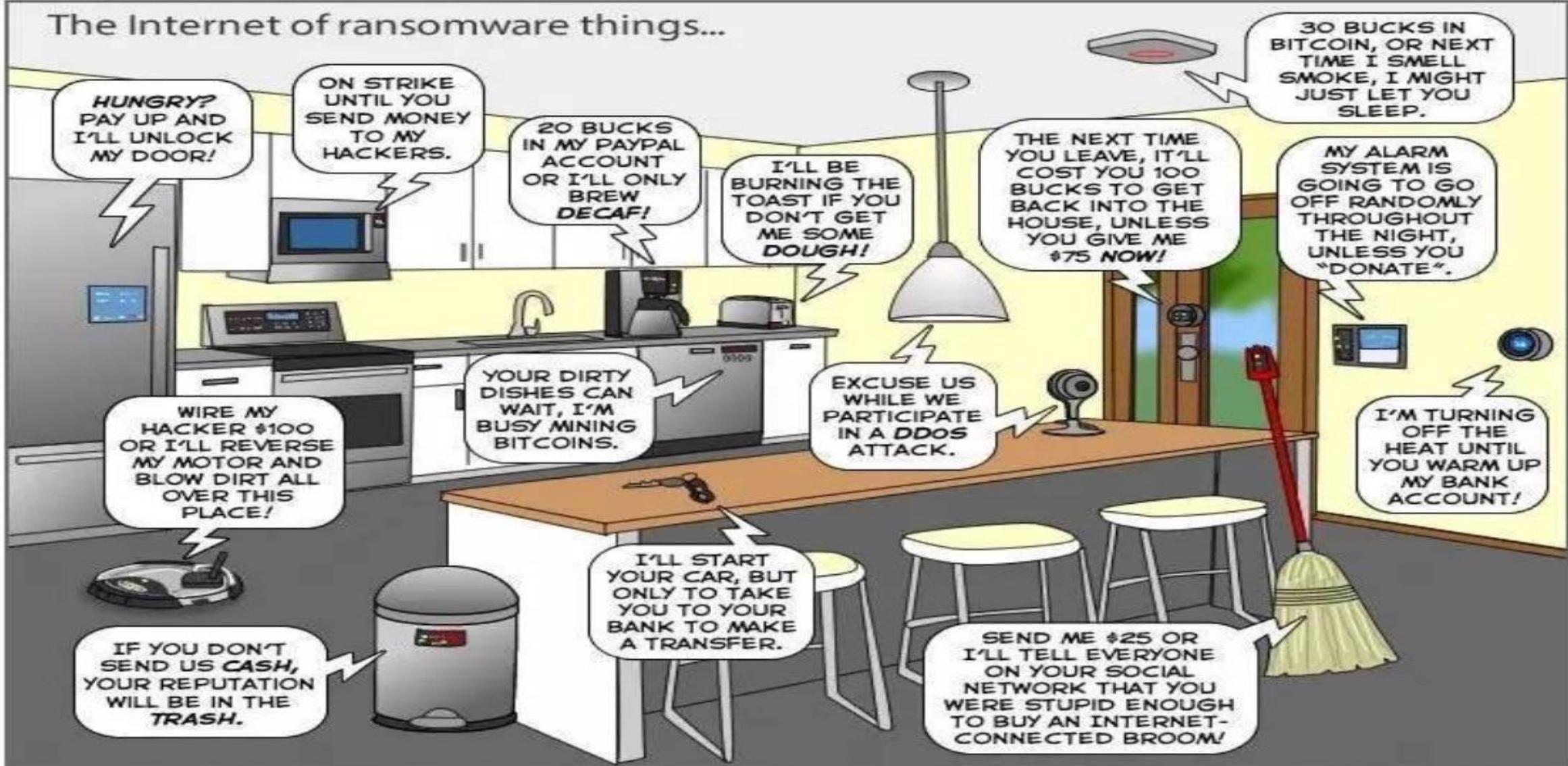


# Ransomware Attack

**It is a kind of malware attack, where the attacker takes hold of some file, folder or an entire device forcibly and does not release it until the ransom amount is paid.**

**It is one of the most common cyberattacks.**

## The Internet of ransomware things...

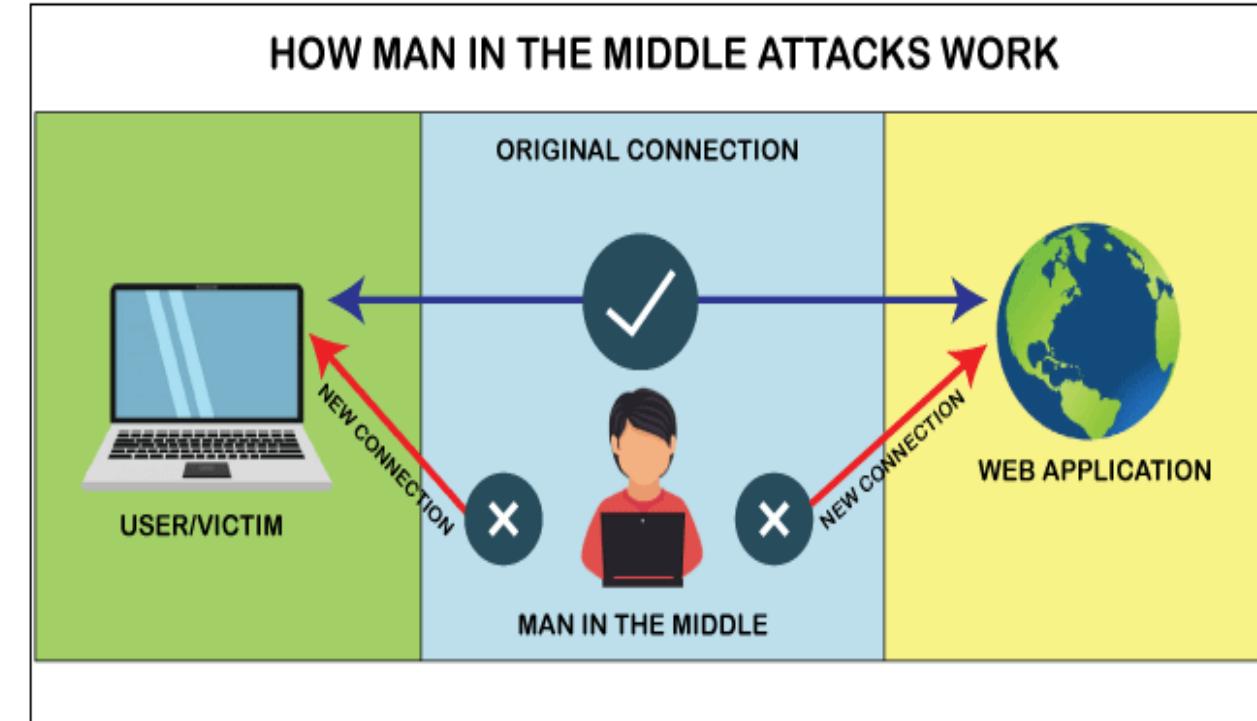


You can help us keep the comics coming by becoming a patron!  
[www.patreon/joyoftech](http://www.patreon/joyoftech)

[joyoftech.com](http://joyoftech.com)

# Man In The Middle (MitM) Attack

MITM occurs when a threat actor inserts himself between two parties, often a user and an application, to intercept their communications and data transfers and utilize them for nefarious ends such as making unauthorized purchases or hacking.



# Traditional Defence Mechanisms

Filter Packets

Adopt Encryption

Employ Robust password Authentication Scheme

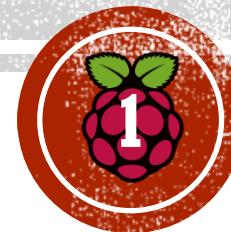
Audit and log Activities

Use Intrusion Detection System

Prevent Intrusion Using Intrusion prevention System

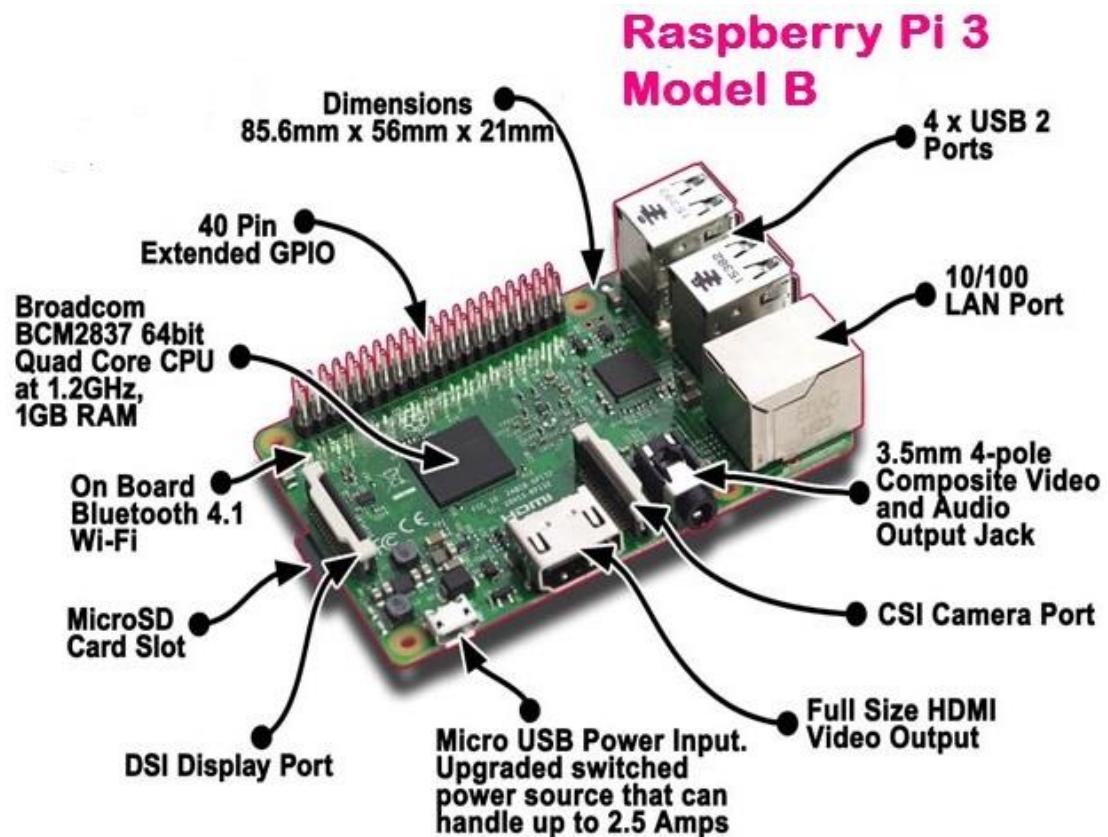
# Thank you

# RASPBERRY PI



# INTRODUCTION

- Raspberry Pi as a world's most inexpensive and powerful Single Board Computer.**



# FEATURES

- **CPU:** Broadcom BCM2837 **SOC** 64-bit quad-core ARM Cortex A53 with 512KB shared L2 cache.
- **Memory:** Provided with 1 GB of RAM
- **Wi-Fi Support:** 802.11n Wireless LAN
- **Bluetooth:** Supports Bluetooth 4.1 Bluetooth Low Energy (BLE)
- **USB Ports:** 4-USB ports.
- **Ethernet Port:** It is useful when we want to setup raspberry pi for the first time without a monitor.
- **GPIO Pins:** Raspberry Pi 3 supports 40 GPIO Pins General Purpose Input Output. Pins can be used to drive LED, Switches, and Sensors.
- **Full HDMI Port:** Support HDMI port (High-Definition Multimedia Interface) which can be used to quickly connect raspberry pi to HDMI Monitor. With HDMI Cable and Monitor we can add Screen to Raspberry Pi.
- **Micro SD card slot:** This Card will hold the operating system which will boot while we power on Raspberry Pi 3.
- **Audio/Video:** Combined 3.5mm audio jack and composite video.
- **Camera interface (CSI):** enable us to interface Camera Module.

# RPI INSTALLATION

**Install Raspbian in SD Card Using  
NOOBs (New Out of Box Software)**

4

# WHAT ALL DO YOU NEED?

- Raspberry Pi
- HDMI Monitor
- HDMI Cable
- USB Keyboard and Mouse
- MicroSD Card
- Power Supply



# PREPARE SD CARD WITH NOOBS

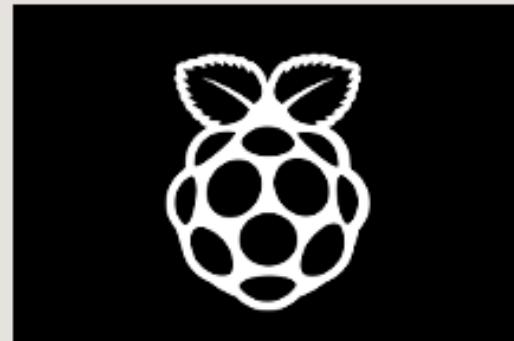
- Insert your SD card into the computer or use SD card reader.
- Download NOOBs (New Out of Box) . From

<https://www.raspberrypi.org/downloads/>

[HTTPS://WWW.RASPERRYPI.ORG/DOWNLOADS/](https://www.raspberrypi.org/downloads/)

## DOWNLOADS

Raspbian is our official operating system for **all** models of the Raspberry Pi. Download it here, or use **NOOBS**, our easy installer for Raspbian and more.



NOOBS



RASPBIAN

Select NOOBS



# NOOBS

Beginners should start with NOOBS – New Out Of the Box Software. You can purchase a pre-installed NOOBS SD card from many retailers, such as [Pimoroni](#), [Adafruit](#) and [The Pi Hut](#), or download NOOBS below and follow the [software setup guide](#) and [NOOBS setup guide video](#) in our help pages.

**NOOBS** is an easy operating system installer which contains [Raspbian](#) and [LibreELEC](#). It also provides a selection of alternative operating systems which are then downloaded from the internet and installed.

**NOOBS Lite** contains the same operating system installer without Raspbian pre-loaded. It provides the same operating system selection menu allowing Raspbian and other images to be downloaded and installed.

## NOOBS

Offline and network  
install

Click on  
**Download ZIP**



### NOOBS

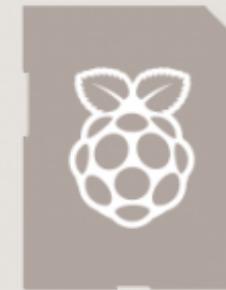
Offline and network install

Version: 2.9.0

Release date: 2018-10-11

[Download Torrent](#)

[Download ZIP](#)



### NOOBS LITE

Network install only

Version: 2.9

Release date: 2018-10-11

[Download Torrent](#)

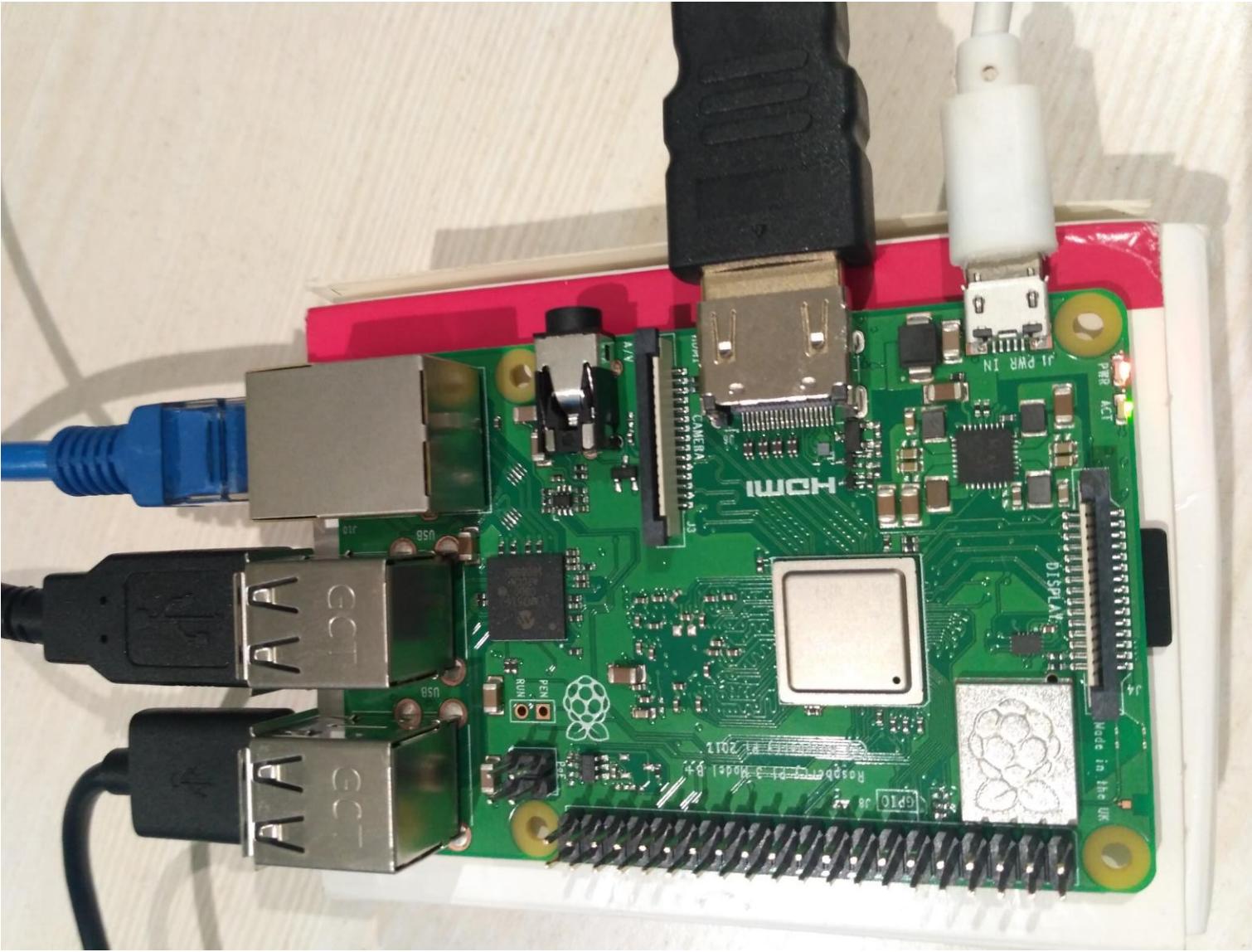
[Download ZIP](#)

# INSTALLATION CONT...

- Format your SD card before copying NOOBs file.
- After formatting, extract the zip file of NOOBs in your computer and copy all the files onto your SD card.
- Once completed, remove the SD card and insert it into your Pi board.

# MAKING CONNECTIONS

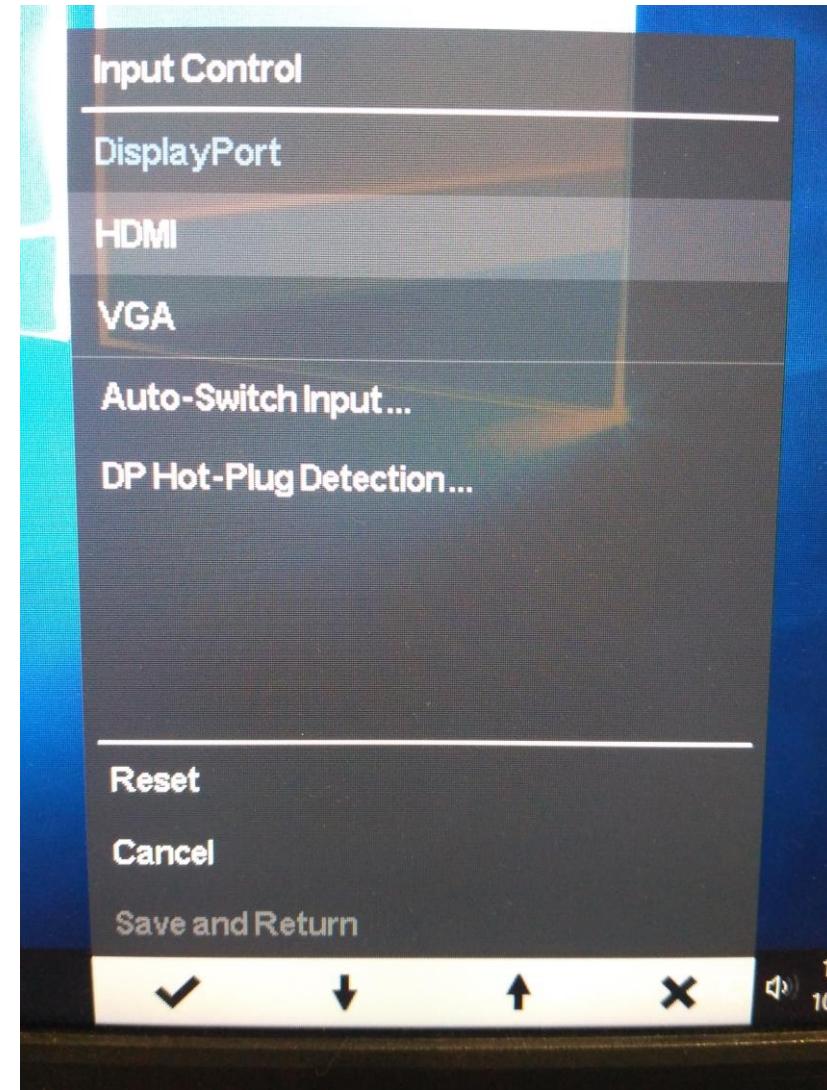
- First of all, connect your HDMI cable to your Raspberry Pi and your Monitor.
- Then, connect all your USB devices.
- If you are using ethernet cables connected to your router; connect it to your Raspberry Pi as well.
- When everything is connected, plug in your power adapter and you are good to go.
- No switch in Raspberry Pi to switch it on/off.

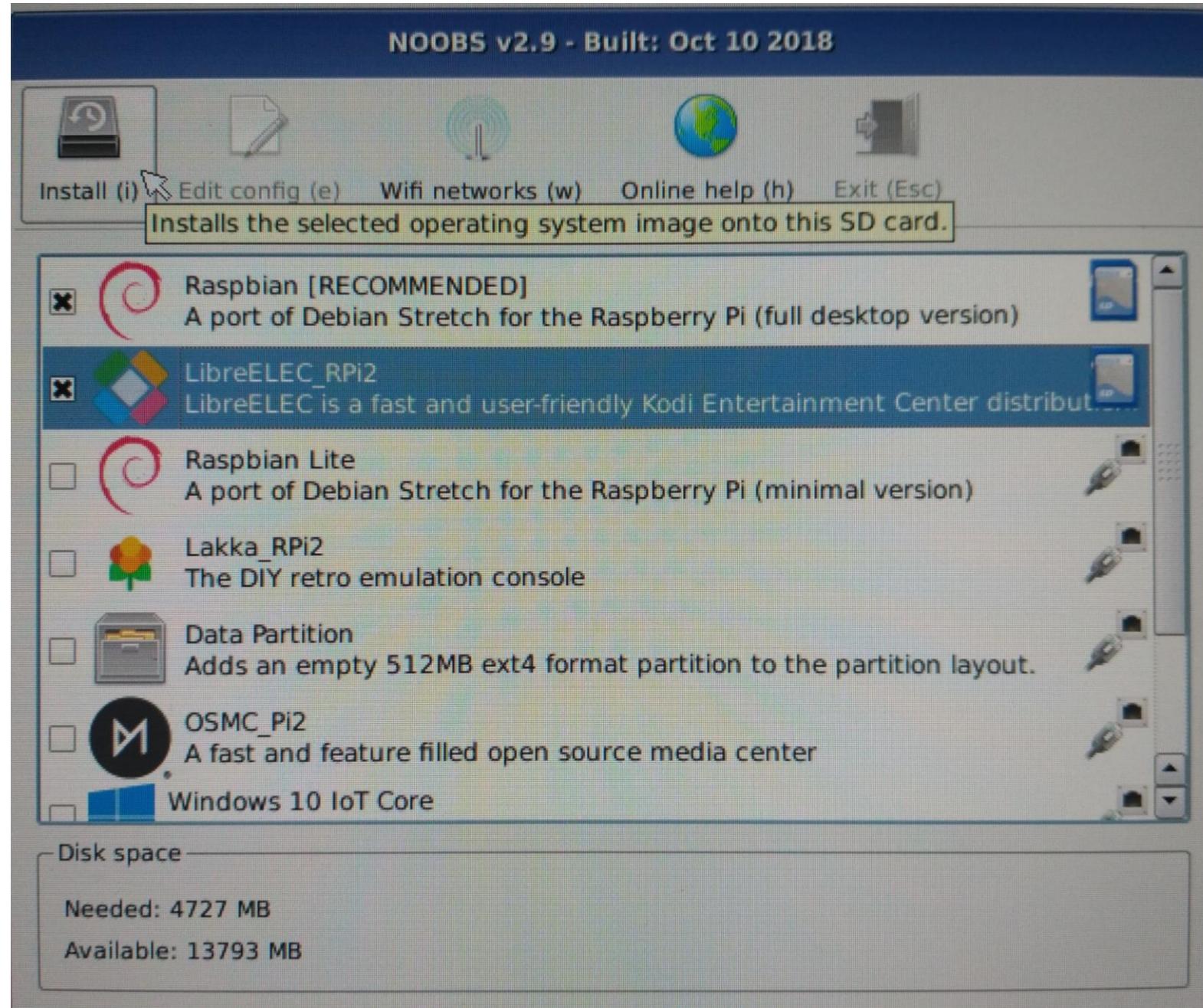


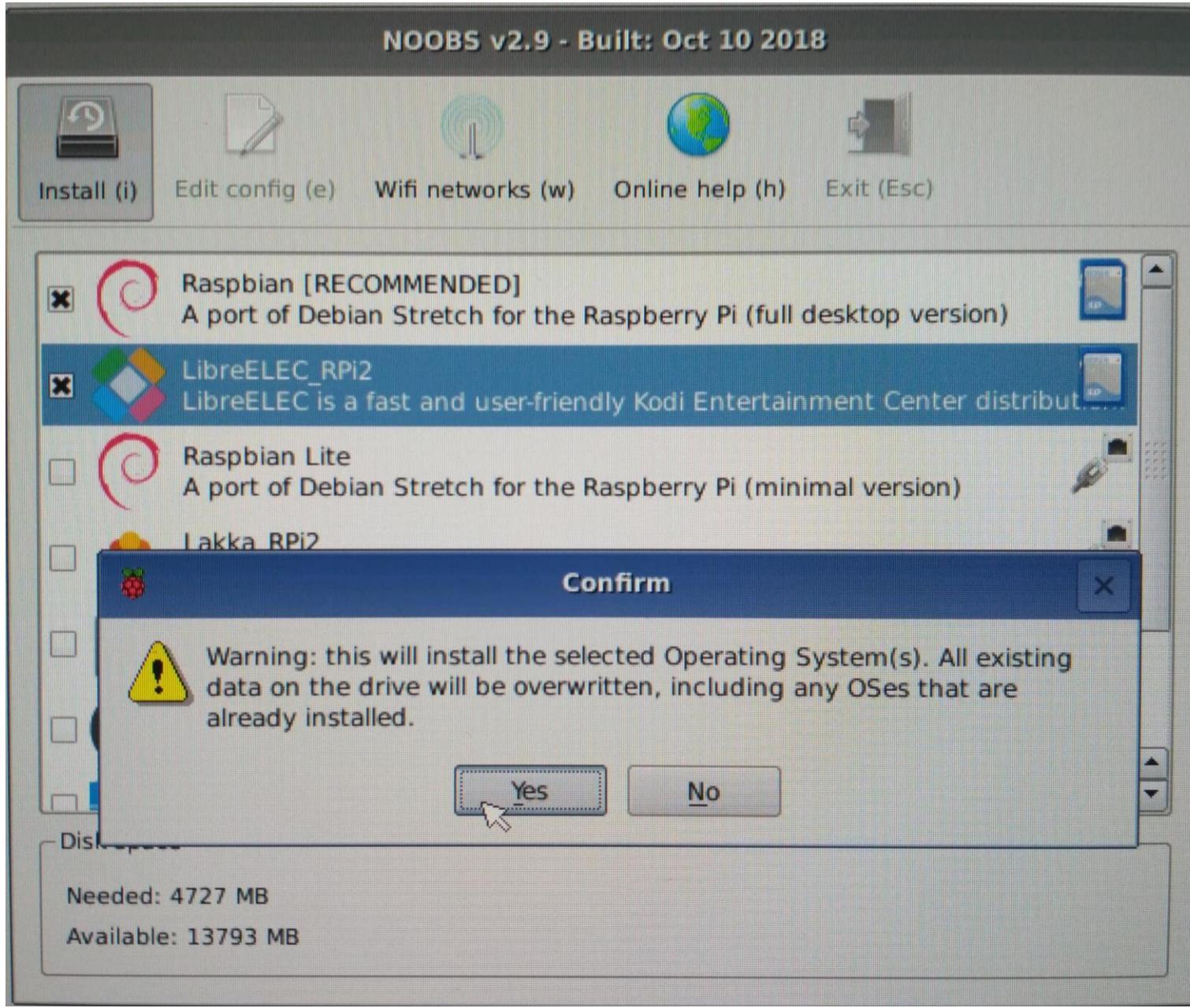
Gaurav Singal

# INSTALLATION CONT...

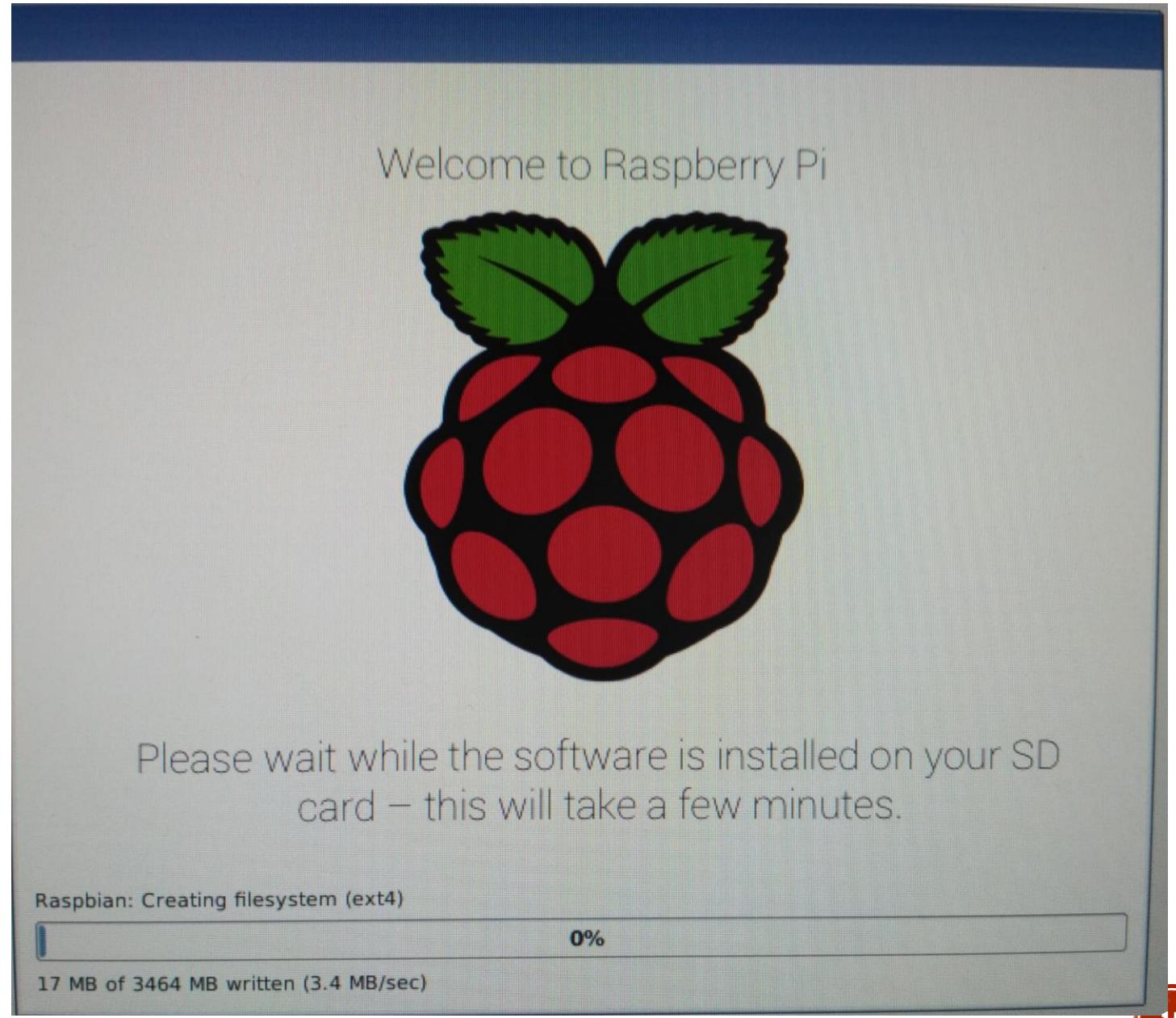
- Select HDMI output on monitor.
- You will see screen as:

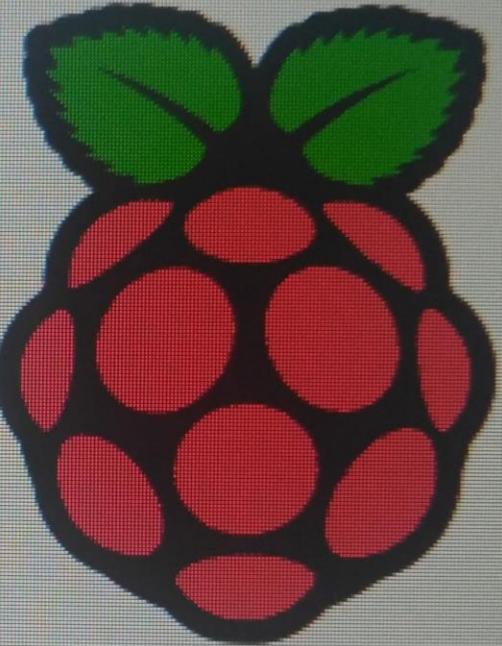
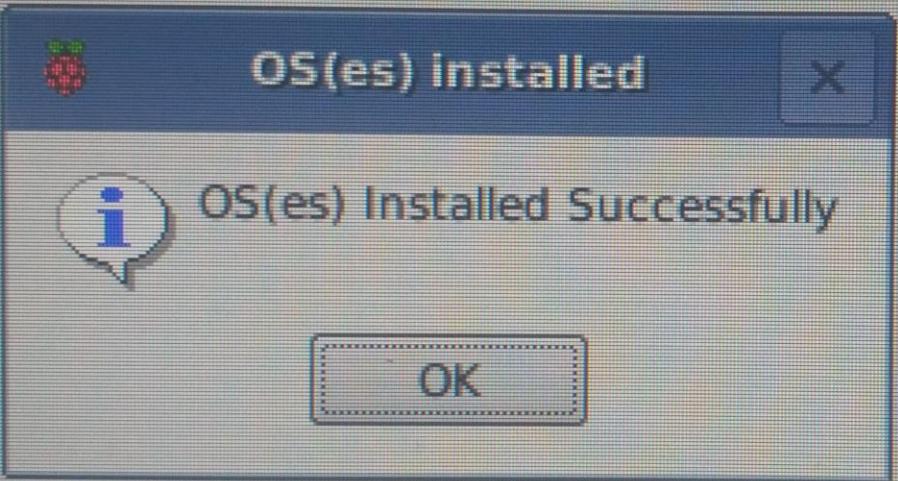


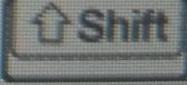


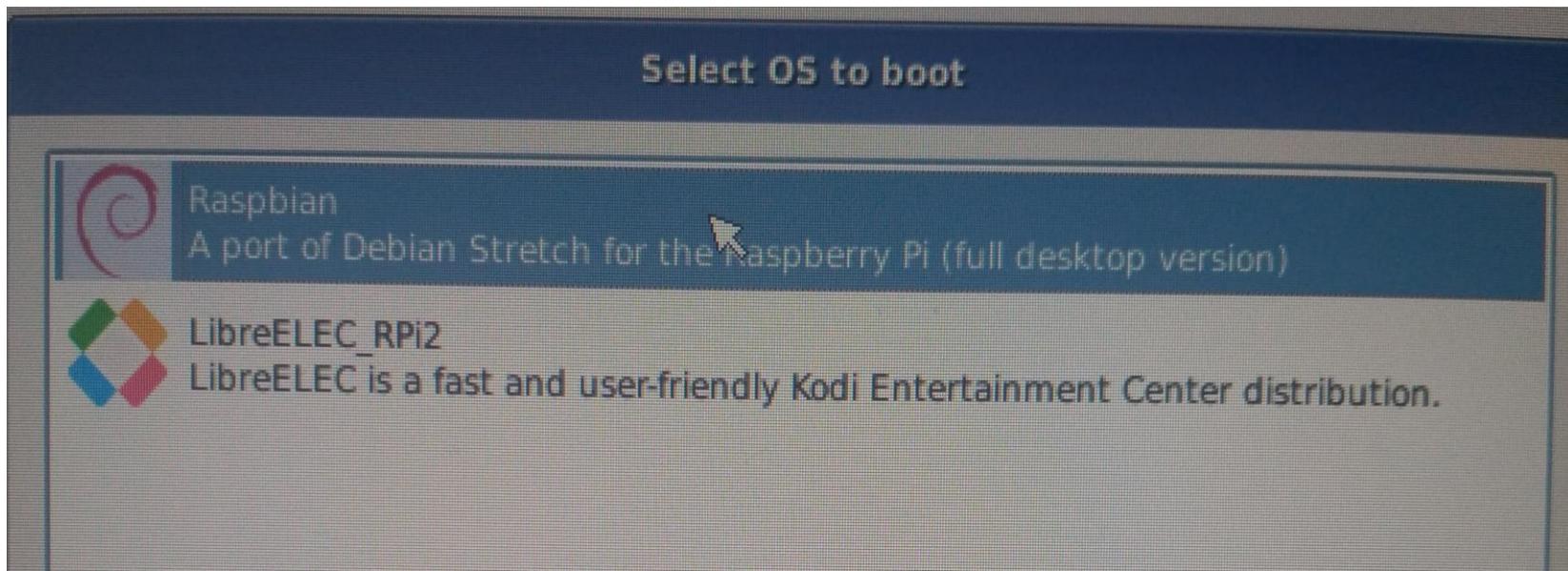


**WAIT FOR  
15-20  
MINUTES**

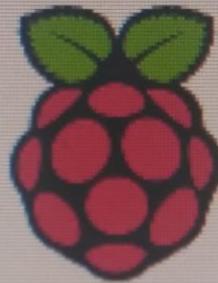




For recovery mode, hold  Shift



# Welcome to Raspberry Pi



Welcome to the Raspberry Pi Desktop!

Before you start using it, there are a few things to set up.

Press 'Next' to get started.

IP : 169.254.224.241

Cancel

Next

## Set Country

Enter the details of your location. This is used to set the language, time zone, keyboard and other international settings.

Country:

India

Language:

English

Timezone:

Kolkata

Use US keyboard

Press 'Next' when you have made your selection.

Back

Next

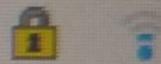
# Welcome to Raspberry Pi



## Select WiFi Network

Select your WiFi network from the list.

iMAC LAB



Press 'Next' to connect, or 'Skip' to continue without connecting.

Back

Skip

Next

## Update Software

The operating system and applications will now be checked and updated if necessary. This may involve a large download.

Press 'Next' to check and update software, or 'Skip' to continue without checking.



Back

Skip

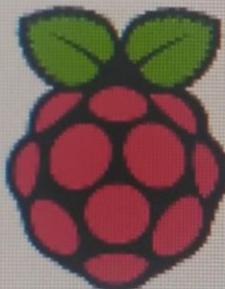
Next



## Setup Complete

Your Raspberry Pi is now set up and ready to go.

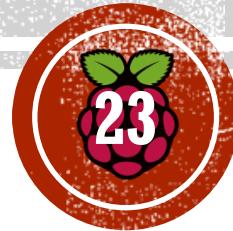
Press 'Reboot' to reboot your Pi for the new settings to take effect.



Back

Reboot

# LED BLINKING



Gaurav Singal

<https://tejalal.wordpress.com/2018/10/16/led-blinking-using-raspberry-pi-and-python/>

# WHAT YOU NEED ?

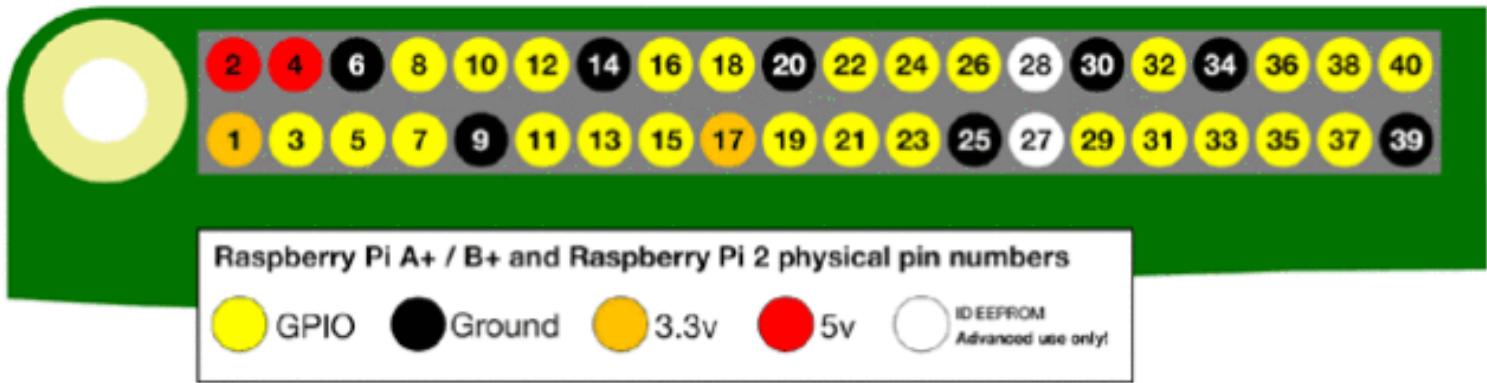
- Raspberry Pi
- Keyboard, HDMI, Mouse
- MicroSD Card
- Power Supply
- LED
- Jumper Wire
- Register 1000 om
- Breadboard

# STEP 1

- Connect all the components
  - HDMI Cable, Power supply,
  - Mouse, Keyboard
  - As show in the diagram.

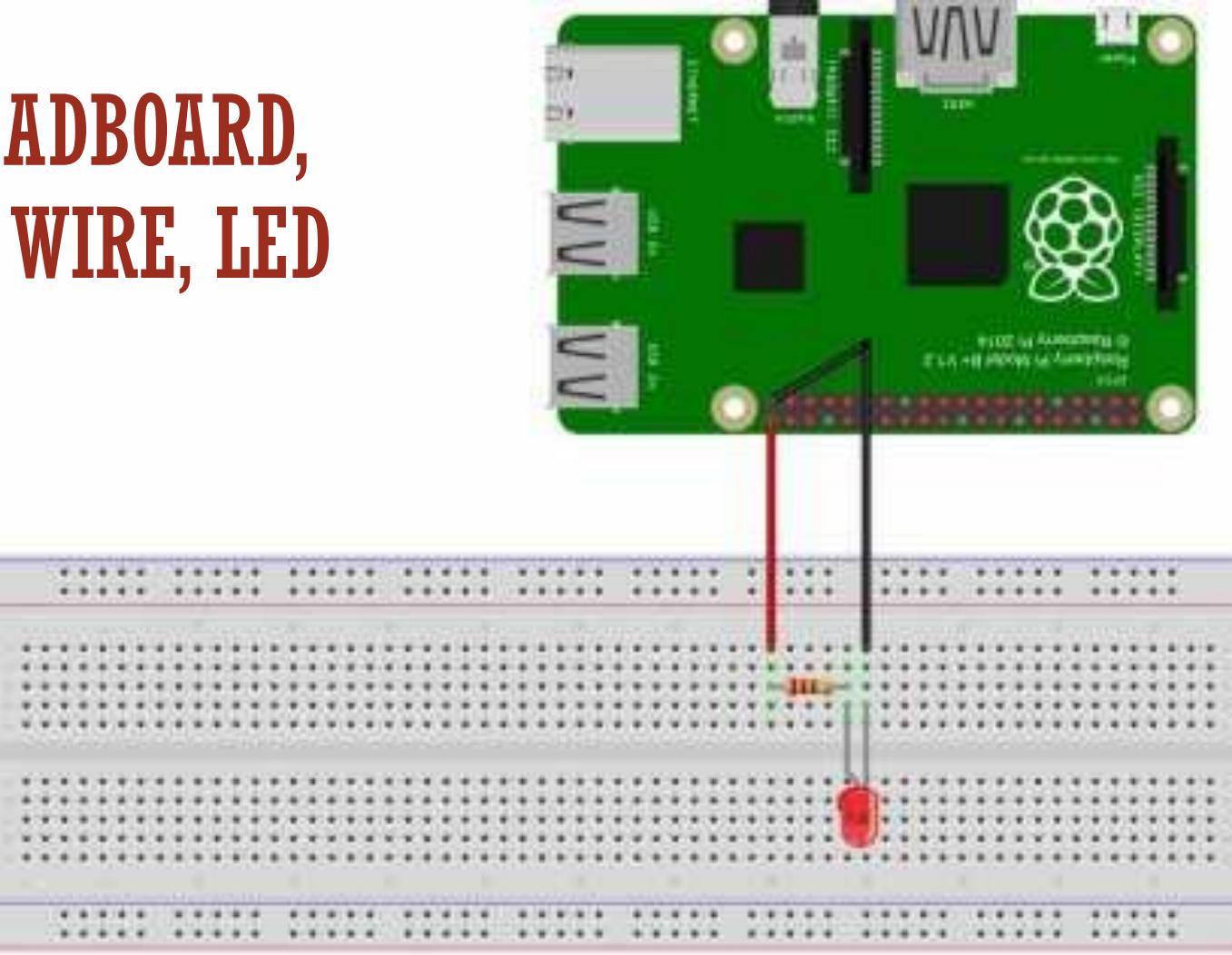


# PIN LAYOUT ON PI



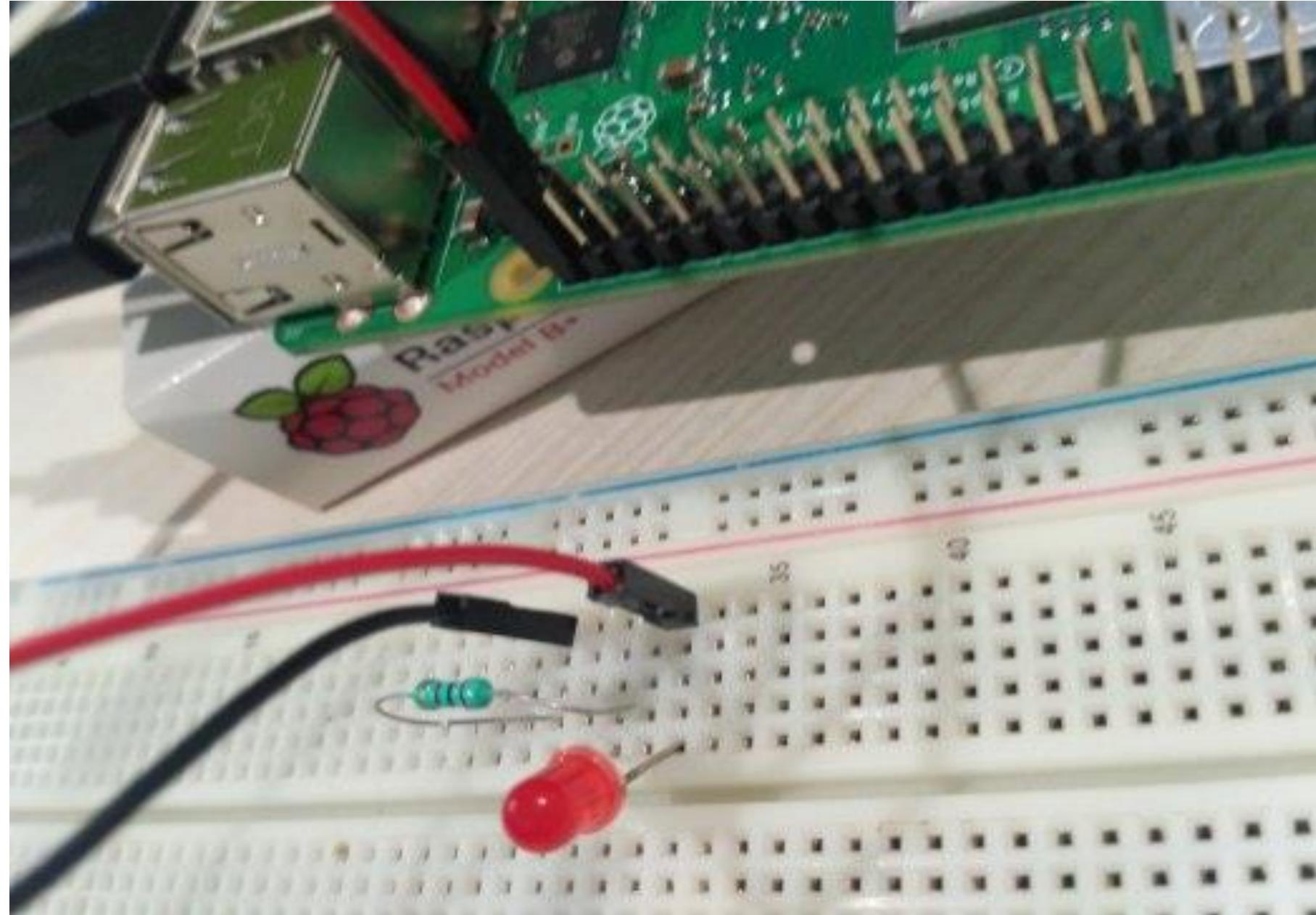
Pi Model B/B+	
3V3 Power	1 2
5V Power	3 4
5V Power	5 6
Ground	7 8
GPIO4	9 10
Ground	11 12
GPIO17	13 14
GPIO27	15 16
3V3 Power	17 18
GPIO10 SPI0_MOSI	19 20
GPIO9 SPI0_MISO	21 22
GPIO11 SPI0_SCLK	23 24
Ground	25 26
ID_SD I2C ID EEPROM	27 28
GPIO5	29 30
GPIO6	31 32
GPIO13	33 34
GPIO19	35 36
GPIO26	37 38
Ground	39 40
5V Power	
Ground	
GPIO14 UART0_TXD	
GPIO15 UART0_RXD	
GPIO18 PCM_CLK	
Ground	
GPIO23	
GPIO24	
Ground	
GPIO25	
GPIO28 SPI0_CE0_N	
GPIO07 SPI0_CE1_N	
ID_SC I2C ID EEPROM	
Ground	
GPIO12	
Ground	
GPIO16	
GPIO20	
Ground	
GPIO21	

## STEP 2 : CONNECT BREADBOARD, REGISTER, WIRE, LED



- Connect **GPIO 21** (Physical **Pin 40** ( 5V: red wire) )
- Connect **PIN 39** (**Ground**: Black)

# ACTUAL CONNECTIONS



# PYTHON CODE 1

```
import RPi.GPIO as IO          # calling header file for GPIO's of PI
import time                   # calling for time to provide delays in program
IO.setmode (IO.BOARD)         # programming the GPIO by BOARD pin numbers.
IO.setup(40,IO.OUT)            # initialize digital pin40 as an output.
IO.output(40,1)                # turn the LED on (making the voltage level HIGH)
time.sleep(1)                  # sleep for a second
IO.output(40,0)                # turn the LED off
time.sleep(1)                  # sleep for a second

#loop is executed second time
IO.output(40,1)
time.sleep(5) # let the LED be on for 5 second
IO.cleanup()
time.sleep(1)
```

# PYTHON CODE 2

```
import RPi.GPIO as GPIO
import time
LedPin = 40
def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(LedPin, GPIO.OUT)      # Set pin mode as output
    GPIO.output(LedPin, GPIO.HIGH)    # Set pin to high(+3.3V) to off the
1
def loop():
    while True:
        print ('LED is on')
        GPIO.output(LedPin, GPIO.LOW)    # led on
        time.sleep(.5)                  # wait 0.5 sec
        print ('LED is off')
        GPIO.output(LedPin, GPIO.HIGH)   # led off
        time.sleep(.5)                  # wait 0.5 sec
```

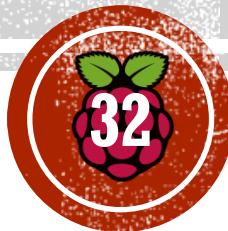
## CODE 2 (CONT..)

```
def destroy():

    GPIO.output(LedPin, GPIO.HIGH)      # led off
    GPIO.cleanup()                      # Release resource

setup()
try:
    loop()
except KeyboardInterrupt: # When 'Ctrl+C' is pressed, the destroy()
will be executed.
    destroy()
```

# FACE DETECTION



# WHAT IS FACE DETECTION?

Face detection refers to identifying a human face in a given frame. The detection is done using a classifier which is trained to detect a face.

# HOW IT CAN BE DONE?

1. Initialise a classifier trained for detecting human frontal face.
2. Read the video frame by frame.
3. Using the classifier detect each face in the frame.
4. Draw a rectangle with the obtained parameters, around the face.

# WHY USE RASPBERRY PI?

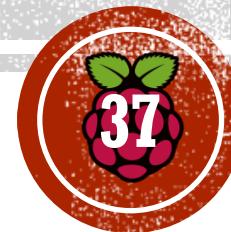
- ❑ Raspberry pi is portable, which gives the advantage of enabling face detection on portable devices.
- ❑ As face detection is the first step towards face recognition (identifying the name of person from database using facial features),
- ❑ hence it also builds up to face recognition which has wide and varied applications.

# REQUIREMENTS

- RPi
- Adapter
- Micro SD Card
- Mouse + Keyboard

# ALEXA (VOICE-ASSISTANT)

By Amazon



# REQUIREMENTS

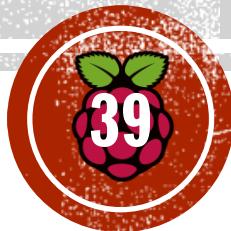
- ❖ RPi with installed OS
- ❖ Microphone and Speaker
- ❖ Adapter and cable for power
- ❖ Micro SD Card
- ❖ Mouse + Keyboard
- ❖ Amazon Developer Account (Free)

Prototyping link:

<https://github.com/alexa/alexा-avs-sample-app/wiki/Raspberry-Pi>

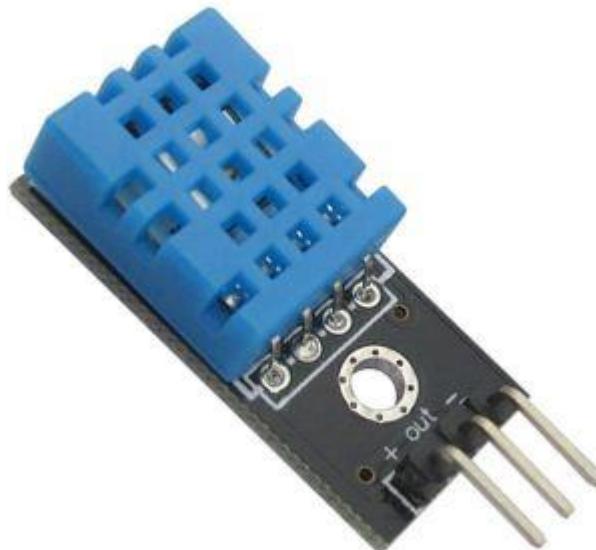
# **DHT11**

## **TEMPERATURE & HUMIDITY SENSOR**



# COMPONENTS REQUIRED

1. Raspberry Pi
2. **DHT11 Sensor**
3. HDMI Cable
4. Power Supply
5. SD Card
6. USB Mouse & Keyboard



# INSTALL ADAFRUIT LIBRARY

```
$ sudo apt-get update
```

```
$ sudo apt-get install build-essential python-dev
```

```
$ sudo git clone https://github.com/adafruit/Adafruit\_python\_DHT.git
```

```
$ cd Adafruit_python_DHT
```

```
$ sudo python setup.py install
```

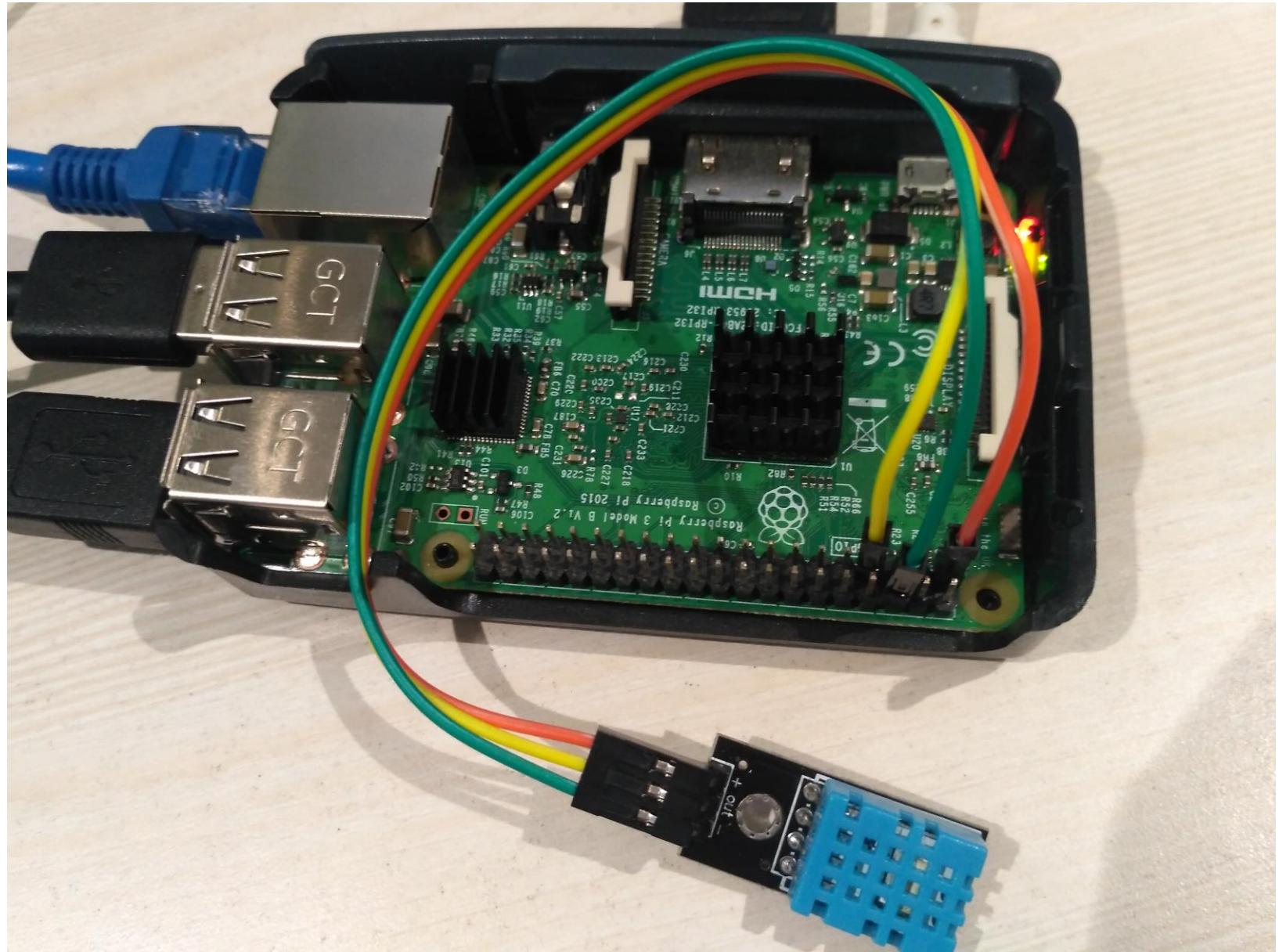
```
import Adafruit_DHT
```

# CIRCUIT CONNECTION

VCC (+) -> PIN 1

OUT (Data) -> PIN 7 (GPIO 4)

Ground (-) -> PIN 6



\$ python dht11.py

# PYTHON CODE: (DHT11.PY)

```
import Adafruit_DHT
sensor=Adafruit_DHT.DHT11 # Set sensor type : Options are DHT11,DHT22 or AM2302
gpio=4 # Set GPIO PIN for sensor
# Use read_retry method. This will retry up to 15 times to
# get a sensor reading (waiting 2 seconds between each retry).

humidity, temperature = Adafruit_DHT.read_retry(sensor, gpio)

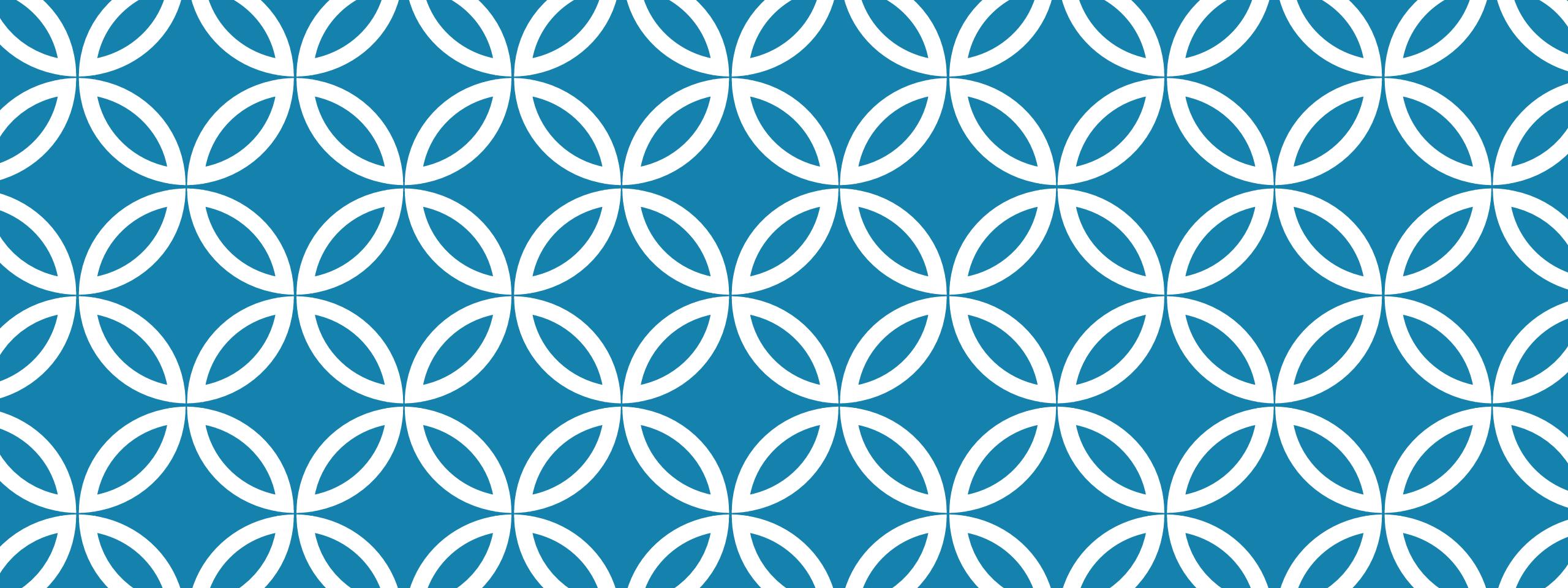
# Reading the DHT11 is very sensitive to timings and occasionally.
# the Pi might fail to get a valid reading. So check if readings are valid.

if humidity is not None and temperature is not None:
    print('Temp={0:0.1f}*C Humidity={1:0.1f}%'.format(temperature, humidity))
else:
    print('Failed to get reading. Try again!')
```

44

# THANK YOU

Gaurav Singal



# BLOCKCHAINS FOR IOT

**Ken Birman**

# BLOCKCHAINS FOR IOT



*Lucas Mearian. Not afraid  
of hyperbole!*

## **What is blockchain? The most disruptive tech in decades!**

“The distributed ledger technology, better known as blockchain, has the potential to eliminate huge amounts of record-keeping, save money and disrupt IT in ways not seen since the internet arrived.”

Lucas Mearian, ComputerWorld Staff Writer

# A MORE TECHNICAL ANSWER?

“A blockchain, originally block chain, is a growing list of records, called blocks, which are linked using cryptography. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data (generally represented as a Merkle Tree root hash).

By design, a blockchain is resistant to modification of the data.”

Wikipedia

# BEST USES FOR BLOCKCHAIN?

Secure, trustworthy shared log (append-only file)

Records describe some form of announcement or transaction of broad value to the users. There are standard (web-like) languages for recording things in these records.

Example: Bank announcements of variable mortgage loan rates. Public announcements of weddings, divorces, births, deaths.

# BEST USES FOR BLOCKCHAIN?



# BEST USES FOR BLOCKCHAIN?

If the blockchain is public, shared and tamperproof there can never be any basis for disagreement about the information.



The blockchain is publicly replicated to ensure that even the bank can't cheat.

Cryptography prevents tampering

# ... OR AIRPLANE MAINTENANCE LOGS

The New York Times

## *Hundreds of Planes Are Stranded in Russia. They May Never Be Recovered.*

Western companies that own the planes face little prospect of getting them back, meaning billions of dollars in losses.

f    g    t    e    s    a    b



While a few planes may have been recovered abroad before international flights were halted, they are of little use to their owners without the meticulous maintenance records that accompany every aircraft and are often stored by airlines themselves, experts said. And the longer a plane is stuck in Russia, the greater the concern that work on the jet's body, engines and flight systems may not be logged, causing its value to plummet.

“Unless you have those records, the aircraft is virtually worthless,” said Quentin Brasie, the founder and chief executive of ACI Aviation Consulting. “They’re literally more important than the asset itself.”

# TERMINOLOGY

In BlockChain settings, a *transaction* is a digital record describing some event.

Some transactions are complete and self-contained.

But BlockChain also supports transaction languages in which one transaction might refer to events defined by a past or *future* transaction.

# CRYPTOGRAPHIC HASH

A cryptographic hash is a bit string computed from some block of data in a manner that yields a constant-length result irrespective of the data size, and yet such that it would be infeasible to find other data that would hash to the same result.

There are a number of hashing schemes. A highly robust one is SHA-256. SHA-512 is even stronger. MD5 and SHA-1 have been compromised and are unsafe.

# EVEN FASTER HASH METHODS EXIST

(single core performance, all Golang implementations, see [benchmark](#)).

BenchmarkHighwayHash	11,986 MB/s
BenchmarkSHA256_AVX512	3,552 MB/s
BenchmarkBlake2b	972 MB/s
BenchmarkSHA1	950 MB/s (insecure)
BenchmarkMD5	684 MB/s (insecure)
BenchmarkSHA512	562 MB/s
BenchmarkSHA256	383 MB/s

Note: the AVX512 version of SHA256 uses the multi-buffer crypto library technique as developed by Intel, more details can be found in [sha256-simd](#).

<https://blog.minio.io/highwayhash-fast-hashing-at-over-10-gb-s-per-core-in-golang-f938b5218a>

# HARDWARE CAN GET EVEN FURTHER

FPGA and ASIC solutions can be purchased that will run SHA-256 or SHA-512 at speeds of 25,000 to 30,000 MB/s

In some parts of the world there are entire datacenters equipped with huge numbers of these accelerator solutions. China dominates the business.

Very hard to be a BlockChain miner with a single desktop computer today!

# CRYPTOGRAPHIC SIGNATURE: ENCRYPTED HASH

Given a message, anyone can compute the cryptographic hash for it

Some applications need a way to sign a document. It is important that this kind of signature be “irrefutable”.



For this, we first compute a hash, then encrypt it in a special way so that only the signatory could encrypt it, yet anyone can decrypt and check it.

# PUBLIC/PRIVATE KEY PAIR (NORMALLY, RSA)

This is a classic cryptographic method.

RSA creates two “keys”, both just long numbers together with a modulus  $n$  that itself is a product of two very long prime numbers. Call them  $K$ ,  $\bar{K}$

One is designated as the public key and shared. You keep the other private.

$$RSA_K(RSA_{\bar{K}}(X)) = RSA_{\bar{K}}(RSA_K(X)) = X$$

# WHY RSA WORKS

*In 1796, Gauss came up with the theory that ultimately gave us the (very simple) RSA technology. Gauss himself didn't suggest this application.*



In RSA encryption and decryption are just mathematical steps that involve a form of “bignum” arithmetic (modular exponentiation), performed block by block.

RSA is secret because there is no known method for factoring a giant composite number that might have 1000's of binary digits. If we could factor the modulus, it would be trivial to recover the secret key from the public one.

Quantum computers *might* offer a path to doing so, but it would require devices with millions of qubits, way beyond anything feasible anytime soon.

# RSA STRENGTHS, WEAKNESSES

Very widely supported, basis of most “certificates” used in the Internet.

Many tricks exist, based on commutativity of RSA computation. Basically, for any two RSA keys, A and B,  $\text{RSA}_A(\text{RSA}_B(M)) = \text{RSA}_B(\text{RSA}_A(M))$ .

But RSA is fairly slow. The speed is a function of the data size. We don’t casually encrypt entire messages with RSA: it would be feasible but slow.



# HOW WOULD PROCESS P SIGN MESSAGE M?

1. Compute the SHA-256 hash of  $M$ .
2. Now use P's private key to encrypt the hash:  $\text{SHA}(M)_{\text{private-key-of-P}}$

Process Q can easily verify that  $M$  has not been tampered with:

1. Q recomputes the SHA-256 hash for  $M$
2. Now Q uses RSA with P's public key to crypt P's signature.
3. If they match, then Q has confirmed that  $M$  hasn't changed.



# NOTARIZING BLINDED DATA

There is even a method, by David Chaum, for signing an object that the signatory cannot see. It would be useful for secure voting:

- Prepare your ballot, then blind it (encrypt) and obtain a signature.
- The signature is proof that your vote was valid and only cast once. Submit it for counting now, unblinded, via a secure anonymous “onion route”
- The ballot itself has no identifying information, and neither does the signature. So a third party can see that your vote is valid, and can count it, and yet can't learn how any particular individual voted.
- Chaum also showed how to get a receipt which can be used to be sure your vote was properly tabulated.

# PARALLELISM?

A further win is to maximize parallelism and reduce record sizes.

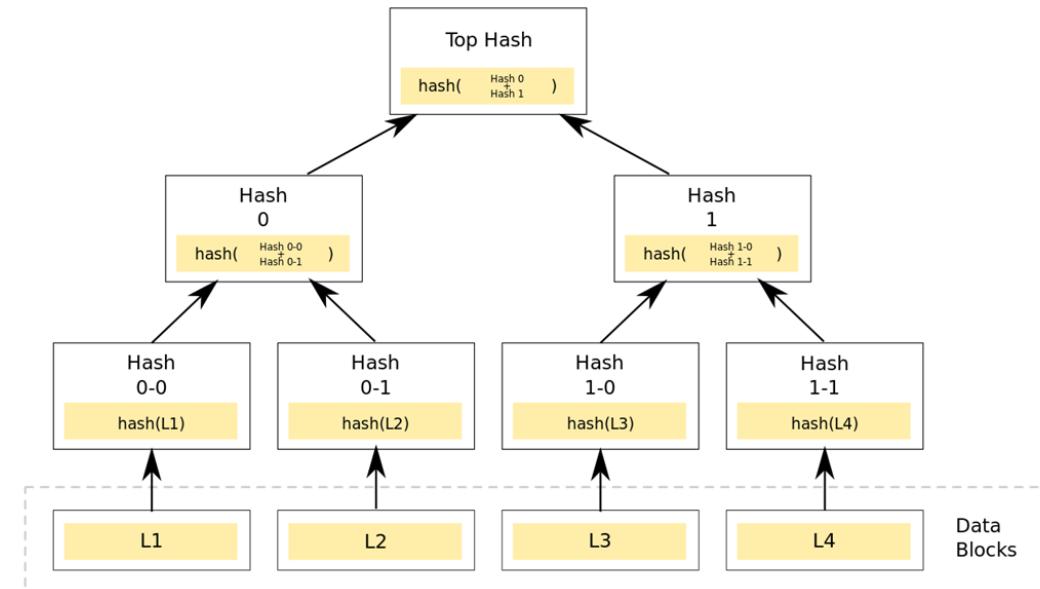
In the case of BlockChain, each block contains a set of transactions represented as binary records. These records might be huge, hence slow to hash (and very slow to encrypt, were you to try that).

By having the creator store the record someplace reasonable and then just storing signatures in the BlockChain, we use it as efficiently as possible.

# MERKLE TREE: A TREE OF SIGNED RECORDS.

Rather than making one list of  $N$  records and then hashing them, we often create a binary tree of hashes.

Very common in BlockChains, permits us to run SHA-256 or SHA-512 in its fastest “mode” of operation.



Often we think of the entire Block chain as a sequence of Merkle trees that change (only) by appending new subtrees (which also changes the root node)

# THIS ALREADY GIVES US A BASIC SOLUTION!

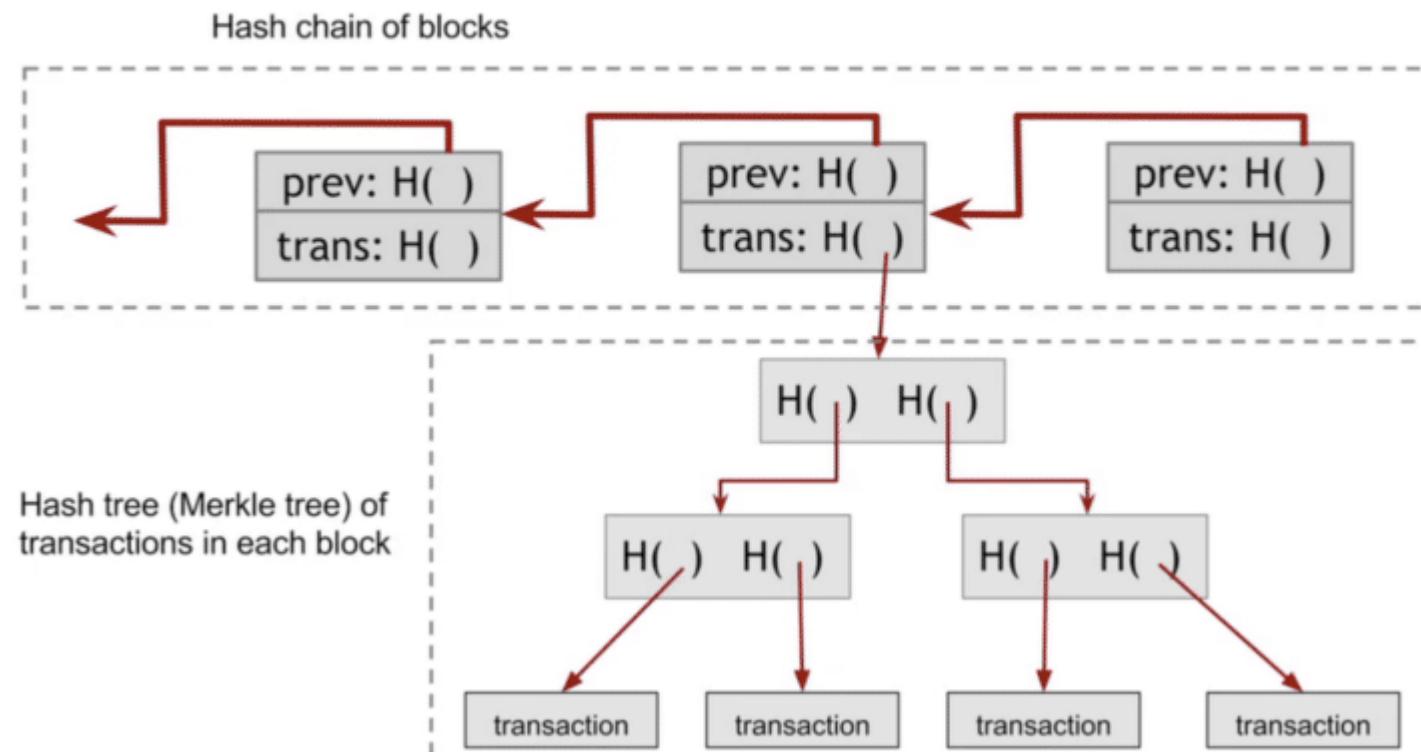
Compute a series of records, each containing transactions signed by the initiator. The record needs to include the “name” of the initiator so that anyone needing to do so can look up the matching public key.

Associate each record with a key for lookup, and insert the (key,record) objects into the Merkle tree.

Then create some form of cryptographic proof that the new tree extends the prior tree. Logs are just one possible representation.

# OUR BASIC SOLUTION

## Bitcoin block structure



<https://coincentral.com/merkle-tree-hashing-blockchain/>

# WHY IS THIS SECURE?

If anyone tampers with any record in the chain, we can sense this by recomputing the Merkle tree. The signature won't match.

To verify the entire chain, block by block recompute the Merkle tree, then recompute the sequence of pairwise hash values.

***Verification is required when an untrusted BlockChain is initially loaded.***

# PERMISSIONED/PERMISSIONLESS

BlockChain solutions split into two categories.

A **permissioned** BlockChain is managed by an authorized group of servers, which could be geodistributed or inside some datacenter. We generally assume that they don't have true Byzantine faults.

A **permissionless** BlockChain is managed by an anonymous group of servers that volunteer to play the role, and might come and go at will.

# **... AND TWO MORE SUB-CASES FOR THE PERMISSIONED CASE**

**Permissioned blockchains** can arise in two settings.

- In a data center, just use Paxos plus logic to compute the needed cryptographic signature chain. WAN mirrors protect against attempts to compromise the whole data center and rewrite the whole log.
- At geoscale, a permissioned blockchain just means the miners get an authorization to mine – it can be withdrawn if they misbehave.

**When people say “permissioned blockchain” they normally mean “in a wide-area environment, with Byzantine behavior controlled by the permissioning key-management technique.”**

# ATTACKS ARE AN ISSUE!

Wide-area replicated BlockChains can come under many forms of attack.

- Compromised server might try to hand out “fake” versions of the chain.
- It might try to generate huge rates of transactions on its own, and not include your transactions. This is a form of DDoS attack.
- It could try to corrupt individual transactions or records.

# PROOF OF WORK

A Proof of Work mechanism adds one more field to the blocks: a “nonce”.

The nonce is just a bit string of some size.

The rule is that to append  $B_{k+1}$  to the chain, in addition to hashing it with the hash of the prior block, P must also find a nonce such that when the nonce is included and a new hash is computed, the hash value ends with some desired number of 0 bits.

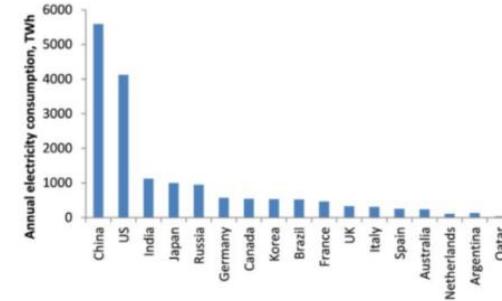
# PROOF OF WORK

Finding such a nonce is hard work!

So while P could be keen to append its block, it may need to search for this nonce for many seconds or minutes.

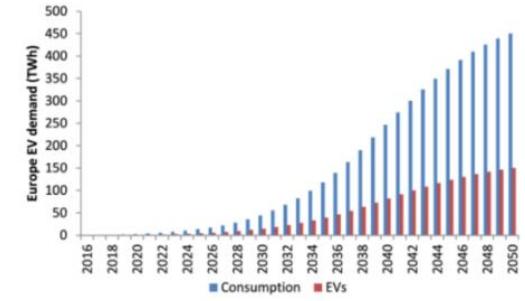
The difficulty of finding a nonce value that will work prevents DDoS attacks.

**Exhibit 1:** According to the IEA, global electricity consumption is c.22k TWh annually, with cryptos annualising nearly 40TWh (in line with Qatar), but could get to c.125TWh in 2018 (in line with Argentina)



Source: IEA 2015 data, Morgan Stanley Research

**Exhibit 2:** To put this in to context, we see EV electricity consumption in Europe at 1-2TWh presently, and 25TWh by 2025, with global EV demand at 125TWh only by 2025



Source: Morgan Stanley Research estimates

# **... OTHER IDEAS HAVE BEEN TRIED, BUT ARE NOT VERY SUCCESSFUL**

For example, proof of “stake” centers on the idea of doing mining in a way that favors wealthy miners who can prove they are big players.

The “stake” is like a printout of a bank account. “I own fifty buildings in New York City, seven golf courses, a hotel chain... I mined more blocks than Bill Gates and Jeff Bezos combined. I am “Mr. Big.” So. you can trust me to mine blockchain blocks (and to keep all my profits from doing that”).

Not very democratic. And so, not very popular!

# HOW MOST BLOCKCHAINS WORK TODAY

You can download the software as a VM or even compile it yourself.

You launch the code on your own servers – this makes you a “miner” as soon as the system has initialized itself.

***The system downloads the entire current BlockChain, from other machines already running the BlockChain software (there is a web site listing some you can contact for copies).***

## ASIDE: WHY THE WHOLE TREE?



There is a lot of research (meaning, a huge amount) on ways to download and verify just a portion of the blockchain. None is really secure, yet.

In some settings this is going to be absolutely obligatory, but right now it isn't how the big public blockchains work.

The most promising approaches provide “countersignatures” for the portion you want, provided by authorities you happen to trust.

# NEXT, YOU VERIFY THE BLOCKCHAIN

You'll need to recompute all the Merkle trees and the chain of hashes.

In fact this may not take enormously long... today. Few BlockChains have huge amounts of content.

But someday, we might have BlockChains with hundreds of billions of records and total sizes in the petabytes. Then download speed and verification time and storage will become an issue!

# MEANWHILE, NEW BLOCKS ARRIVE

Each block extends some specific sequence of prior blocks.

If you turn out to have downloaded the wrong sequence, you may have to truncate your chain and download the longer sequence.

This is a “rollback”. During startup, substantial rollbacks can occur. Later they shouldn’t (assumes a fully connected network of mostly “correct” miners).

# AT THIS POINT YOU CAN CREATE TRANSACTIONS

So, you open for business.

Someone shows up to buy a glass of your fresh lemonade.

You'll generate the transaction (think "credit card payment slip") and submit it to the system. It enters a pool of pending transactions.



# WHEN WILL YOUR TRANSACTION GO THROUGH?



Within an hour or so, you should see that your transaction got included into some block, and also that everyone seems to have adopted that block.

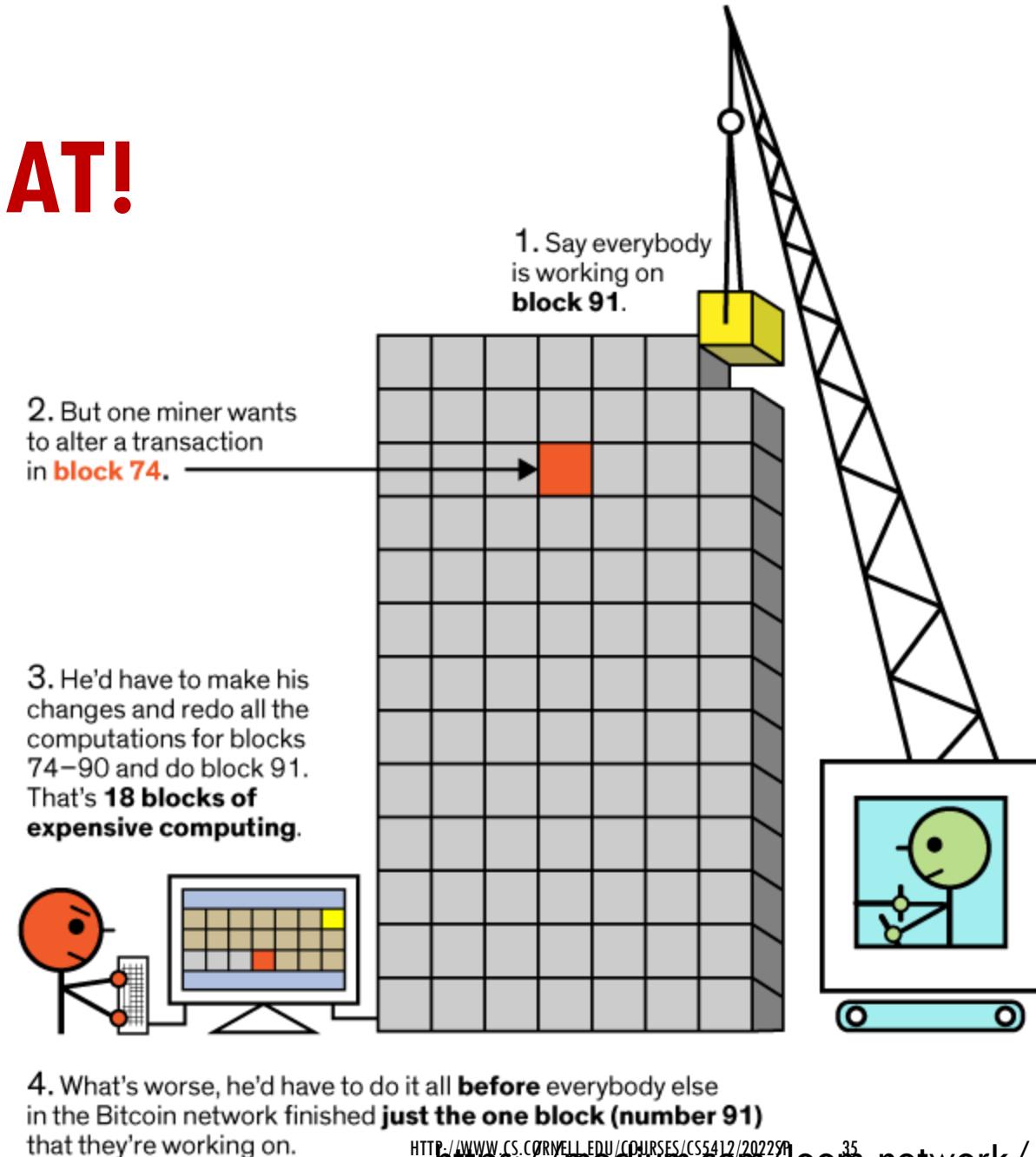
The chain has moved six or more blocks into the future.

So now you can hand that glass of frosty bliss to your happy customer!

# BUT NOBODY CAN CHEAT!

To modify a past record you need to also modify every signature subsequent to that record.

The step where you have to find these nonce values will be very slow and you'll lose the race.



# WHAT IF THE ATTACKER IS A COUNTRY?

A country could build whole datacenters, equipped with hardware to compute SHA-256 at ultra-high speeds.

In this case P (using the datacenter) could generate a lot of blocks quickly, for which they would be paid. Or could have an entire second BlockChain starting from months ago, and *longer than the official main one*.

To prevent most such attacks, BlockChain solutions make the proof-of-work task harder as a function of the rate at which blocks are being found.

# WHAT IF THE ATTACKER IS A COUNTRY?

In effect, if P controls enough computing power, he can “gain control” of the BlockChain. The proof-of-work can become so hard that only P has the compute power to solve the puzzle!

P could then refuse to post some transactions, or cause trouble in other ways.

But this form of attack has not (yet) been seen.

# WHAT ABOUT RACES?

Permissionless BlockChains are at risk of a “race” situation in which one group of miners is working to append record R, and some other group, record S. A tie can easily occur.

BlockChain systems “adopt the longest chain” (may the best miners win). This can cause a rollback if a few blocks were appended by group A, but then group B suddenly publishes a longer extension.

In practice, rollbacks longer than 6 blocks are never observed.

# IN CONTRAST, PERMISSIONED BLOCKCHAIN DOESN'T NEED PROOF OF WORK

A permissioned system is operated by known, trusted, authorized servers.

They won't attack the chain by trying to overload it with transactions in an unfair way, and they would charge for any transactions they append on behalf of external clients.

So we can avoid this costly step with datacenter BlockChain solutions.

# WHAT IF YOU DON'T REALLY TRUST THE PERMISSIONED PROVIDER?

We can mix methods: a global “proof” with a local “data store”

Our permissioned provider can commit to some form of cryptographic root of each new version of the log (or tree), and to a proof that the new version extends the old version.

The “commit” is broadly shared and pins the provider down. Then for an append or a query, the provider can be asked to also provide a proof that they did the append, or that the query response is correct & complete

# WHAT'S IN A TRANSACTION?

Some BlockChain systems are very rigid. For example, a BitCoin BlockChain record can only support a few operations on BitCoins.

These represent transactions: Ken sells Ittay a packet of gum for 10฿

In a permissionless scheme, Ken would probably wait for a while before handing the gum to Ittay. With permissions, rollback risk can be reduced or even completely eliminated.

# FANCIER TRANSACTIONS

There are several standards for encoding fancier “digital contracts” into records suitable for BlockChain.

One, called HyperLedger, uses HTML as its underlying “language”.

A second, Ethereum, has a sophisticated language of its own, and can even encode computational tasks into the transaction record.

# ONE ISSUE: VALIDATION

In existing blockchain systems, every participant maintains a replica of the entire blockchain.

But as blockchains grow, this could become unworkable. A blockchain might be hundreds of gigabytes long... users will want to download portions

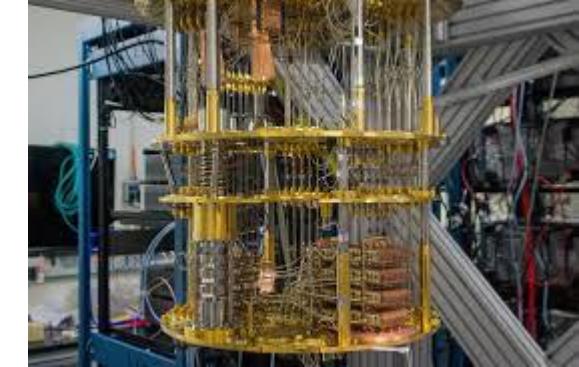
If a user only downloads the record they are seeking, how can the system validate that the *entire chain* is intact and properly signed?

# MORE ISSUES

With permissionless BlockChain, is it really “safe” to trust that after six blocks have been appended, the chain won’t roll back and invalidate my transaction? (“When should Ken give the lemonade to Sally?”)

If a smart contract references future events, what would be the “semantics” of that contract, in a PL sense? Does the meaning depend on waiting for the future to occur? Can chains of dependencies arise, or contracts that are undecidable, or infeasibly complex to “evaluate”?

# A REALLY BIG ISSUE



Quantum computers are advancing rapidly.

But many blockchains use cryptographic techniques known to be at risk if practical quantum computers emerge and can deal with really large quantum computations (on “bignum” numbers with thousands of digits).

People are asking: Is this becoming a real risk?

# WILL QUANTUM COMPUTING BREAK CRYPTOGRAPHY?

Which is closer to the truth?

A quantum computer can make non-deterministic guesses, check to see if any are right (like guessing the factors of an RSA key), and then output the correct one.

- A quantum computer can compute a near-infinite number of discrete fourier transforms “concurrently”, but you can only read out one data-point of the result at a time.

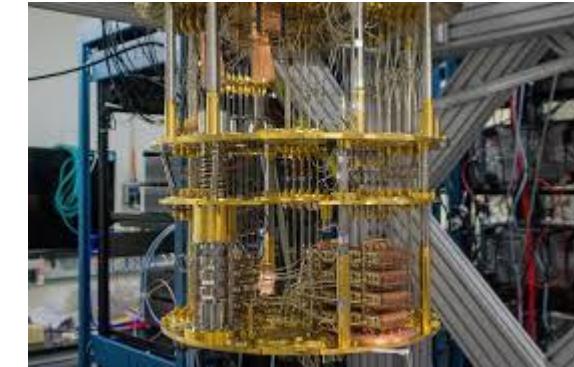
# PUBLIC MISUNDERSTANDING

Popularity of the “many worlds” interpretation of physics has clouded the public conception of what a quantum computer can do!

In fact many worlds could be a valid model, for the most elementary level of Planck-scale physics (the layer where people talk about membranes and string theory, and loop-quantum gravity).

But our macroscopic (“causally emergent”) world is very remote from that most basic layer of physical reality.

# SHOR'S ALGORITHM



To factor RSA, Shor's algorithm requires a special circuit specific to the size of the keys.

Then we input “all possible” n-bit integers, where n is the key length, like 1024. This involves a “coherent entanglement” of n qubits. But due to errors, qubits rapidly decohere. Error correction will require vastly more qubits, and nobody is sure how many. Perhaps millions or billions.]

The entangled data is then transformed by the circuit, which computes a DFFT

# READING THE OUTPUT

You read the output of a quantum computer by setting up the experiment again and again and then repeatedly extracting a single sample.

Over time, the values you read build up to a kind of probability density image, like a photo created pixel by pixel.

In the case of Shor's algorithm this photo shows peaks that hint at the values of the factors. Now you can search for the factors close to those peaks. Quality of the search will depend on the sharpness of the peaks.

# A LOT OF ASSUMPTIONS!

Nobody knows how quantum error correction “scales”. Today it works for 3 to 5 q-bit entanglements, at best.

Nobody knows how complex a computation we can perform without destroying coherence. In fact these quantum DFFT operations must be reversible in order to remain coherent, and hence perfectly precise.

Nobody knows how quickly we can set up such a run and sample it.

Nobody knows how sharp the peaks will need to be as a function of key length.

# AND WORST OF ALL...

*Unfortunately, neither Euler nor Ramanujan really looked closely at this question!*



Nobody knows if factoring large numbers is even a “hard” problem!

True, we lack a fast solution today. But the complexity of factoring is unknown.

But perhaps some numerical savant will find a solution... with classical computers! The same goes for finding a nonce with the desired hashing properties to mine blocks...

# THE ENTIRE EDIFICE COULD COLLAPSE!

If you bet heavily on BlockChain,  
you are betting that people will  
figure out a way to ensure that  
it won't yield to some kind of attack.

But in fact this is just a bet, today.



# PROVABLY SECURE SYSTEMS



There is a theory of **semantic cryptography** safe against quantum attacks. It was developed by Goldwasser and Micali, who won the Turing Award for the insight.

They proved that secure encryption schemes must be probabilistic, rather than deterministic, with many possible encrypted texts corresponding to each message.

The **Goldwasser–Micali** (GM) lattice cryptosystem demonstrates the idea.

# COULD A BLOCKCHAIN USE LATTICE CRYPTOGRAPHIC TECHNIQUES?

At present, lattice cryptography is too computationally slow for practical use, and also causes too much “inflation” in the size of data.

Each bit in the data becomes a point in a very high dimensional space, leading to a billions-to-one increase in message sizes.

But continued research may yield much more compact solutions with the same properties. A new research initiative just started on this topic.

# SUMMARY OF BLOCKCHAIN CONCERNS

Permissioned or Permissionless? Energy cost of permissionless block mining.

If the BlockChain gets really large, costs of downloading a copy.

Cost of verifying that the BlockChain hasn't been tampered with.

National-scale “disruption” scenarios that cause massive rollbacks, chaos.

Accidental loss of some chunk of the chain, making verification impossible.

Smart contracts might be too smart for their own good.

# MORE CONCERNS



<https://www.joe.ie/news/pics-this-pile-of-cash-worth-22bn-was-found-inside-the-insane-home-of-a-mexican-drug-lord-409313>

Today's most enthusiastic Blockchain use cases seem to center on a mix of illegal transactions, money laundering, and a gigantic technology boom but without much of a "market" for the associated products.

The model also depends on some hardness assumptions: finding a nonce, factoring RSA key. Quantum computers could shake up these assumptions.

Unresolved privacy concerns: "everything is on the table."

# WALKING THROUGH THE ISSUES

Which issues would arise on a “smart farm”?

Would a BlockChain solve those issues? What new risks would it introduce?

What limits the speed of new technology adoption?

# BLOCKCHAIN ON A FARM

Main uses seem to be for audit trails of various kinds:

- Capture data about something we are supposed to trace or record.
- Write it digitally into the ledger, securely. Tamperproof and automatic
- Auditors given access to the record.

But they will want to know:

- Why should I trust this BlockChain record? Is the underlying data valid?

# BLOCKCHAIN FOR SENSORS

A smart farm would store BlockChain records that include sensor data.

Very likely the sensors themselves would be securely connected to their host machines, for example using Azure's IoT Hub or the AWS equivalent.

Reminder: With an IoT Hub model, the sensor itself uses security keys, and will only connect and talk to the hub. This enables a *digital twin* concept.

# TRUST WITH SENSORS

... so we can assume the connection to the sensor is secure. But how would a digital twin for a farm work?

Azure IoT Hub will only allow authorized sensors to be part of the system, and it patches the software and configuration automatically. Feeds events to the Azure IoT function server, where functions consume them.

So presumably these functions log the event records to the Blockchain.

# QUESTIONS THIS MIGHT NOT ANSWER

Is this sensor the correct one for the information the application claims to have captured?

Is the sensor working correctly?

Has the data the sensor generated been modified before it was logged?

# CAN WE ANSWER THE QUESTIONS AN AUDITOR MIGHT ASK?

A sensor records shows that cow 2143 was milked on Tuesday at 10am. Later the milk turned out to have a dangerous bacteria in it, like Listeria. It got through and a consumer became quite sick.

Was she properly clean when she was milked? Had she been evaluated for mastitis as required by the health department? Was she periodically checked for overall health? Did she receive any “off records” meds?

# CAN WE ANSWER THE QUESTIONS AN AUDITOR MIGHT ASK?

Realistically, an audit using sensor data would provide “evidence” but can’t answer these questions. Non-technical people might find this surprising, but in cloud computing we have seen why many either cannot be answered, or we *can’t have confidence our answers will be correct*.

BlockChain does protect against tampering, and does record what the sensors reported, and when. This is already valuable, as long as we are honest about the capabilities and limitations.

# TO HAVE REAL CONFIDENCE...

Azure IoT Hub would need to log **management** events too.

The audit-trail examination tool would need to visualize this information and be able to convince the auditor that yes, this is the proper sensor, it seems to be properly calibrated, the data wasn't tampered with...

The mention of time suggests that we might also need to log events related to the way the system tracks time, or at least have a “story” there.

# BLOCKCHAIN-SPECIFIC FORM THIS TAKES?

We noted that early adopters are people with transactions to carry out anonymously, or maybe with money to launder. Strongly motivated mostly because for now, Blockchain feels like a way to evade oversight and taxes.

Business community has many people keen to adopt the next new thing. Startup frenzy and huge fortunes made on ICOs adds fuel to the flames.

But farming is a case for mainstream use and these questions need to be answered. This is why the mainstream technology community is more cautious.

# **LET'S LOOK AT A BLOCKCHAIN CREATED SPECIFICALLY FOR IOT**

Cornell “smart farms” research effort (CIDA) is highly visible.

Led by Susan McCouch, Hakim Weatherspoon, Steve Wolf and Abe Strouck.

One early accomplishment: Vegvisir, a BlockChain specifically for agriculture.

# SOME ISSUES THEY THOUGHT ABOUT

A lot of the “events” that matter in an agriculture or farming setting are in remote places, disconnected from the main system.

The BlockChain would probably be used primarily as an audit trace, to track events in the food chain from farm to table.

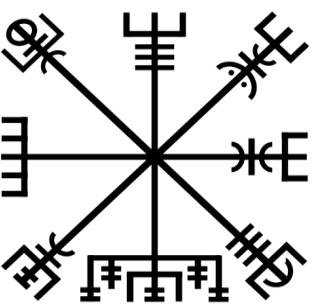
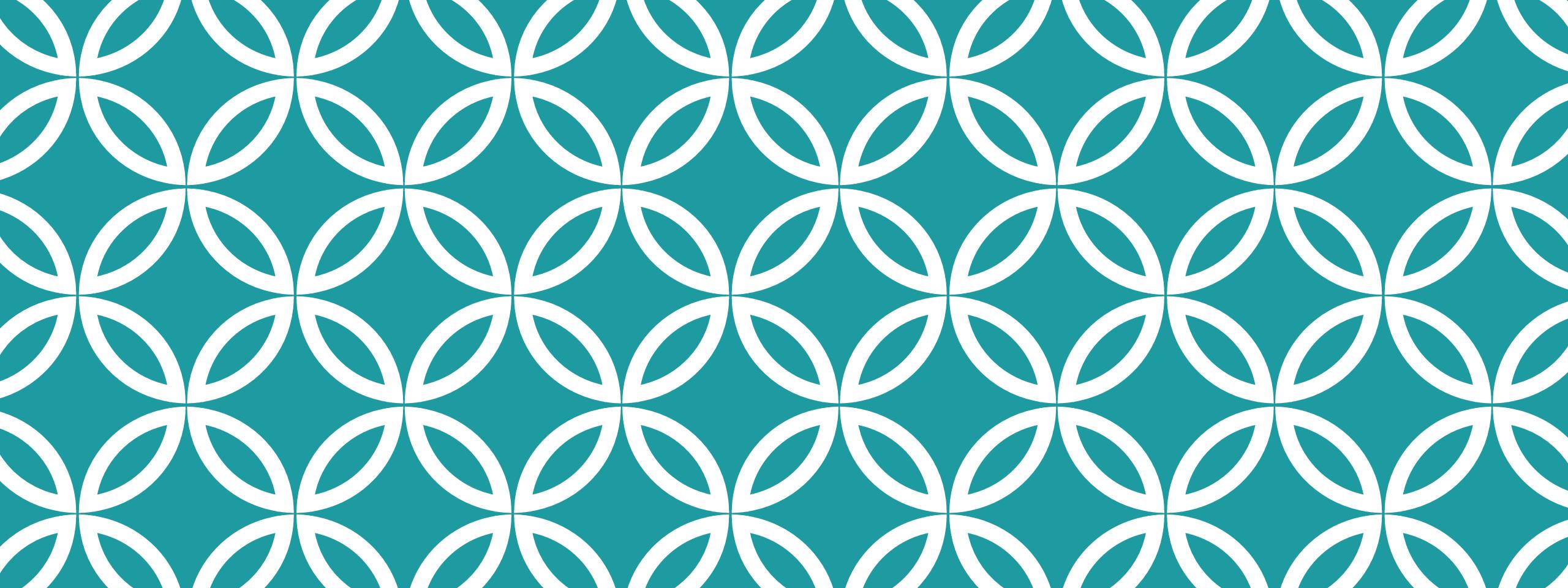
So this raises issues like intermittent connectivity, how we know that the sensor that generated a record is the “correct one” for that role, etc.

# **CONNECTIVITY: JUST ONE ISSUE OF MANY!**

Vegvisir is a research project and a proof of concept, but not deeply integrated with Azure IoT Edge.

Any real product will need more ties to the Azure infrastructure.

But an Azure Blockchain would also benefit: as a part of the official Azure ecosystem, we might gain better answers to some of the trust issues!



# VEGVISIR

Slides from Robbert van Renesse

Talk presented at ICDCS 2018

# A BLOCKCHAIN FOR THE FOOD SUPPLY CHAIN

Robbert van Renesse

joint work with Hakim Weatherspoon, Danny Adams,  
Kolbeinn Karlsson, and Stephen B. Wicker

Initiative for Crypto-Currencies and Contracts (IC3)  
Cornell Digital Agriculture Initiative

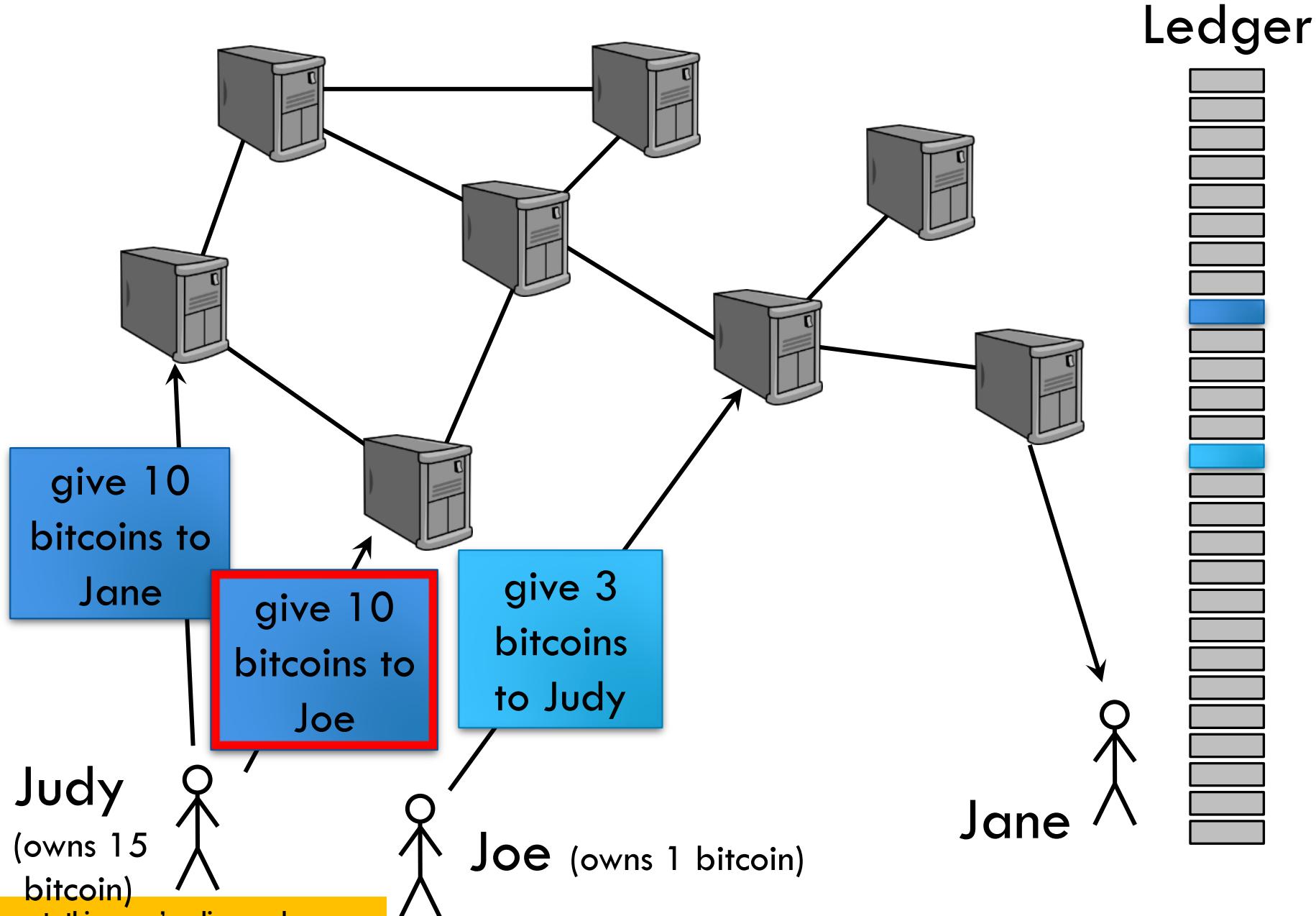
# BLOCKCHAIN'S PROMISE

## Promises

- Global currency
- Smart contracts
- Notarization
- Accountability
- ...



# A REPLICATED LEDGER OF TRANSACTIONS



# SMART CONTRACTS

*Smart contracts* are executable programs on the BlockChain, take input from the BlockChain, and produce output on the BlockChain

Main use: *automated escrow*, where disbursement depends on agreed upon conditions

Caution: Smart Contracts have been found to be prone to (very expensive) bugs

# POTENTIAL USE CASES

Killer app: cryptocurrencies

Other potential uses:

- Reduce opaqueness of supply chains
  - One “trustless” place for all transactions along the way
  - Improvements over paper-based systems and many disjoint databases
- Eliminate middlemen
  - Why does farmer make so little and consumer pay so much?
- Reduce fraud
  - India, Russia, Sweden, Georgia... are building blockchain-based land registries to fight “land fraud” and simplify international property transactions

# FOR THE FOOD SUPPLY CHAIN?

Supply chain management

- Walmart is building one for the food supply chain
  - Food safety: fast identification of tainted foods
  - Consumers are demanding more information about the products they buy (organic, fair trade, ...)
- Simplify international transactions

Help farmers

- Want to know what happens to their products for fair pricing
- What products should they be producing?

Reduce food scandals

- illegal production, misrepresentation, loss and waste, ...

# INDUSTRIAL UPTAKE?

ripe.io:

- A company that is building a “blockchain of food” with IoT interfaces

Walmart:

- partnered with IBM and Tsinghua to identify sources of contaminated products and speed up recall

But today's blockchain technology may not be appropriate for all use cases

- too dependent on availability of plentiful power, networking, and storage

# DESIRED BLOCKCHAIN PROPERTIES

## *Performance:*

- High Throughput, Low Latency
- Energy-Efficient

## *Security:*

- Always available for reading (verifying) and appending
- Fair
- Tamperproof (Integrity)
- Possibly confidentiality as well

## *No Single Administrative Domain*

- *no need to trust a single provider*

## *Open membership (or not)*

# OPEN MEMBERSHIP IS HARD

Traditional secure logs are based on voting

Members vote on which transactions to add to the log and in what order

**Problem: “Sybil” or impersonation attacks**

- a participant may try to vote multiple times
- with closed membership, cryptographic signatures can identify the source of a vote
- with open membership, anybody can create identities and that way vote many times

# PERMISSIONLESS VS PERMISSIONED BLOCKCHAINS

	Permissionless	Permissioned
Approach	Competitive	Cooperative
Basic technique	Proof-of-Resource	Voting
Membership	Open	Closed
Energy-efficiency	Often terrible	Excellent
Transaction rate	At best hundreds / sec	Many thousands per second
Transaction latency	As high as many minutes	Less than a second

# BITCOIN BLOCKCHAIN

Permissionless, open membership

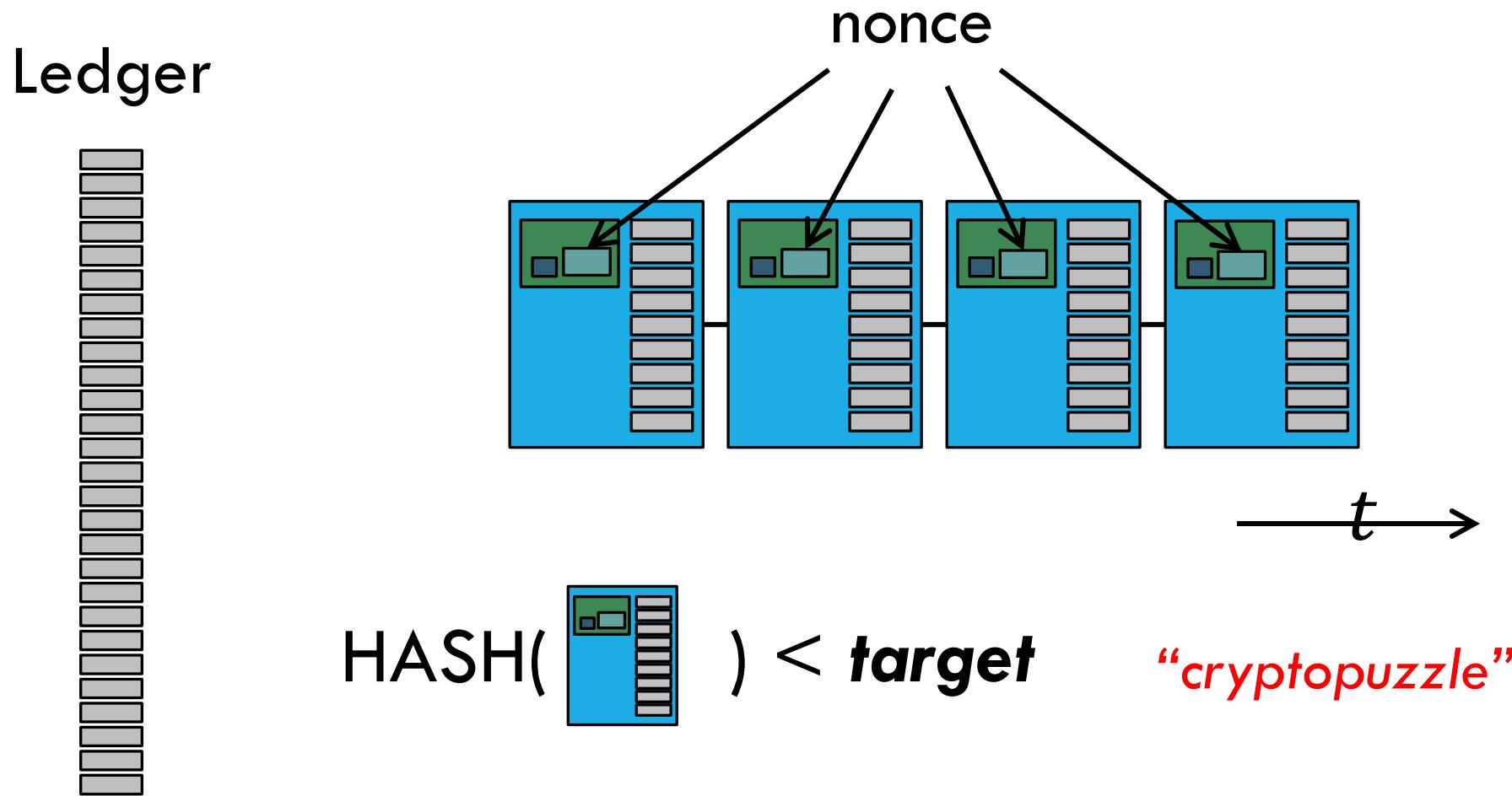
Proof-of-Work

There are thousands of Bitcoin miners

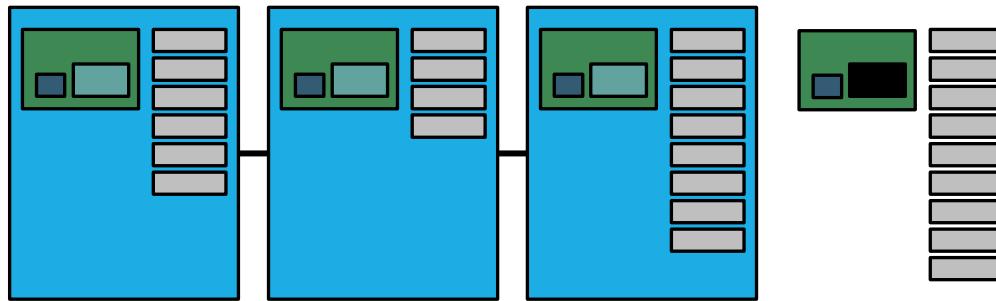
- they use ASIC hardware to compute SHA256 hashes
- use about more energy than the country of Denmark

Overall rate is a few transactions per second

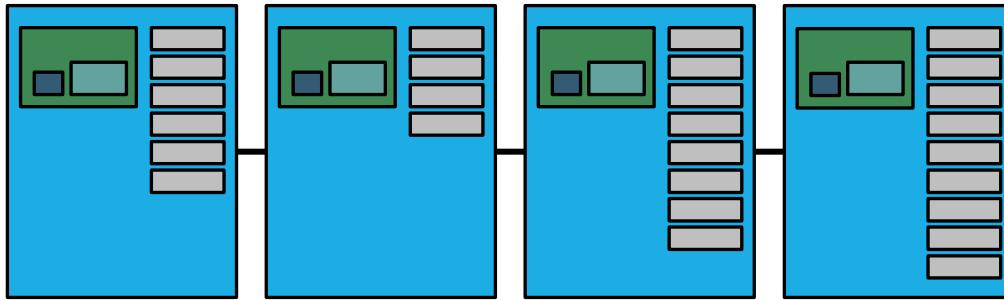
# THE BLOCKCHAIN



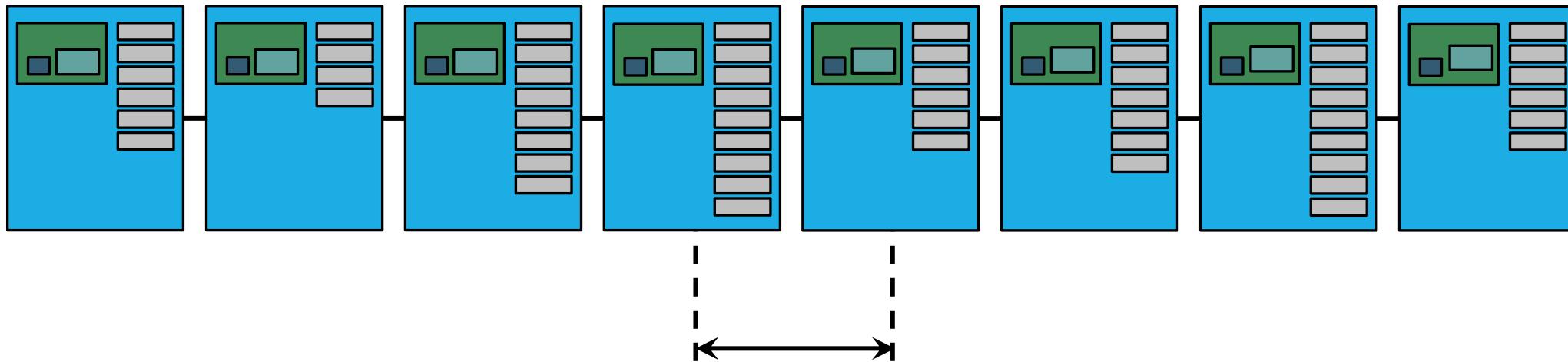
# THE BLOCKCHAIN



# THE BLOCKCHAIN



# THE BLOCKCHAIN



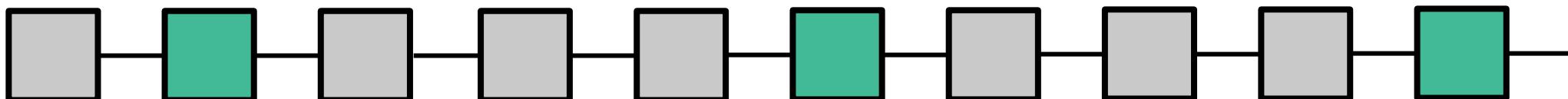
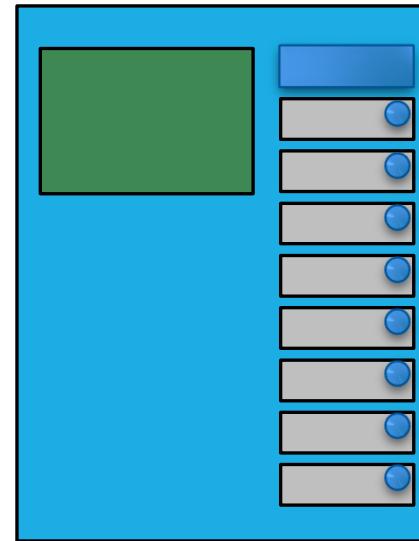
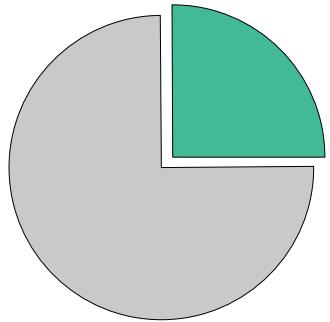
Exponentially distributed rate of new blocks, with  
constant mean interval

**target** automatically adjusted every 2016  
blocks so that mean interval is **10 minutes**

# INCENTIVES FOR MINING

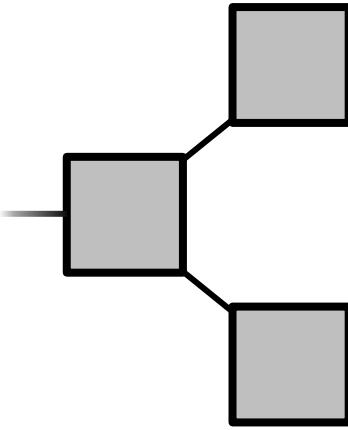
Prize:

- “**Minting**”
- **Transaction Fees**



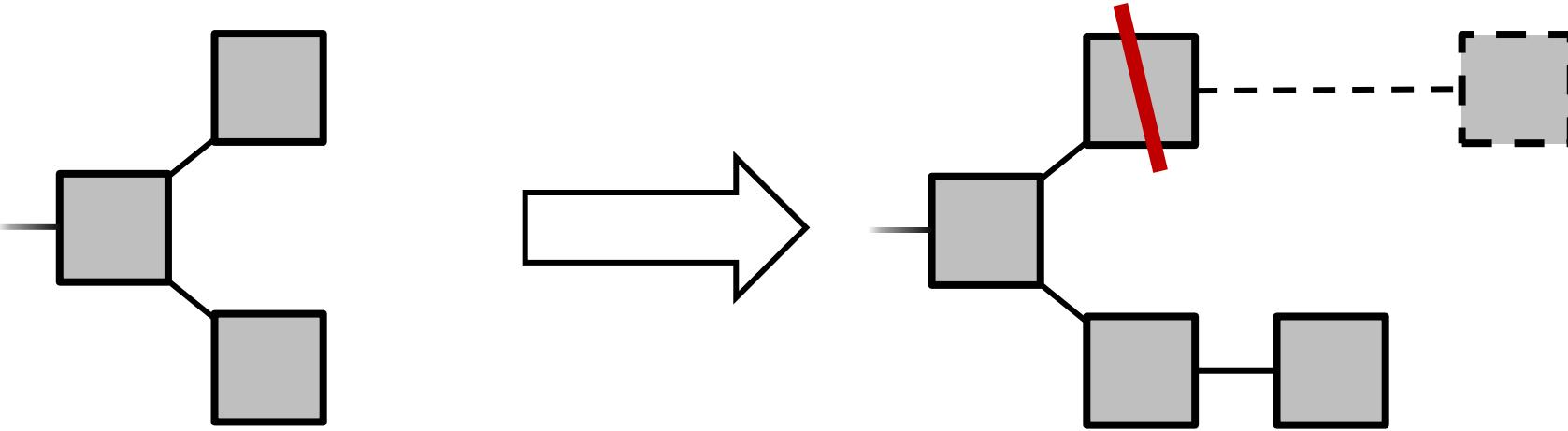
*Wins proportional to computation power*

# FORKS



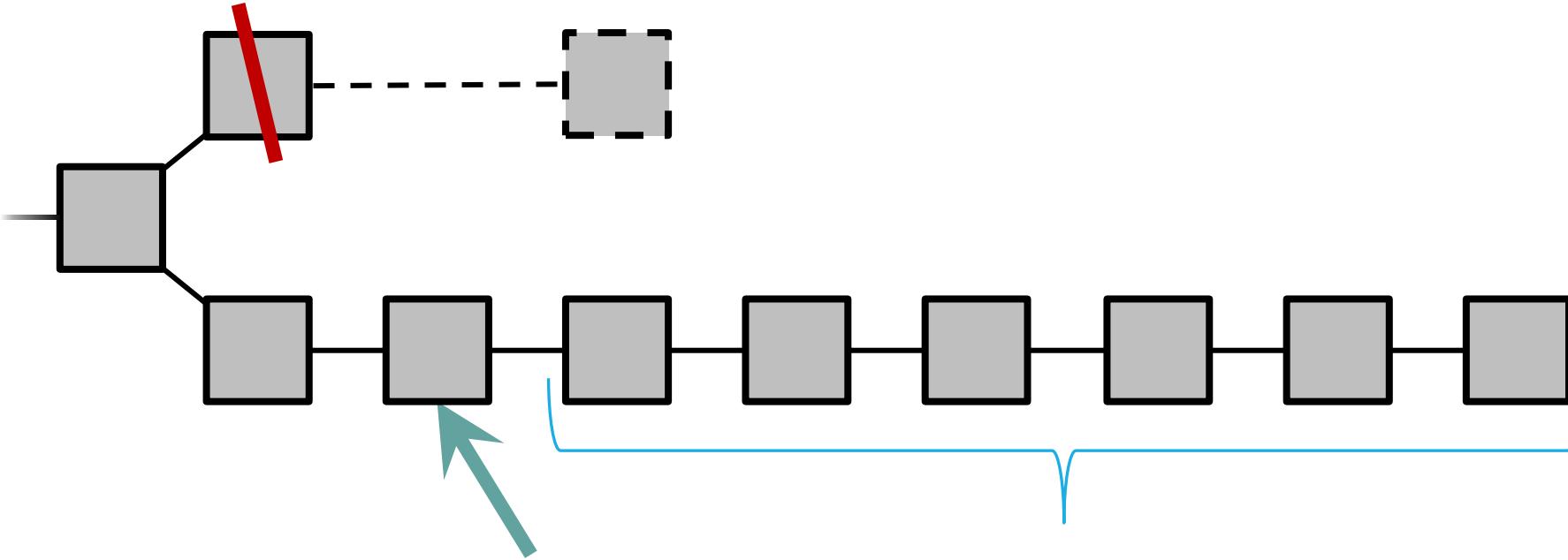
Two blocks “mined” at approximately the same time  
by two different miners

# FORK RESOLUTION



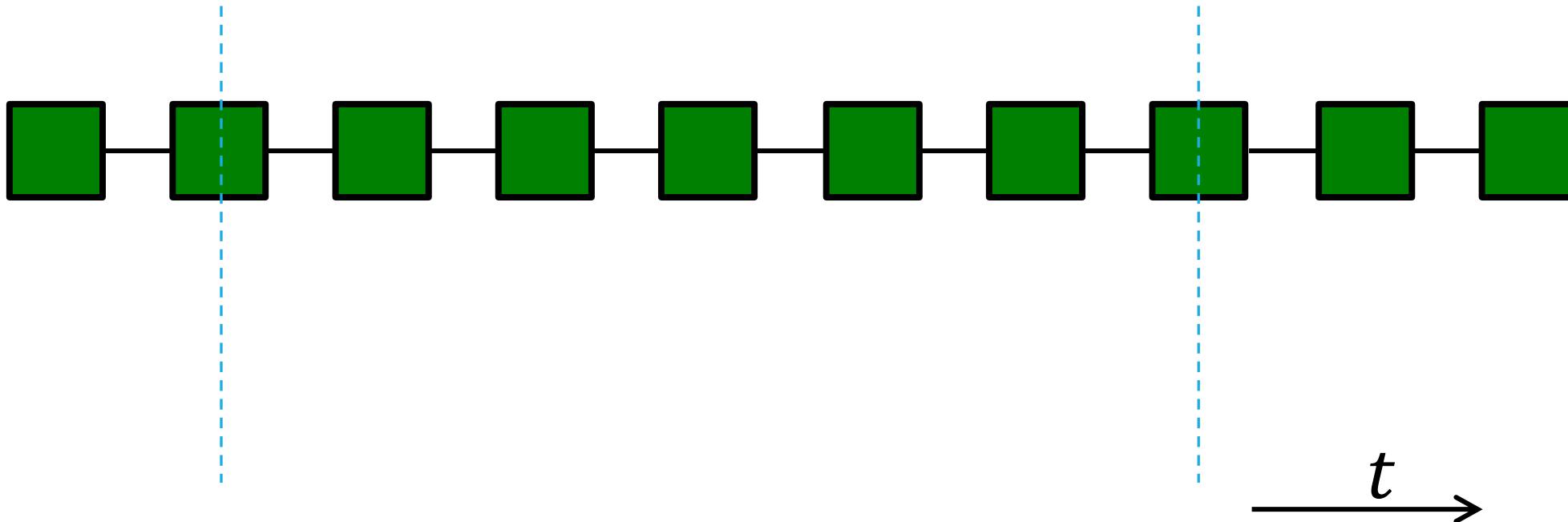
- **Longest chain wins**
- Transactions on short chain are reverted

# FORK RESOLUTION

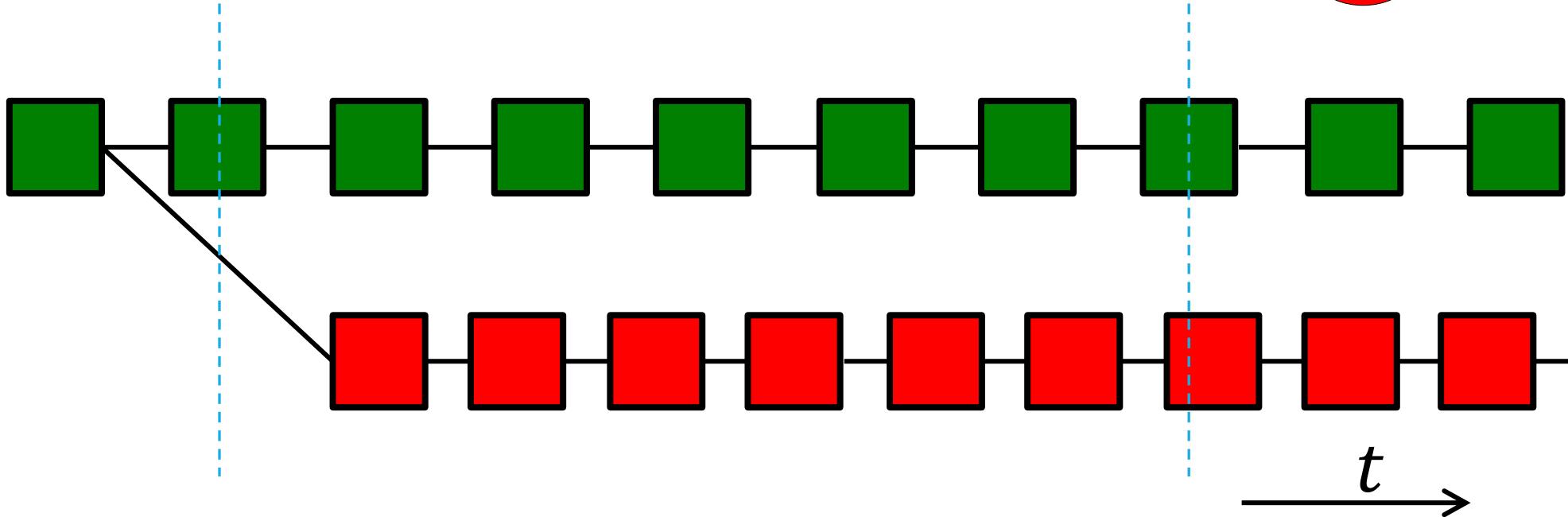
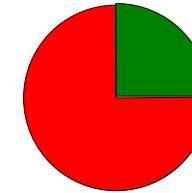


A transaction is **confirmed** when  
it is **buried** “deep enough”  
(typically 6 blocks – i.e., one hour)

# SECURITY THREAT!

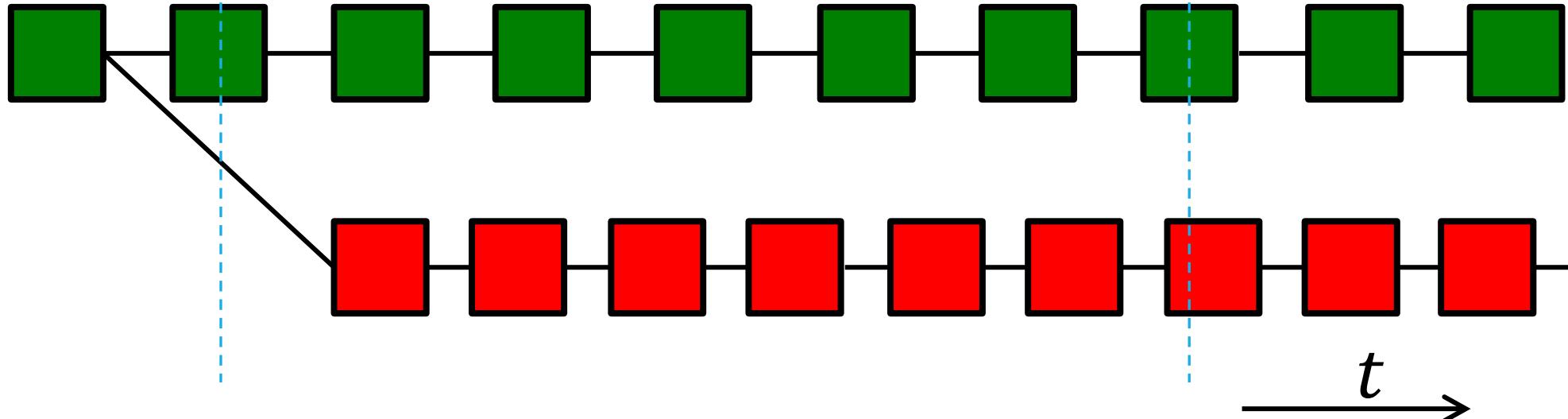


# SECURITY THREAT!

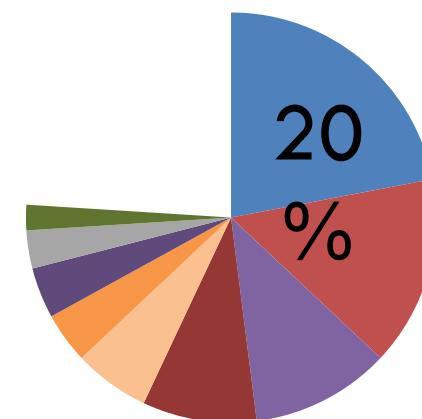


Threat: attacker outruns good miners

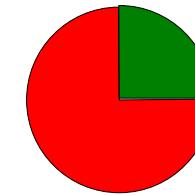
# SECURITY THREAT!



Threat: attacker outruns good miners  
→ **Security Assumption:** good miners  
own  $>.5$  of the total compute power



[blockchain.info,  
April 2015]



# **PERMISSIONLESS BLOCKCHAINS**

**Open membership, but inefficient**

**Vulnerable to 50% attacks**

**Examples include Bitcoin, Ethereum, IOTA**

# PERMISSIONED BLOCKCHAINS

## Performance:

- High Throughput, Low Latency
- Energy-efficient

## Security:

- No forks!

Closed membership

Examples include Ripple, Hyperledger

# BLOCKCHAIN FOR THE FARM?

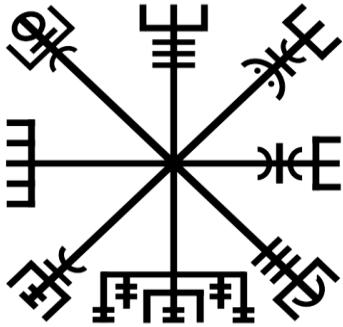
Blockchains require strong network connectivity and lots of storage

Permissionless blockchain are power-hungry

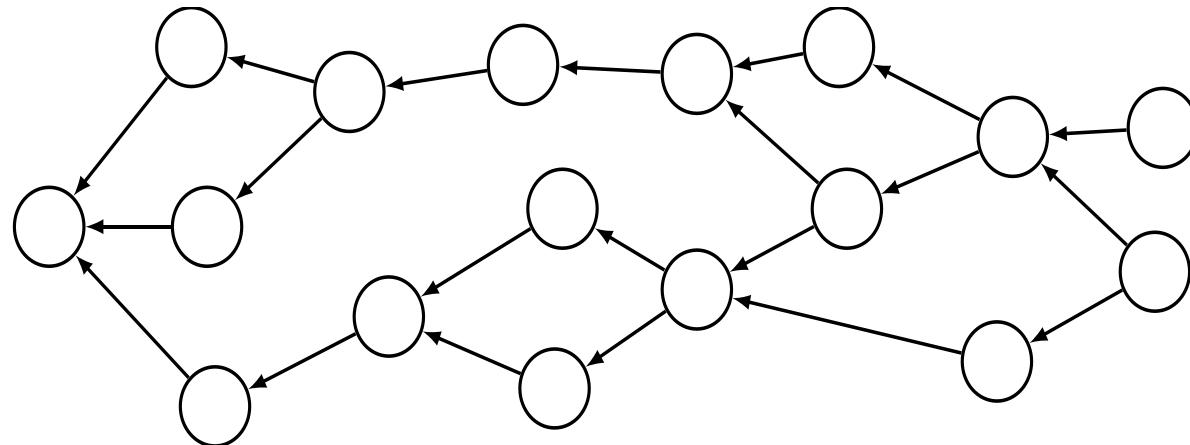
Sensors have limited resources

- Sensors for growing conditions, storage conditions, shipping conditions, ...

*Blockchain for a farm will generate records in a decentralized way, and hence it \*must\* work in a network-partitioned or -challenged environment*



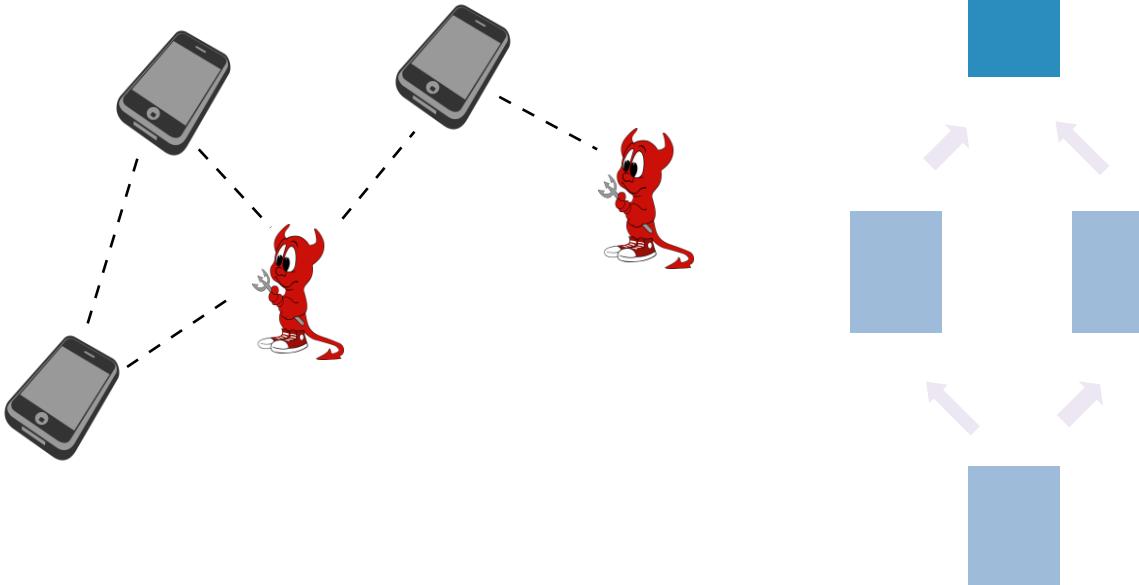
# Vegvisir: tolerate branches



- The key innovation is to allow branching as a feature.
  - Leads to DAG structure instead of linear blockchain
  - Still maintains full causal history of events (respect's Lamport's →)

# *Proof-of-Witness* to persist blocks securely

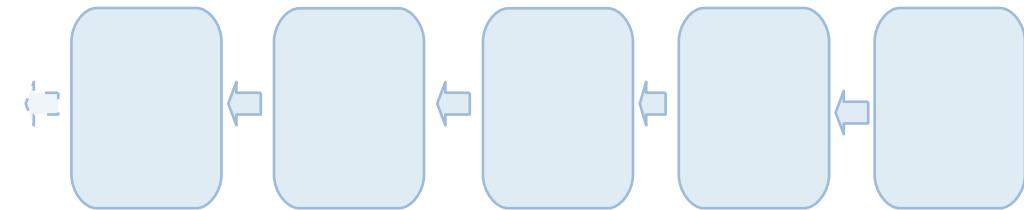
No more than  $k$  malicious nodes  
in any neighborhood



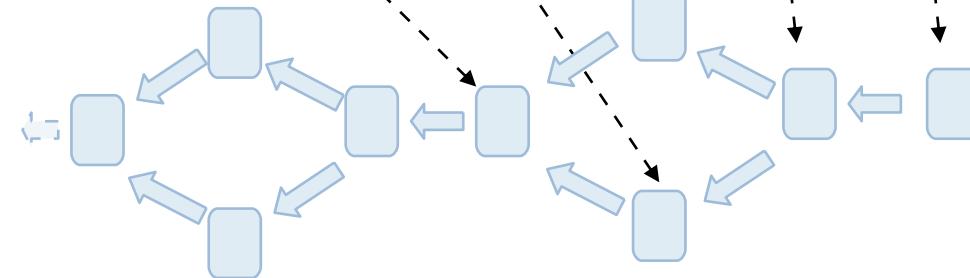
At least one copy of a valid block will survive if  $< k$  malicious peers

# *The Support Blockchain reduces sensor storage needs*

Support Blockchain

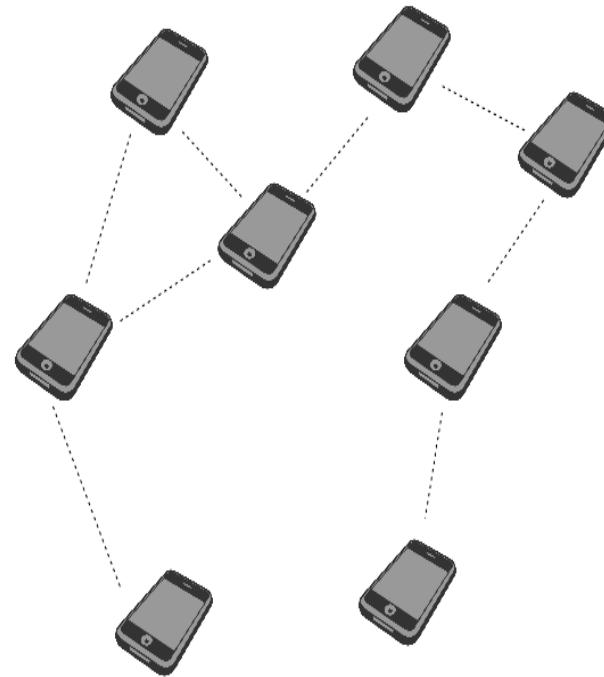


IoT Blockchain



Allows regular peers to discard old blocks when storage space is low

# Blocks are *gossiped* over ad hoc network

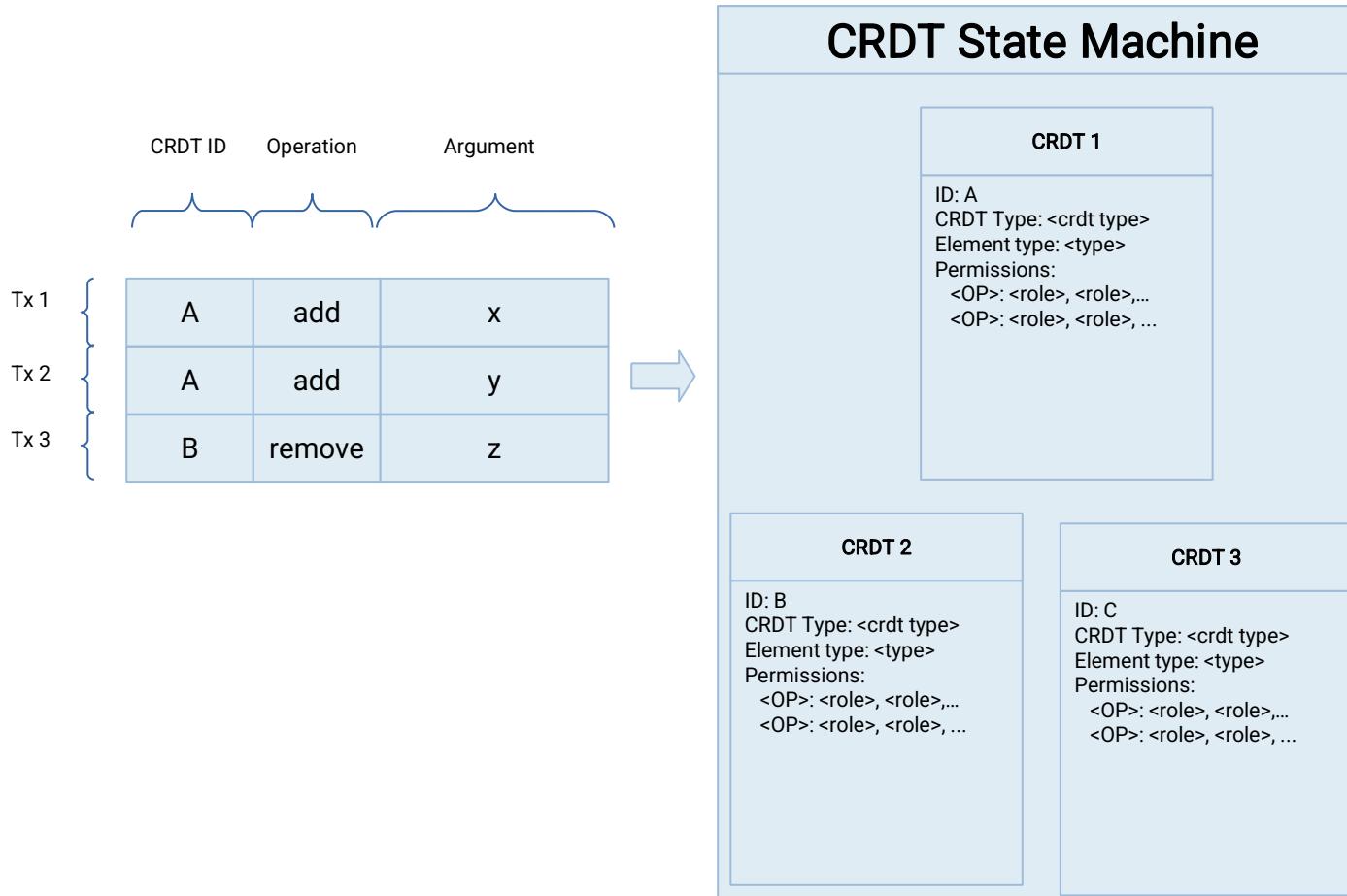


Heterogeneous, opportunistic networking

# *CRDTs for strong semantics in partitioned world*

- *Conflict-Free Replicated Datatype*
- Updates must be associative, commutative, idempotent
- Replicas can be updated independently and concurrently
- Basic CRDTs form registers, counters, sets

# Transactions manipulate CRDTs



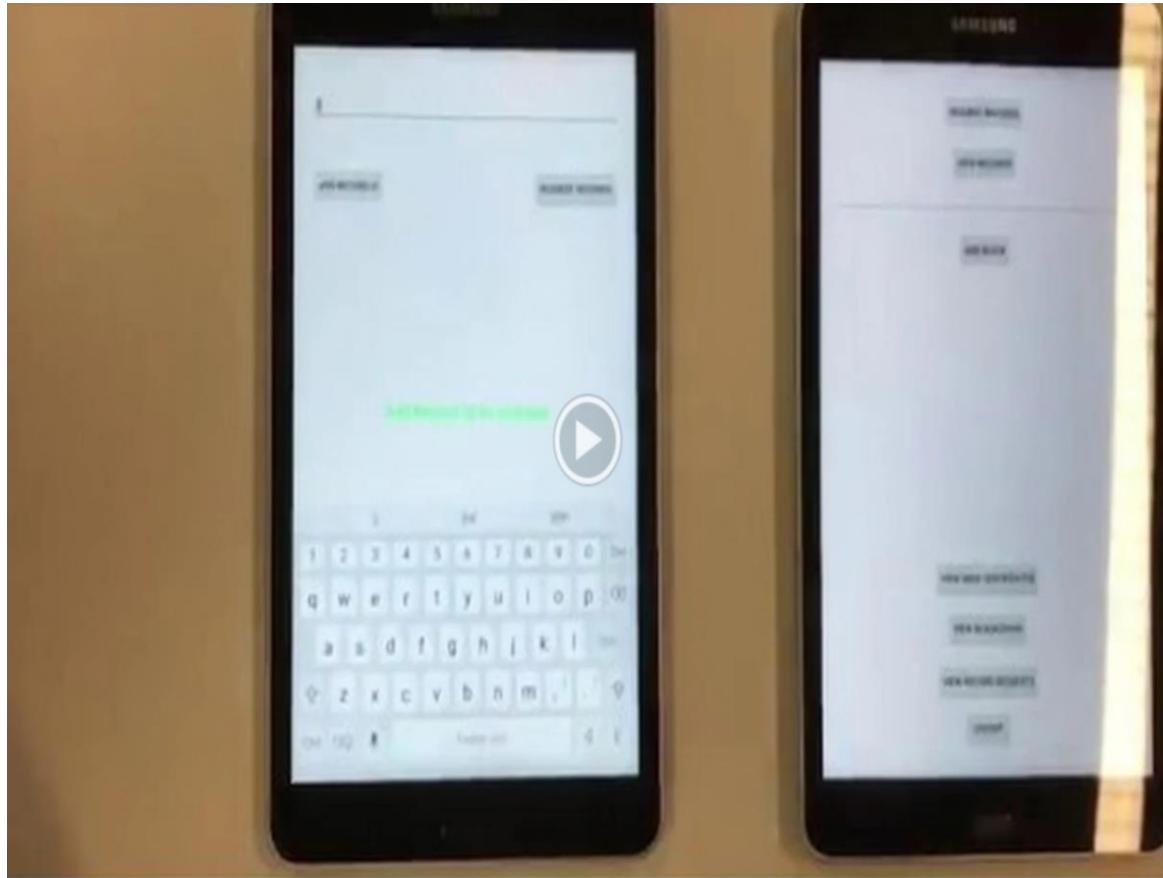
# BUT SOME QUESTIONS REMAIN OPEN

How would Vegsivir handle “double spending” (same coin spent in both branches), or other kinds of semantic conflicts that might not involve coins?

- The actual meaning of the operation changes, or it becomes invalid.
- This could cascade to impact subsequent operations, too.
- We can’t simply merge the chains and walk away...

Also, although smart farms have many sensors, Vegvisir lacks an answer to the issue of trust: we need the IoT hub to log enough information to know why we should trust a sensor, but this topic is out of scope for the paper.

# DEMONSTRATION VIDEO



Proof of concept system

# CONCLUSION

Exciting possibilities for blockchains in the food supply chain

But current blockchain designs may not be compatible with some deployment scenarios in the food supply chain

Vegvisir supports partitioned operation and has low power/networking/storage requirements