

Prepared by Dr. Wesam Ahmed

IT402
Internet of Things

Lecture 7



IoT Security and Privacy

Lecture 7

IT402

Internet of Things

Prepared by
Dr. Wesam Ahmed

07

Learning Objectives:

- 1.1 Describe the different IoT security challenges
- 1.2 Describe the different IoT security requirements
- 1.3 Describe the different IoT Domain Attacks

07

IoT Security Challenges

- **Multiple Technologies** :Each IoT technology (e.g. wireless sensor, RFID, cloud, virtualization) has its own **vulnerabilities** (insecure communication channels, data leakage, weak authentication).
- Since IoT systems depend on a chain of interconnected technologies, **the overall security is only as strong as the weakest link** in that chain.
- **Multiple Verticals**: The IoT paradigm will have numerous verticals such as healthcare, smart cities, industrial automation, agriculture, and transportation.
- Each vertical has **unique security requirements** (e.g., patient data privacy in healthcare vs. device reliability in industrial IoT).
- The **security requirements of each vertical** are quite different from the remaining verticals.

07

IoT Security Challenges

- **Scalability:** 30 billion smart devices will be connected to the Internet by 2022.
- Managing authentication, data protection, and secure communication for billions of devices is a **massive challenge**.
- **High Availability (HA):** HA refers to characteristic of a system that is continuously operational for a desirably long period of time.
- It is often measured against the **ideal of being “100% operational” or “never failing.”**
- **Security Impact:** Security and HA are closely linked. Network administrators sometimes **avoid using security tools** (like network discovery or active threat response functions) because they may **disrupt critical systems**.

07

IoT Security Challenges

- **Big Data:** Each smart object is expected to be supplied by numerous sensors, where each sensor generates huge streams of data over time.
- Protecting these **large data streams** requires **efficient, scalable, and real-time security mechanisms** that ensure **data integrity, confidentiality, and privacy**.
- Traditional data security tools often **struggle to keep up** with the **volume, velocity, and variety** of IoT-generated data.
- **Resource Limitations:** The majority of IoT end devices have limited resource capabilities such as CPU, memory, storage, battery and transmission range.
- These limitations make them **highly vulnerable to attacks**, especially **Denial of Service (DoS)**, where attackers can easily **overload the system** and cause **service disruptions**. **Flooding Attacks:** Sending massive amounts of data packets to exhaust device or network bandwidth. Designing **lightweight and energy-efficient** security solutions is essential.

07

IoT Security Challenges

- **Remote Locations:** In many IoT verticals (e.g. Smart Grid, Railways, Roadsides), IoT devices, especially sensors, will be installed in unmanned locations that are difficult to reach.
- Attackers can interfere with these **devices without being seen.**
- **To counter this, IoT systems must include:**
 - **Cyber and physical security monitoring**
 - Devices capable of **remote operation and maintenance**
 - **Durability** to operate under **extreme environmental conditions**
 - **Compact designs** suitable for limited space installations

07

IoT Security Challenges

- **Mobility:** Smart objects are expected to change their location often in the IoT paradigm.
- Examples: (wearable devices, connected vehicles).
- Constant movement creates **security and connectivity challenges**,
- such as maintaining **secure authentication, data integrity**, and **trust** across changing networks.
- Traditional static security measures are not sufficient; therefore, **defensive mechanisms must be adaptive, context-aware, and capable of real-time response** to protect devices in these **dynamic and mobile environments**.



07

IoT Security Challenges

- **Delay-Sensitive Services**
- Many IoT applications (e.g., **health monitoring, autonomous vehicles, industrial automation**) are **delay-sensitive**.
- Even minor delays caused by security processing or attacks can **disrupt operations** or **endanger safety**.
- Systems must be protected from any attack that **increases latency** or **causes service interruptions**.

07

IoT Security Requirements

To ensure secure and trustworthy IoT operations, several fundamental security requirements must be met:

- **Confidentiality:** ensures that the exchanged messages can be understood only by the intended entities.
- Achieved through **encryption** and **secure communication protocols** (e.g., TLS, SSL).
- **Integrity:** ensures that the exchanged messages were not altered/tampered by a third party.
- Protects against **tampering, modification, or injection attacks** by malicious entities.
- **Authentication:** ensures that the entities involved in any operation are who they claim to be.
 - Prevents **impersonation attacks**, where an attacker pretends to be another device or user.
 - Implemented using **passwords**.

07

IoT Security Requirements

To ensure secure and trustworthy IoT operations, several fundamental security requirements must be met:

- **Availability:** Ensures that IoT services and devices remain accessible and operational at all times.
 - Denial of Service (DoS) and Distributed DoS (DDoS) attacks directly threaten this requirement by causing service disruptions.
 - Achieved through redundancy, load balancing, failover mechanisms, and robust network management.
- **Authorization:** Ensures that entities (devices or users) have the appropriate permissions to perform specific actions.
 - Prevents unauthorized operations or access to restricted data.
 - Managed through access control lists (ACLs), role-based access control (RBAC), or policy-based authorization systems.

07

IoT Security Requirements

- **Freshness:** Ensures that **data is current and not reused** from an earlier session.
 - Protects against **replay attacks**, where an attacker resends old messages to trick a device or system into reverting to a previous state
 - Typically enforced using **timestamps**.
- **Non-Repudiation:** Ensures that an **entity cannot deny** having performed a specific action, such as sending a message or initiating a transaction.
 - Provides **accountability** in IoT communications, ensuring that every action or message can be verified, logged, and attributed to the responsible entity.

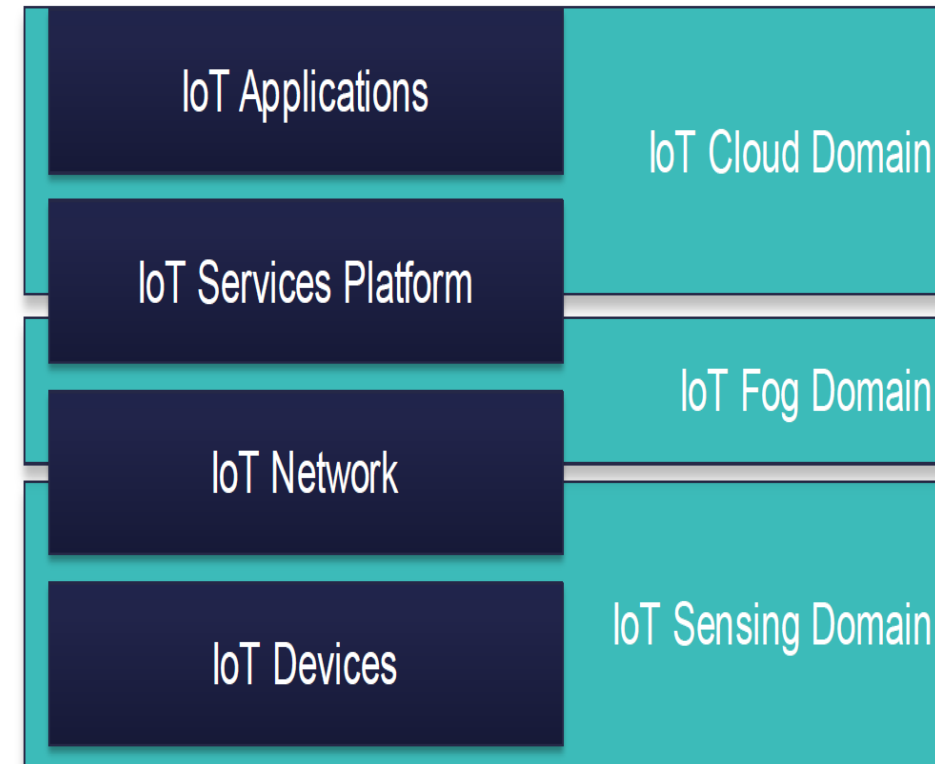


07

Mapping of IoT Domains

- IoT Devices (IoT Sensing Domain)

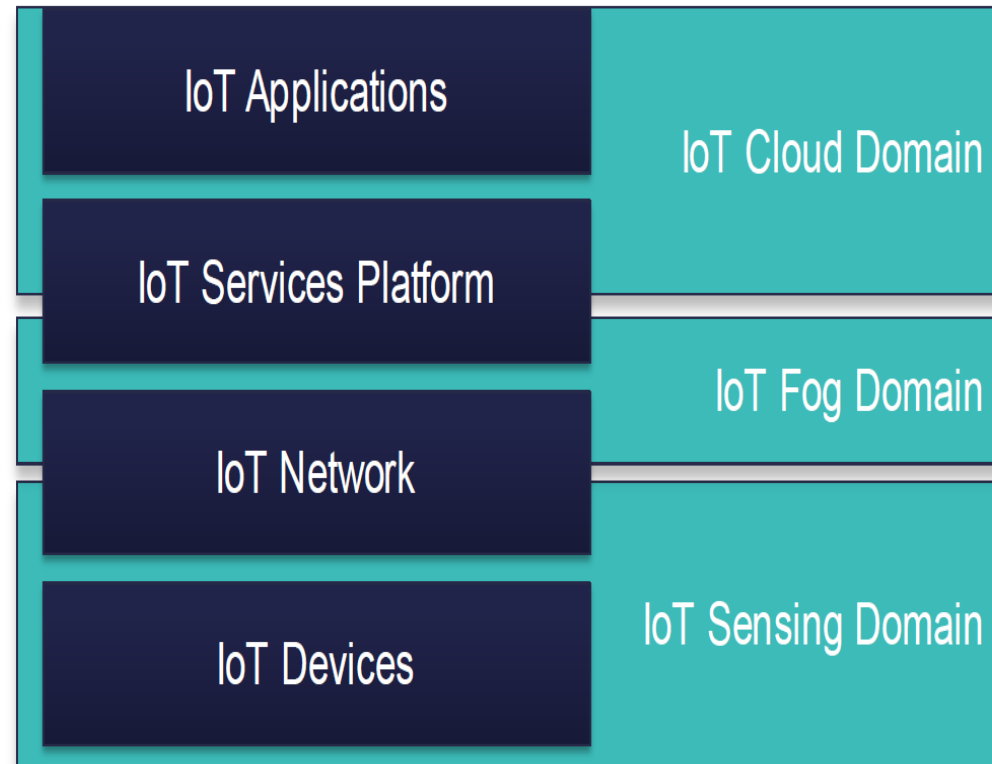
- This is the **foundation layer** of the IoT architecture.
- It includes **sensors, actuators, RFID tags, and embedded systems** that collect data from the environment.
- **Functions:**
 - Sense and gather physical data (e.g., temperature, motion, pressure).
 - Send this data to the network layer for processing.



07

Mapping of IoT Domains

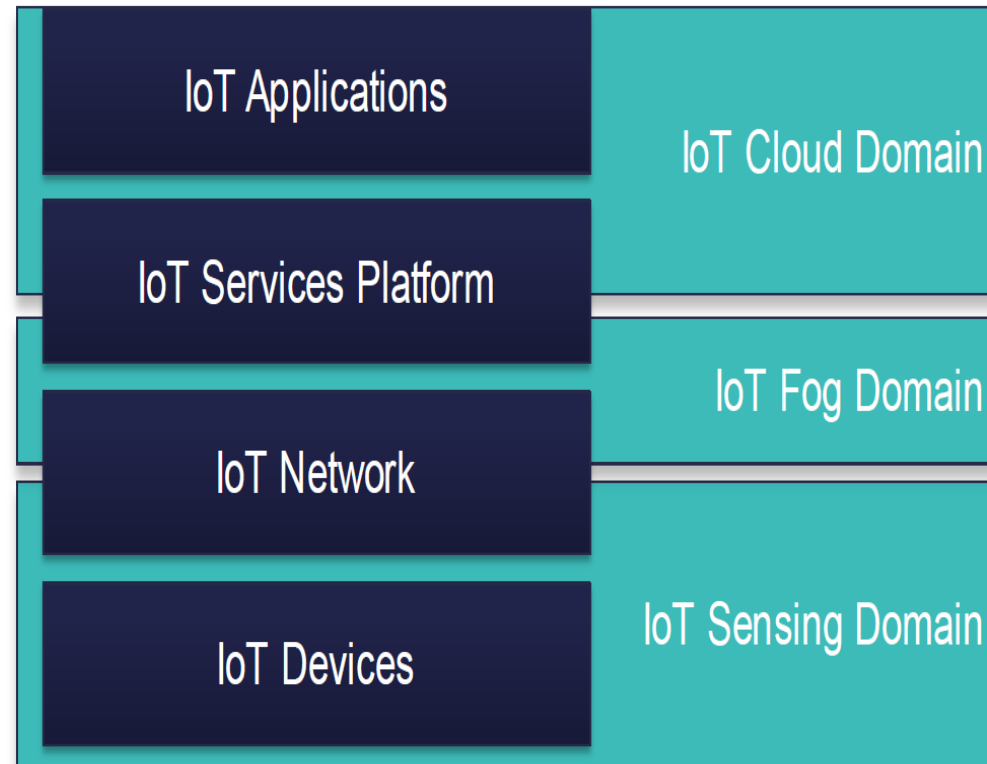
- **IoT Network (IoT Sensing Domain → Fog Domain)**
 - Responsible for **transmitting data** between IoT devices, fog nodes, and the cloud.
 - Uses multiple communication technologies such as **Wi-Fi, Bluetooth, ZigBee, 5G, and LPWAN**.
- **Functions:**
- Enable **data routing and connectivity**.
 - Ensure **secure, reliable, and low-latency** communication.



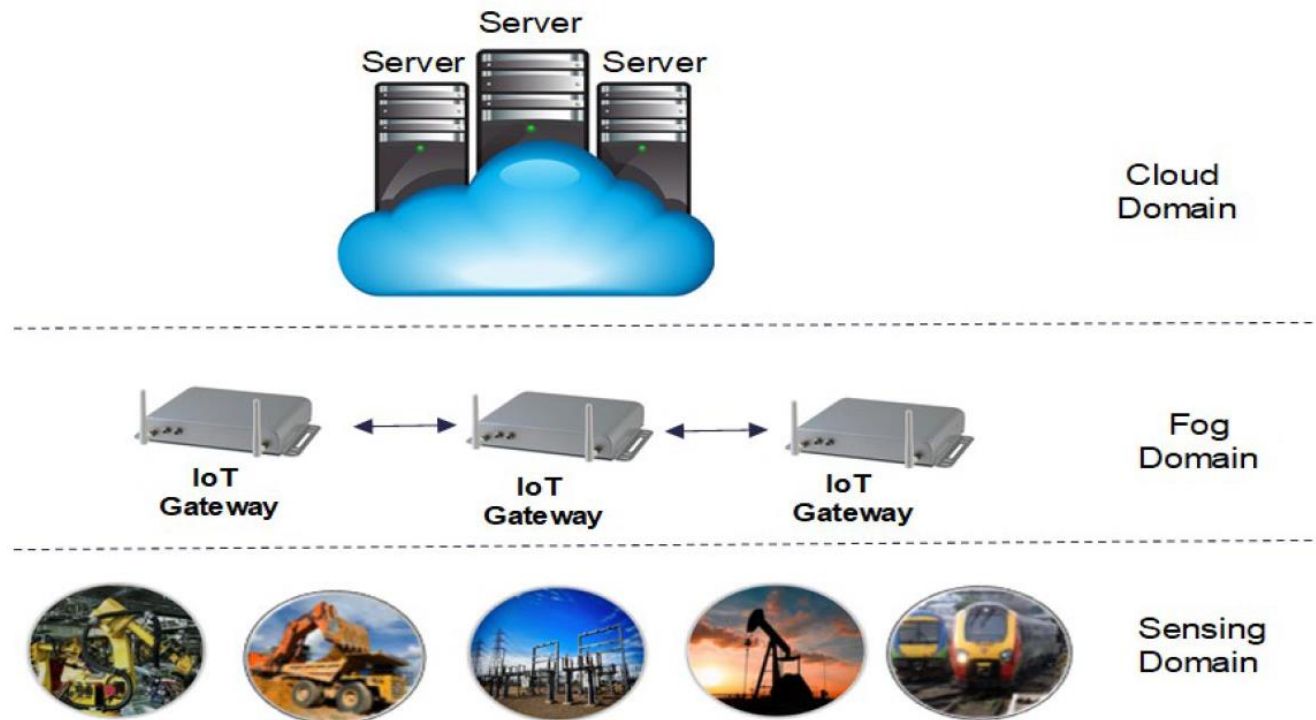
07

Mapping of IoT Domains

- **IoT Services Platform (IoT Fog Domain)**
 - Acts as an **intermediate processing layer** between IoT devices and the cloud.
 - Includes **edge or fog computing nodes** that perform **local data processing, filtering, and analytics** close to the source.
- **Functions:**
- Reduce latency and network congestion.
 - Improve responsiveness for time-sensitive applications.
 - Support **real-time decision-making** without full dependence on the cloud.



Mapping of IoT Domains



07

Mapping of IoT Domains

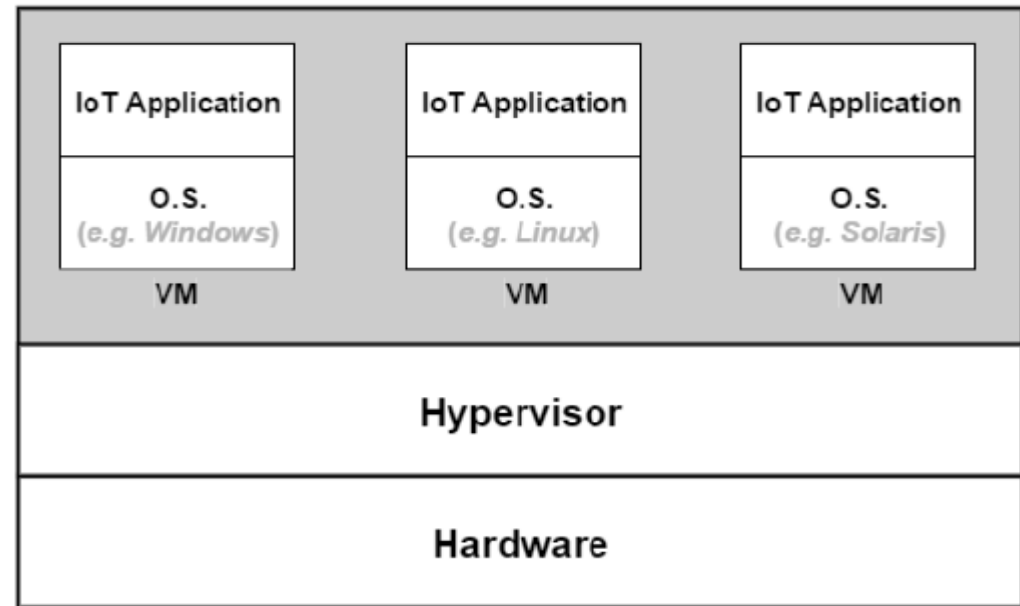
Summary of Domains

Domain	Components	Primary Role
IoT Sensing Domain	Devices & sensors	Data collection
IoT Fog Domain	Edge/fog nodes, local servers	Local data processing & control
IoT Cloud Domain	Cloud platforms & applications	Global data analytics & management

07

Cloud Domain Attacks

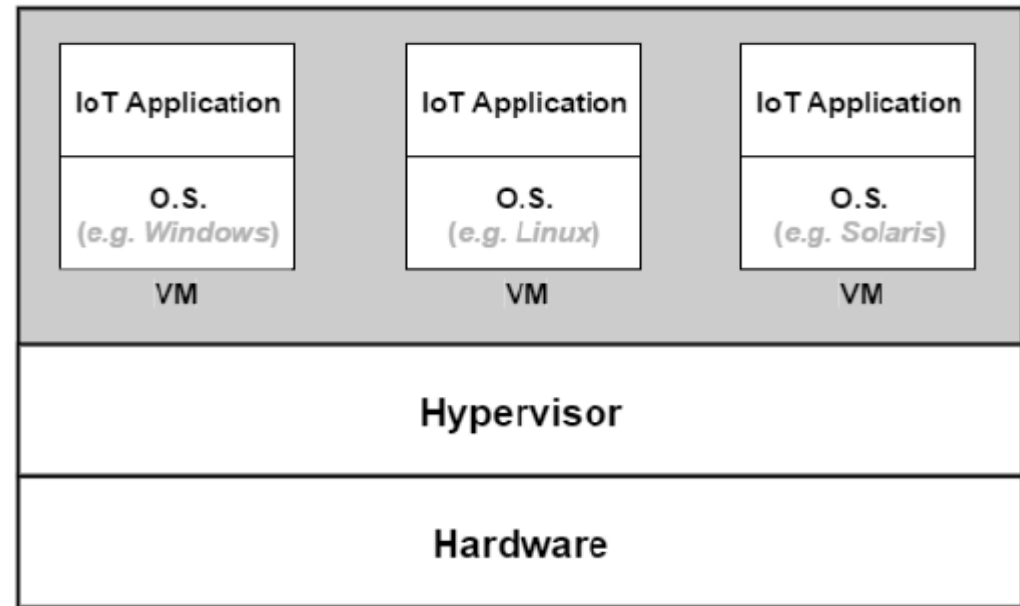
- **Cloud platforms** run many **IoT applications**, each performing **specific operations** (data processing, analytics, storage).
- Each **IoT application** is deployed on **one or more Virtual Machines (VMs)**.
- A **VM operates** like a separate computer, with its own **operating system (OS)**.



07

Cloud Domain Attacks

- Allocation of VMs to Physical Machines (PMs):
- VMs run on **Physical Machines (servers)**.
- Each **VM** is allocated specific resources:
 - **CPU**
 - **Memory**
 - **Storage**
- Attackers may attempt to overwhelm or exploit these resources.



07

Cloud Domain Attacks

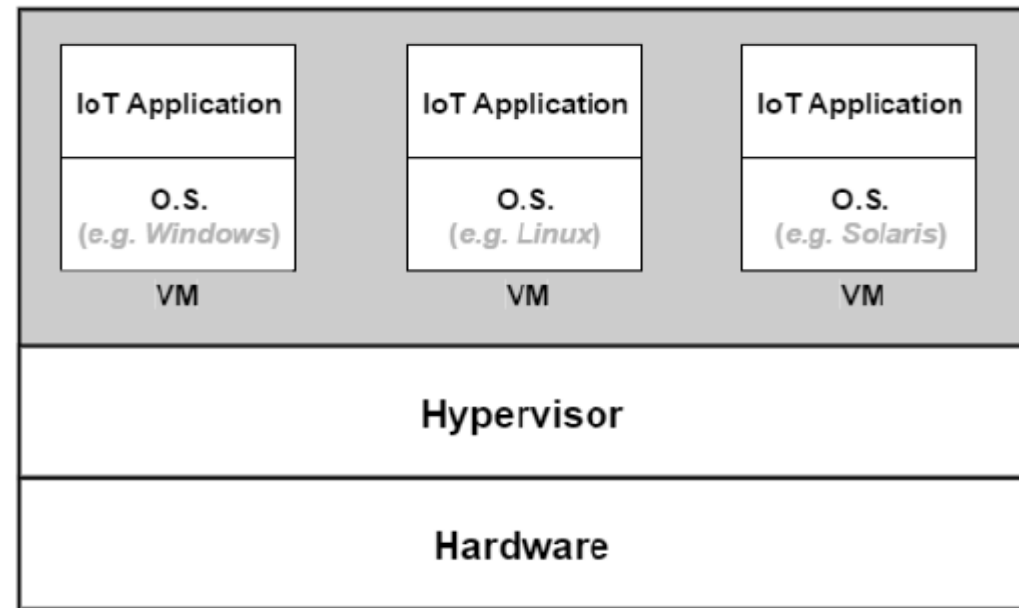
➤ Role of the Hypervisor (Virtual Machine Manager):

✓ **Monitoring Running VMs**

- Tracks VM performance and resource usage.
- Ensures the server is running efficiently.

✓ **Separating VMs from Hardware**

- VMs do not interact directly with physical hardware.
- Hypervisor acts as a layer in between.



07

Cloud Domain Attacks

Hidden Channel Attacks:

- Even though virtual machines (VMs) running on the same physical server are logically isolated by the hypervisor, they still share some physical hardware components, such as:
 - CPU cache
 - Memory buses
 - Network interfaces (virtualized but still connected to the same hardware)
- Because these components are physically shared, complete isolation is not always guaranteed.



Hidden Channel Attacks:

- Shared hardware introduces opportunities for **side-channel attacks**, where an attacker VM can observe patterns in shared hardware to infer sensitive information from another VM.
- Example: **Cache-based Side-Channel Attack**
 - Two VMs share the same CPU cache.
 - The attacker VM measures cache **access times to detect which memory areas were recently used by the victim VM.**
 - From these timing differences, the attacker can extract information such as:
 - **cryptographic keys , sensitive data operations**



Hidden Channel Attacks:

- To mitigate hidden-channel or side-channel attacks between VMs, several countermeasures can be applied. These techniques attempt to strengthen isolation and reduce the chances of information leakage through shared hardware:
- Hard Isolation Solutions:
 - ✓ Separate the cache dedicated for each VM through **hardware or software**: The idea is to **partition the CPU cache**, ensuring each VM gets **its own cache portion**. This prevents attackers from **observing cache usage patterns of other VMs**.
 - ✓ Another way to achieve hard isolation is by **assigning only one VM to each server**. This prevents data leakages across **VMs because no other VM shares hardware..**

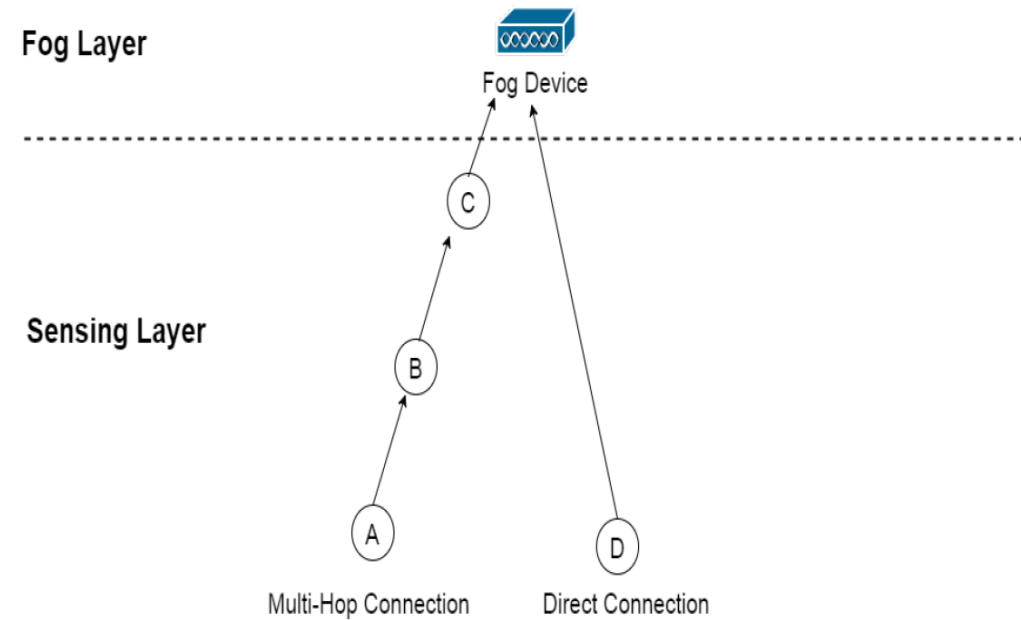


- **Noisy Data Access Time:** This technique aims to prevent attackers from using **timing side-channel attacks** to infer data access patterns of other VMs.
 - Adding Random Noise to Data Fetch Time
 - ✓ Adds **random noise to the amount of time needed to fetch data**, which makes it hard to tell whether or not the data was **fetches from the cache or from the memory**.
 - ✓ By doing this, it becomes harder for a **malicious VM to identify** what segments of the cache were populated by another VM that shares the same server.
 - ✓ The attacker VM cannot **accurately detect which cache lines were accessed** by the victim VM. This prevents the attacker from reconstructing: **Memory access patterns , Sensitive computations ,** What the victim VM is doing, What data or code it is accessing



The diagram shows **how sensor nodes communicate with a fog device** in a layered IoT architecture.

Sensing Domain



➤ Types of Connections

a. Multi-Hop Connection

- Data travels through **intermediate sensor nodes** before reaching the fog device.

In the diagram:

A → B → C → Fog Device

- Used when **nodes are far from the fog device or have limited transmission power.**

b. Direct Connection

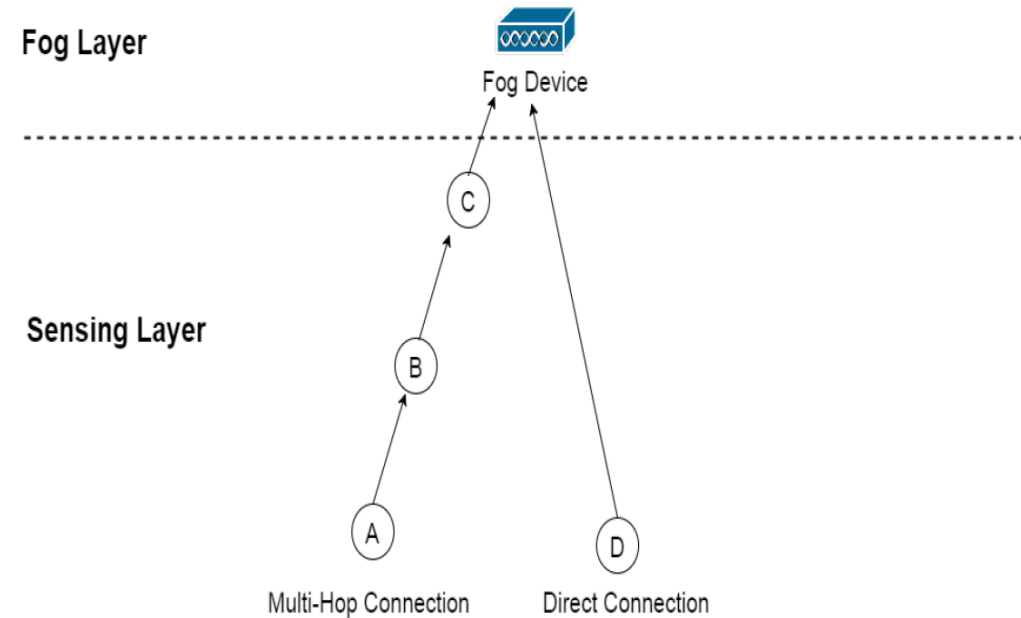
- Sensor nodes send data **directly to the fog device** without passing through other nodes.

In the diagram:

D → Fog Device

- **Requires sufficient communication range or power.**

Sensing Domain



07

Sensing Domain Attacks

➤ Jamming Attacks:

- ✓ A malicious attacker transmits **unwanted radio signals** on the same frequency used by legitimate devices.
This disrupts communication and **prevents devices from sending or receiving data properly**.
- ✓ Other Types of Jamming Attacks: Constant Jamming, Deceptive Jamming, Reactive Jamming, Random Jamming.



07

Sensing Domain Attacks

➤ Jamming Attacks Preventive Techniques:✓ **Machine Learning Techniques**

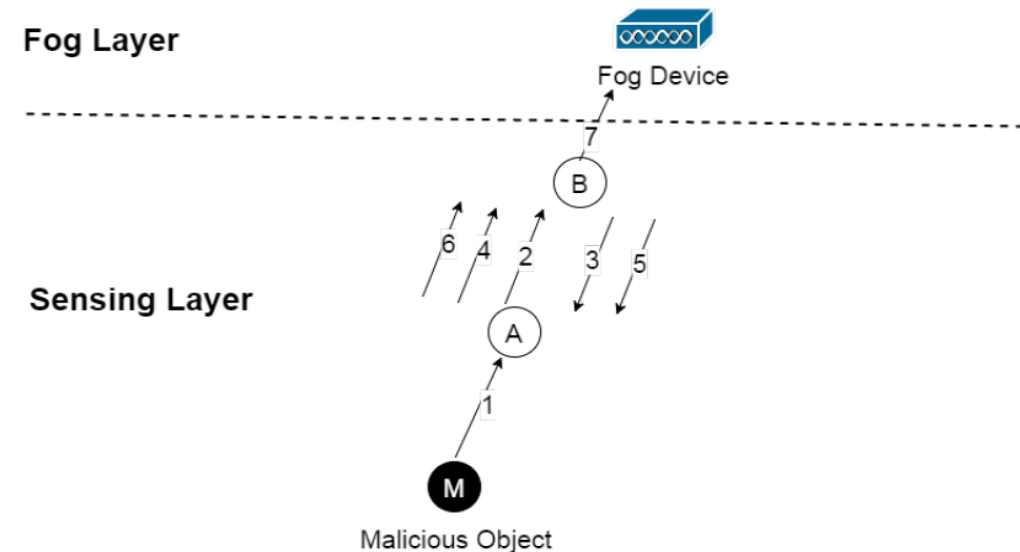
- Advanced **ML algorithms** (e.g., decision trees, SVMs, neural networks) can classify signals as:
 - **Legitimate communication**
 - **Jamming interference**
- ML can **detect subtle patterns** that are hard to identify using simple thresholds.

✓ **Signal Analysis**

- The collected data **is analyzed to determine if the patterns** match **normal network behavior** or a jamming scenario.

➤ **Carousel Attack:**

- ✓ Attacker specifies routing paths that include loops where same packet gets routed back and fourth among the other objects wasting their power (A sends to B and B sends to A).
- It is designed to **waste the energy** of network nodes by creating unnecessary routing loops.
- **How the Carousel Attack Works**
- The attacker **creates a fake routing path** that includes **loops**. Attacker sends a malicious packet with the route field
- A packet is intentionally routed in a cycle — for example:
A → B → A → B → A → ...
- Instead of reaching its destination, the packet keeps traveling back and forth.

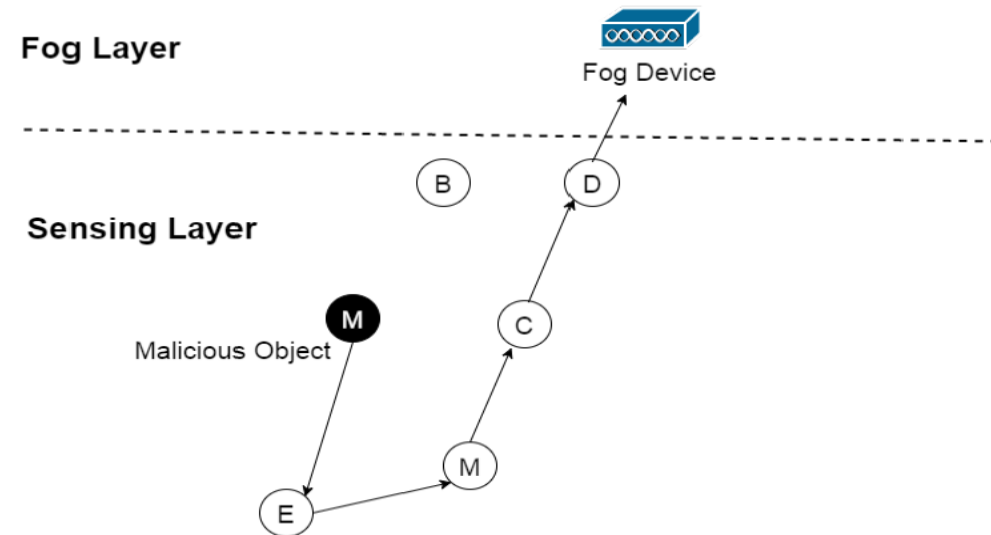


07

Sensing Domain Attacks

➤ Stretch Attack

- ✓ If the routing protocol supports source routing, then an attacker can send the packets that it is supposed to report to the fog device through very long paths rather than the direct and short ones (M selects longest path)



07

Sensing Domain Attacks

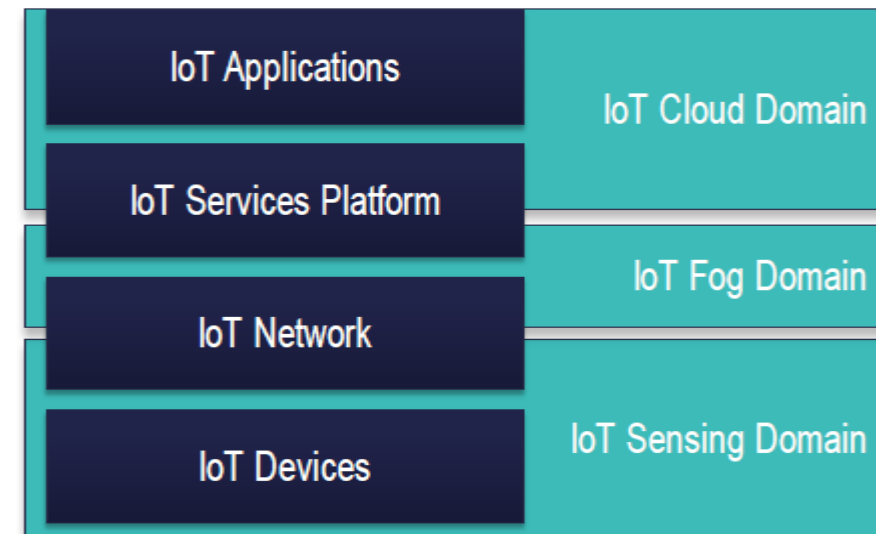
➤ How the Stretch Attack Works

- ✓ If the routing protocol allows the sender to specify the exact path (source routing), then:
- ✓ A malicious node **M** creates a **fake, very long route** for the packets.
- ✓ Instead of going through the **shortest path** to the fog device or base station, the packet is sent through many extra nodes. The attacker intentionally inserts many intermediate nodes to make the path as long as possible
- ✓ **Example:**
- ✓ Normal shortest path:
A → Fog Device
- ✓ Malicious stretched path chosen by attacker M:
A → X → Y → Z → W → ... → Fog Device

07

Fog Domain Devices and Challenges

- Fog Layer in IoT
- The **Fog Layer** sits between the **edge (sensors/devices)** and the **cloud** in an IoT architecture.
- It is designed to **process data closer to where it is generated**, reducing latency and network load.



07

Fog Domain Devices and Challenges

➤ Fog Layer in IoT:

✓ Functions of Fog Layer Devices

- ✓ Fog devices are intermediate computing nodes that handle **local processing** before sending data to the cloud.

✓ Data Collection

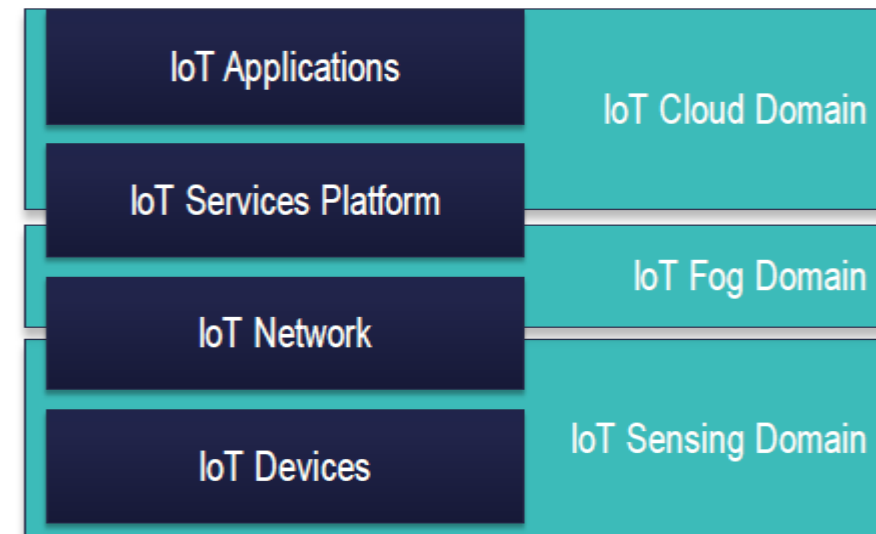
- Collect sensing data from sensors or smart objects.

✓ Local Processing

- **Data aggregation:** Combine data from multiple sources to reduce redundancy.
- **Data preprocessing:** Clean, filter, or normalize data.
- **Data storage:** Temporarily store data for analysis or caching.

✓ Data Forwarding

- ✓ Send processed or summarized results to the **cloud layer** for further storage, analytics, or decision-making.



07

Fog Domain Devices and Challenges

Fog Layer is similar to cloud layer with key differences:

Feature	Fog Layer	Cloud Layer
Location	Close to users or IoT devices; high-popularity areas	Centralized data centers, may be far from devices
Mobility	Supports dynamic locations of devices; mobile fog nodes possible	Usually fixed infrastructure
Computing Capacity	Limited; lightweight processing and storage	High; heavy processing and large-scale storage

07

Fog Domain Attacks and Countermeasures

- Fog Domain Attacks and Countermeasures:
- Authentication and Trust Issues:
 - Unlike cloud data centers which are offered by well-known companies, fog devices will be owned by multiple and less-known entities.
- Countermeasures:
 - **Mutual authentication protocols:** Ensure both fog device and smart object verify each other's identity.

07

Fog Domain Attacks and Countermeasures

- Fog Domain Attacks and Countermeasures:
- **Higher vulnerability to DoS Attacks:**
 - Fog devices have **lower computing capacity** compared to cloud servers.
 - Attackers can easily **overload them with traffic** (DoS attacks), disrupting service for connected smart objects.
- **Countermeasures**
 - **Traffic monitoring and rate limiting** to prevent resource exhaustion.
 - **Lightweight intrusion detection systems (IDS)** tailored for fog environments.

07

Fog Domain Attacks and Countermeasures

➤ Fog Domain Attacks and Countermeasures:➤ **Container-Related Security Threats**

- Many fog devices run multiple **containers** on the same OS and hardware.
- In computing, a **container** is a **lightweight, standalone, and executable software package** that includes everything needed to run a piece of software: the code, runtime, system tools, libraries, and settings.

➤ **Compromise of one container** can affect other containers sharing the same environment.➤ **Countermeasures**

- **Strong container isolation using sandboxing** (Each container runs in a “sandbox,” a protected environment that **limits what it can access** outside itself.).

07

Fog Domain Attacks and Countermeasures

➤ Fog Domain Attacks and Countermeasures:

Summary

Attack Type	Risk	Countermeasure
Authentication & Trust	Malicious fog owners	Mutual authentication, trust scores
VM Migration	Exposure during transfer	Encrypted migration, secure protocols
DoS Attacks	Resource exhaustion	IDS, rate limiting, redundancy
Container Security	Cross-container compromise	Container isolation, hardened images

07

Summary

1. IoT security
2. Domain Attacks



Thank You