Chapter-4

Transmission Control Protocol/Internet Protocol (TCP/IP)

* Based on packet switching.
* Connection-oriented protocol which sends the data in packets.
* TCP can retransmit the data packets if the packets have been lost.
* TCP is a reliable end-to-end protocol -
  - detects duplicate message & discards them
  - reduces data transfer rate.
* TCP/IP is different from OSI layer as it is 5-layered only. application, transport, network, data-link, physical.
* Its protocols are TCP & UDP; defines level of service & connection status while transmission of data.
* The Internet layer of TCP/IP protocol suite divides the messages into packets known as datagrams. Internet layer is responsible for routing of packets.
* Main protocols of internet layer are IP, ICMP, Address Resolution Protocol (ARP), RARP, IGMP.
* Routers has components such as CPU, RAM, flash memory, ROM & interfaces

IoT Reference framework

* consists of four levels - Device level, network level, Application level, application service Platform level.

IoT Network Layer (Addressing Protocol)

* IP is responsible for packet routing

IPV4 Protocol

* IPV4 addresses represented in dotted-decimal format.
* 32-bit addressing scheme ⟶ 2 parts - ① network address ② host address.
* A subnet mask extracts an IP address as a host or a network
* This helps TCP/IP protocol to identify whether the host is in local subnet or on a remote network
* NAT extends IPV4 usability by allowing private networks to reuse IP addresses
* lacks in built-in security mechanisms.
* Classes A, B, C identify actual networks, multicasting is done using class D, class E is reserved for class for experiment purpose

(192.168.1.1)

IPV6 & its role in IoT:

* 128-bit addressing scheme. hexadecimal [83DA:B3:0:3F3B:3AA1:FF:FE28:9C5B]
* Improved security - built-in support for IPsec encryption & authentication
* mobility support, Scalability, Connectability
* It is crucial for IoT bcoz it enables direct device-to-device communication without requiring NAT.

classification of IPV6 addresses:

3 types of addresses -

① Unicast:
* Used for one-to-one communication b/w devices
* Each device has a unique address

② multicast:
* Used for delivering packet to multiple interfaces
* It identifies a group of address interfaces belong to different nodes

③ Anycast:
* Data is sent to nearest available node.
* optimizes load balancing.

IoT Data Link Protocol:

Ethernet & IEEE 802-3:

* Ethernet refers to CSMA/CD LANs that conform to Ethernet specifications including 802.3 (IEEE).
* CSMA/CD nodes listen to the network to find if it is in use or free before sending data. If free, trasmit, else wait for some time, otherwise collision occurs.

ContikiMAC:

* It is a protocol that proposes enhancement over X-MAC protocol
* Based on asynchronous mechanisms, radio duty cycling protocol that uses periodical wake-ups to listen for packet transmissions
* Channel check rate is important.
* Its packets are ordinary link-layer messages.
* It uses fat sleep optimization approach and transmission phase-lock optimization.

IEEE 802.15.4:

- commonly used as IoT standard for MAC layer.
- low power multi-hop networking protocols are needed for IoT.
- provides high reliability & low cost solutions.
- slot frame structure, node activity scheduling, synchronization of nodes, frequency channel hopping, network formation.

IEEE 802.11 AH (WiFi):

- Drawbacks of original wifi - frame overhead, substantial power consumption.
- It is for less overhead requirements that are suitable for IoT.
- used for all digital device applications such as digital TVs, mobiles, —
- Main features are: Frame synchronization, Bidirectional packet exchange, short length MAC frame, preamble.

IEEE 802.16:

- Latest series of wireless broadband standards from IEEE.
- recognized as WiMax (Worldwide Interoperability for Microwave Access).

Application Layer IoT Protocols:

HTTP:

- It is for hypermedia information both for distributed as well as collaborative systems.
- HTTPS was supported by SSL (Secure Sockets Layer) protocol (initially), now TLS (Transport Layer Security).
- current version of HTTPS is in RFC 2818.

FTP:

- suitable for web applications.

Messaging protocols:

- To reduce chunks of data in a message.
- CoAP: Constrained Application Protocol.
  - synchronous request/response type.
  - low power small devices uses this.
  - supports computation & communication capabilities for RESTful communication.
  - Runs over connectionless UDP.
  - Has two sublayers - messaging & request/response → for communication
  - four messaging modes: confirmable, non-confirmable, piggyback & separate.

* MQTT: Message Queue Telemetry Transport
  * Developed by IBM, is a lightweight messaging protocol designed for M2M communication & IoT applications.
  * Uses publish/subscribe model over the TCP protocol stack.
  * A central broker manages communication blw publishers & subscribers.
  * Publishers send data to the broker, subscribers receive messages whenever there's an update to the topics they subscribed to.
  * Applications: remote telemetry, communication in low-bandwidth, high-latency environments.
  * QoS Levels:
    Level 0: At most once (best effort, no acknowledgement)
    Level 1: At least once (acknowledged & retransmitted if needed)
    Level 2: Exactly once (ensures message delivery with four-step confirmation)

* SMQTT: (Secure MQTT)
  * It integrates lightweight attribute-based encryption for better security.
  * After encrypting one message it can be delivered to multiple nodes
  * Works in four stages: setup, encryption, publish & decryption.
  * The broker handles encrypted messages, forwarding them to subscribers.

* XMPP: Extensible Messaging & Presence Protocol.
  - designed for messaging & chatting applications.
  - Supports both publish/subscribe & request/response architecture
  - QoS is not ensured, not suitable for M2M communication.

* AMQP: Advanced Message Queuing Protocol.
  * designed & developed for applications of financial industry
  - Uses TCP to transport messages. provides asynchronous publish/subscribe communications with messaging.
  - Reliable communication achieved by message delivery guarantee - at-most once, at-least once, exactly once delivery.

* DDS:- Data Distribution Service.
  * publish/subscribe protocol. 23 types of quality service levels offered based on criteria such as priority, security, urgency, reliability & durability.
  - Two sublayers - data-centric publish-subscribe & data-local reconstruction.
  - Guarenteed delivery of message since broker-less architecture.

Low-Power Wide Area Network (LPWAN)

* For low baud rate & long-range communications blw connected objects
* Battery powered things.
* NarrowBand - IOT : NBIOT
  + standards-based LPWAN technology.
  - Attains low power consumption of devices, enchances system capacity, spectrum efficiency
  - It is simpler, cost decreases, demand increases
  - goals are to reduce cost, increase battery life, better indoor coverage & provide high density connections.
* LoRa - LoRa WAN protocol
  - low power, low-bitrate, long-range & wireless communication system.
  - long battery & low powered devices are needed. Low energy consumption.
  - It is a specification based on CSS (Chirp spread spectrum) for physical layer
  - High robustness, multipath resistance & Doppler resistance
  - Benifits - * large range relatively 10 km.
    * different spreading factor orthogonal to each other.
    * when multiple transmissions occur at same time with same parameters then the high probable transmission is considered as strongest.

Chapter-6 : IoT System Management with NETCONF-YANG

Need for IoT system management:-

* IoT systems can have complex software, hardware and deployment designs including sensors, actuators, software & network resources, data collection & analysis services & user interfaces.
* The need for managing IoT systems is described as follows;

(i) Automating Configuration :-

* IoT system management capabilities can help in automating the system configurations
* System management interfaces provide predictable & easy to use management capability & ability to automate system configuration.
* Automating the system configuration ensures all devices have same configuration and variations or errors due to manual configurations are avoided.

(ii) Monitoring Operational & Statistical Data :-

* Operational data is the data which is related to the system's operating parameters & is collected by system at runtime.
* Statistical data is the data which describes the system performance.
* Management systems can help in monitoring operational & statistical data of a system. This data can be used for fault diagnosis or prognosis.

(iii) Improved Reliability :-

* A management system that allows validating the system configurations before they are put into effect can help in improving the system reliability.

(iv) System Wide Configuration :-

* For IoT systems that consist of multiple devices or nodes, ensuring system-wide configuration can be critical for the correct functioning of system.
* Management approaches in which each device is configured separately can result in system faults or undesirable outcome.
* System wide configuration is required where all devices are configured in a single atomic transcation.

(v) Multiple system Configurations:-

* for some systems it may be desirable to have multiple valid configurations which are applied at different times.

(vi) Retrieving & Reusing Configuration :-

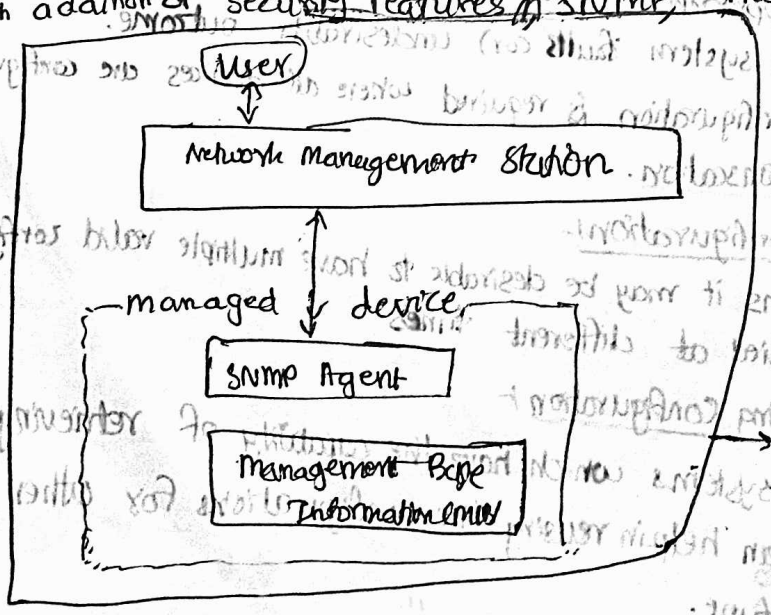
* Management systems which have the capability of retrieving configurations from devices can help in reusing the configurations for other devices of the same type.

SNMP (Simple Network Management Protocol)

* It allows monitoring & configuring network devices such as routers, switches, servers, printers, etc.
* Network Management Station (NMS) executes SNMP commands to monitor and configure the managed device.
* Managed device has MIB that has all information of device attributes to be managed.
* MIBs use Structure of Management Information (SMI) notation for defining the structure of management data.
* The structure of management data is defined in the form of variables which are identified by Object Identifiers (OIDs).
* SNMP is application layer protocol that uses User Datagram Protocol (UDP) as the transport protocol.

Limitations of SNMP:

* Stateless in nature & each SNMP request contains all the information to process the request.
* For a sequence of SNMP interactions, the application needs to maintain state and also smart enough to roll back the device into a consistent state in case of errors.
* SNMP is a connectionless protocol that uses UDP, making it unreliable.
* MIBs lack writable objects without which device configuration is not possible using SNMP. So without writable objects, SNMP used only for device monitoring & status polling.
* Retrieving the current configuration from a device can be difficult with SNMP.
* With addition of security features in SNMP, increased complexity a lot.



Network Operator Requirements

- Ease of use : Key requirement for any network management technology.
- Distinction blw configuration and state data : configuration data is the set of writable data that is required to transform the system from initial to current state. State data is non-configurable & includes operational & statistical data so, important to make a clear difference.
- Fetch configuration and state data separately : To make above thing possible it should do this thing. This is useful when configuration & state data from different devices need to be compared.
- Configuration of the network as a whole : This is important for systems which have multiple devices & configuring them within one network wide transaction is required to ensure the correct operation of the system
- Configuration transactions across devices : should be supported.
- Configuration deltas : support with minimal state changes
- Dump & restore configurations
- Configuration validation.
- Configuration database schemas.
- Comparing configurations : Devices should not arbitrarily reorder data, so that it is possible to use text processing tools such as diff to compare configurations.
- Role based access control : so that user is given minimum access necessary to perform a required task
- Consistency of access control lists
- Multiple configuration sets
- Support for both data-oriented & task-oriented access control.

---

* For managing a network device the client establishes a NETCONF session with the server.
* NETCONF allows the management client to discover the capabilities of the server.
* NETCONF gives access to native capabilities of the device.
* NETCONF defines one or more configuration datastores that contains all the configuration information to bring the device from its initial state to operational state
* NETCONF is a connection oriented protocol & NETCONF connection persists blw protocol operations
* For authentication, data integrity & confidentiality NETCONF depends on the transport protocol. eg. SSH or TLS.
* NETCONF overcomes the limitations of SNMP, suitable for monitoring state information & configuration management.

YANG :-

* data modeling language used to model configuration and state data manipulated by NETCONF protocols
* YANG modules define the data exchanged between NETCONF client & server.
* A module comprises of a number of 'leaf' nodes which are organized into a hierarchical tree structure.
* Leaf nodes are specified using 'leaf' constructs and organised using 'container' constructs.

operations:
connect, get, get-config, edit-config, copy-config, delete-config, lock, unlock, get-schema, commit, close-session, kill-session.

Node Types:
Leaf nodes, leaf-list nodes, container nodes, list nodes.
- Leaf nodes: contains simple data structures such as integer or string.
- leaf-list nodes: sequence of leaf nodes with one value of particular type
- container nodes: used to group related nodes in a subtree
- list nodes: defines a sequence of list entries. Each entry is like a structure or a record instance.

IoT system Management with NETCONF-YANG :-

management system: The operator uses a management system to send NETCONF messages to configure the IoT devices & receives state information & notifications from the device as ~~follows~~ NETCONF messages.

Management API: allows management applications to start NETCONF sessions, read and write configuration data, retrieve configurations & invoke RPCs

Transaction Manager: executes all the NETCONF transactions and ensures the ACID properties hold true for transactions. (Atomicity, Consistency, Isolation, Durability)

Rollback Manager: Responsible for generating all the transactions necessary to rollback a current configuration to its original state.

Data model Manager: keeps track of all YANG data models, corresponding managed objects & applications which provide data for each part of model

Configuration validator: checks resulting configuration after applying a transaction to be valid configuration or not.

Configuration Database: contains both configuration & operational data.

Configuration API: using this, read configuration data from configurational datastore, write operational data to operational datastore by applications on IoT device

Data Provider API: applications can register for callbacks for various events and report statistical & operational data by this API

NETOPEER:

* It is set of open source NETCONF tools built on the Libnetconf library.

Netopeer tools:

- Netopeer-server: It is a NETCONF protocol server that runs on managed device. Provides environment for configuring the device using NETCONF RPC operations.
- Netopeer-agent: It is NETCONF protocol agent running as a SSH/TLS subsystem. It accepts incoming NETCONF connection and passes the NETCONF RPC operations received from NETCONF client to NETOPEER server.
- Netopeer-cli: It is a NETCONF client that provides a command line interface for interacting with netopeer-server. To send NETCONF RPC operations for configuring the device & retrieving the state information.
- Netopeer-manager: Allows managing the YANG & Libnetconf transaction API modules on the NETOPeer-server. modules can be loaded or removed from the server by it.
- Netopeer-configurator: It is a tool that can be used to configure the Netopeer-server