Software Refined Network (SDN)

~~Earier~~ New way to managing computer Networks

- Earier to manage
- More flexible to control.

Three Planes

Low level

Data Plane ~~Frame~~ Packet Forwarding (Frame / Packet, TCP/UDP Segment, datagram)

Focuses on configuring and monitoring N/w devices

High Level

Control Plane Routing decisions (OSPF, BGP, MPLS)

Focuses on data forwarding packets based on routing decisions.

Management Plane Configuration, Security, Monitoring. (SNMP, Telnet, SSH)

Focuses on managing the overall network topology.

Traditional Routers + Switches → Control logic + Data Forwarding

SDN Switches —Data Plane→ Packet Forwarding

SDN Controller —control Plane→ control logic

- SDN Does not eliminate H/W Switches & routers →
- changes the functioning

SDN controller Located in

Network core or cloud

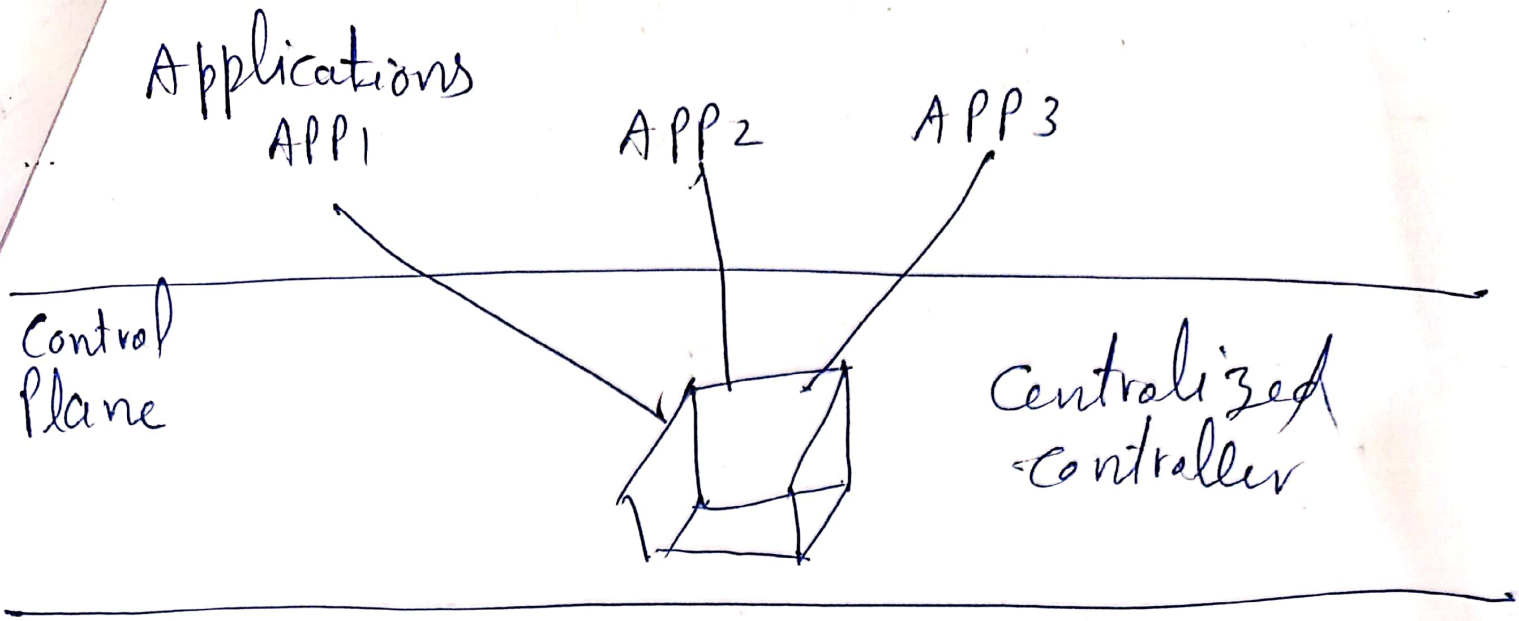
cloud bared Networks :- AWS, Google cloud, Microsoft azure

Enterprise data centres :- Cisco ACI, VMware NSX

Telecom Networks — 5G core NIWs, Edge data centers

SDN Switches located in Network Edge & core

- Packet forwarding bared on rules provided by SDN controller
  - Located in Data centers
  - Enterprise Networks
  - IoT Networks
  - Tele com networks



Question: A cloud data center has ~~1000 servers~~ connected through 50 traditional NW switches. Each switch operates i

Advantages of SDN over Traditional Networking

- Centralized control.
- Programmability
- Automation

(1) Centralized control

Trad.
- Each switch/router makes independent decision
- Local stored (table based) routing
- Complex difficult to manage.

SDN
- Web centralize SDN controller.
- Manages all H/W devices dynamically.

- Centralize firewall (Unlike each device having own firewall)
  - Overcom DDoS Attack

- SDN bare Software automate network configuration
  (traditional - manually)

01.4.25

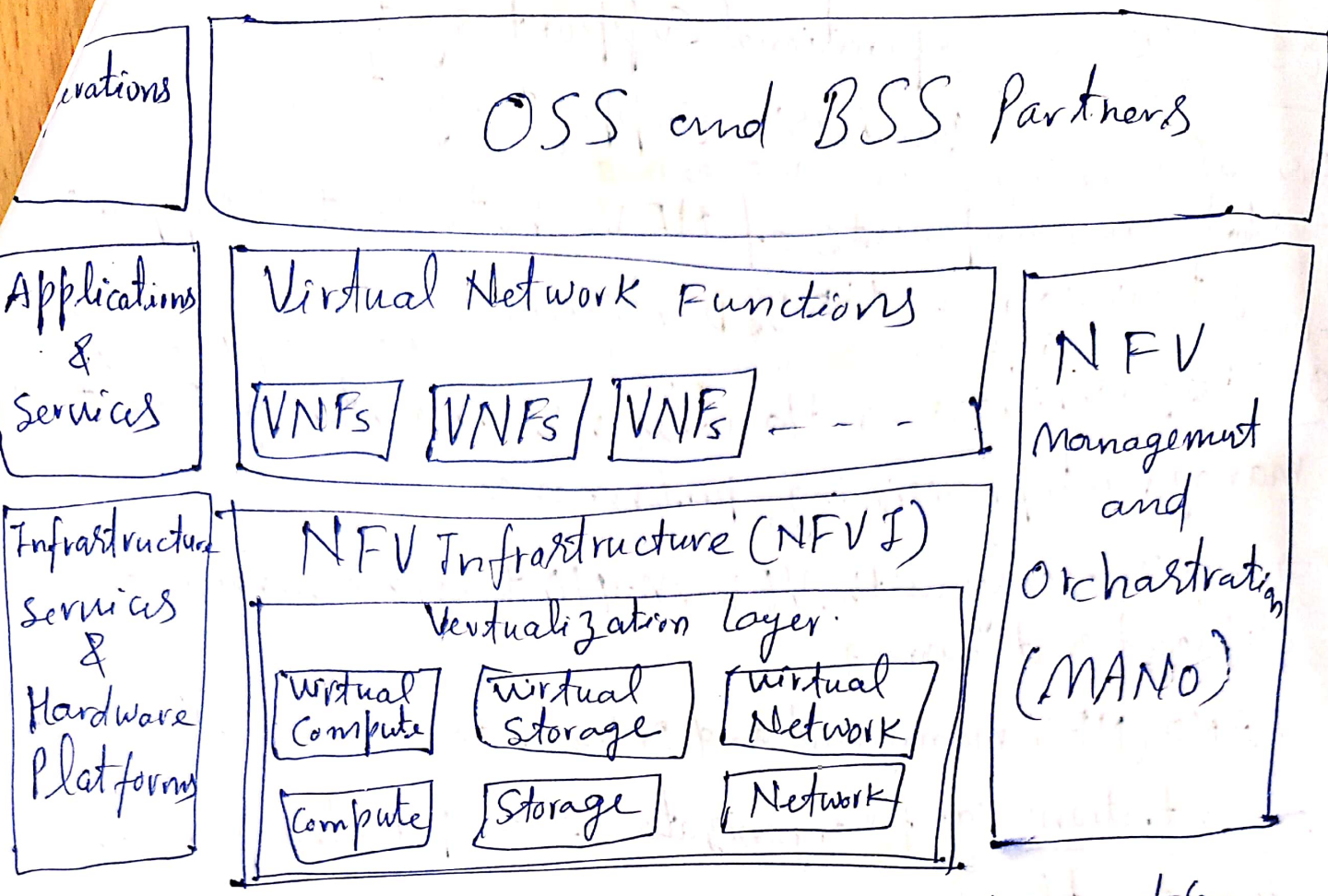
~~12,15,24,01,81~~ ~~10,65,77,63,89,114~~
~~57,20,05,113~~

Results : criteria to category definition
group multiple criterias & make single criteria.

① * Generic criteria
Criteria Importance matrix
② Different Ranking models comparision
- WSM
- Weighted Sum
- " Product
- AHP
- TOPSIS

- Ankit - Nirwal
- Sachin Jhipp
- Shubham chand

- Different weight Assignment Technique (Identify better weight assignment technique)



Concept introduced in 2012 in a conference at Germany

Network Function Virtualization :- (NFV)

- Replacement of n/w Hardware with virtual machines
- Hardwares like Routers, Firewalls.

Router → Virtual router (vRouter)
Firewall → " Firewall (vFirewall)
Load Blancer → " Load Banacer (vLB)
WAN optimizer → " WAN optimizer.
NAT → vNAT
Intrusion detection system (IDS) → vIDS
Intrusion Prevention System (IPS) → vIPS

OSS / BSS — operational Support system
Business Support system
(Business logic, billing, customer management)
- Not core components of NFVs but essential support systems to run NFV-based services
- Enable business & operational community by linking the NFV setup to org. goals & service delivery.

OSS Responsibilities ⇒ Handles n/w monitoring, Fault management, provisioning, performance

⇒ works closely with NFV MANO to ensure Services are deployed and running.

MANO (NFV management and Orchastration)

orchastration: careful organization of complicated plan or event done secretly. → (Planning & organization, coordination, control & manipulation)

BSS:
- Manages customer facing activities.
- which custome is using — How much Bw — VNF resources

ual Network Functions (VNF)

Performs actual n/w functionalities (routing, firewalling)

- Provide core networking and security functions without H/W
- Run in virtual environments on general purpose H/w.
- Deep packet inspection, traffic filtering, access control.

Network Function Virtualization Infrastructure (NFVI)

Foundation that hosts and runs all VNFs

(i) Virtual Compute

- Provides the processing power (GPU/CPU) via virtual machine or containers.
- VNFs are deployed on top of compute instances.
- Hypervisors (Virtual machines, KVMs) or container runtimes (Docker) manage virtual computes.

(ii) Virtual Storage.

- Persistent and temporary storage to VNFs
- Used for storing logs, configurations, databases or caching packet
- Supports fast data read/write (SSD, SAN, NAS)

(iii) Virtual Network

- Provides Network connectivity ~~based on~~ between VNFs and external networks.
- Implements overlay networks, virtual switches and SDN controlled paths.
- Ensures Isolation, QoS & Bandwidth allocation.

NFV Management and Orchestration (MANO)

- Brain of NFV architecture
- Automates and manages the resources dynamically

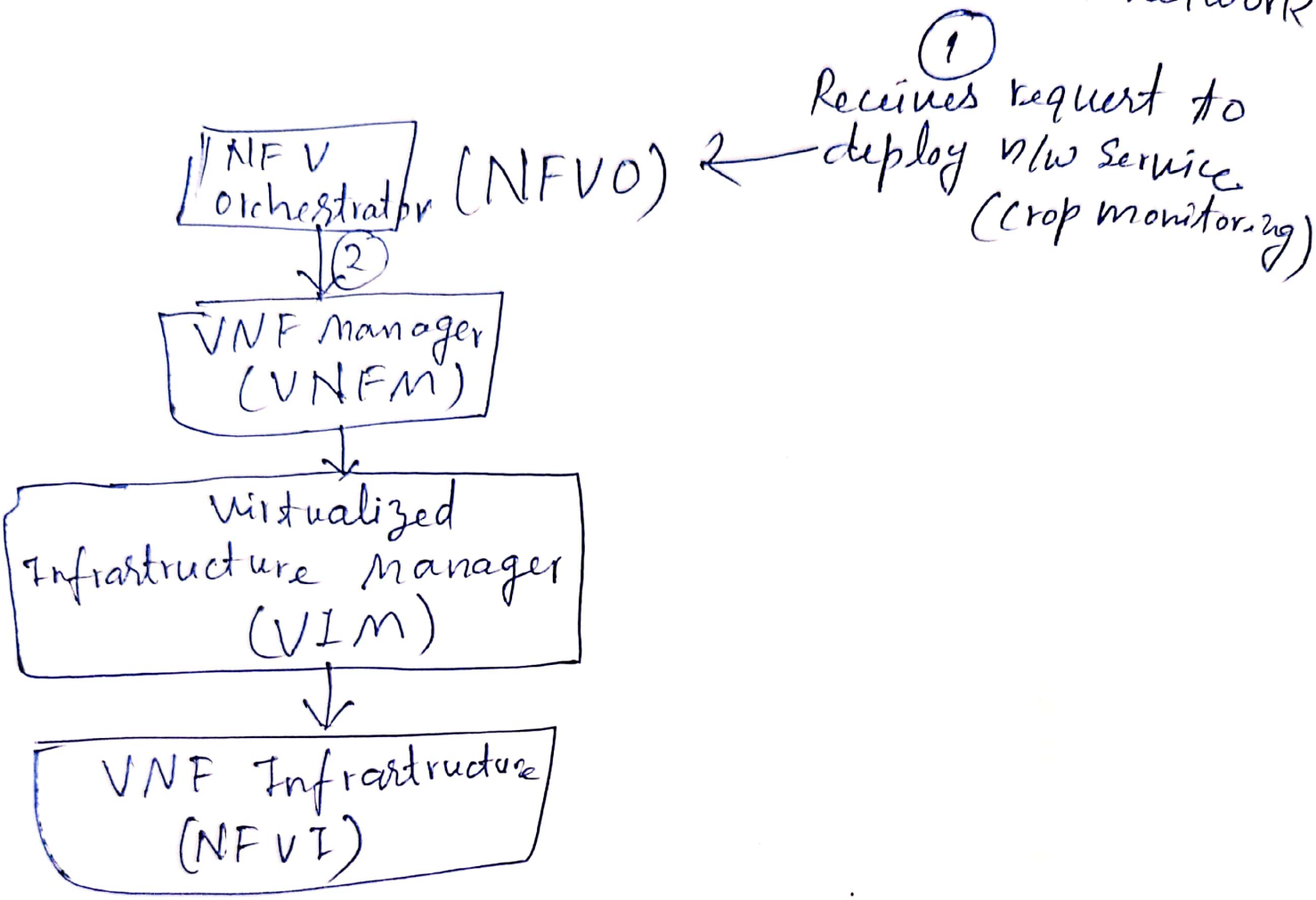
Components

1. NFVO (NFV orchestrator)
   - manages the lifecycle of services. (start, stop, update, scale, VNFs)
   - e.g. Launching a new firewall for a growing branch n/w.

2. VNFM (VNF manager)
   - Manages lifecycle of individual VNFs

③ VIM (Virtualized Infrastructure Manager)

- Manages the NFVI resources (compute, storage, network)



② → NFVO communicates with VNFM to intentiate the required VNFs (Moisture data Analysis, Fer

1, 9, 15, 20, 24, 45, 58, 63, 65, 81, 116, Kumkum

vantages of NFV for IoT

= Flexibility
- Scalability
- Low Latency
- Security
- cost efficiency

Traditional IoT gateway require:

Physical Firewall, Physical Router, Physical load balancer

Component	use in IoT
vRouter:	Routes sensor data to the correct Gateway/Servers
vFirewall:	Provides lightweigh, Programmable security at the edge.
vLoadBalancer:	Balances load between multiple analytics server.
vIDT/vIPS:	Monitors and detects anomalies in sensor data traffic.
vNAT:	Translates IPs to help IoT devices Communicate over the Internet.

Edge: Physical location where data is generated and initially processed (oftnen near the source)

- Instead of Sending all the data to the cloud/datacenter edge computing procsses data closer to where it is generated.
  - Reduce latency, Save BW, improve response time,
  - Enhance security.

Edge Computing ⇒ Distributed computing paradigm that brings computation and data storage closer to data sources.

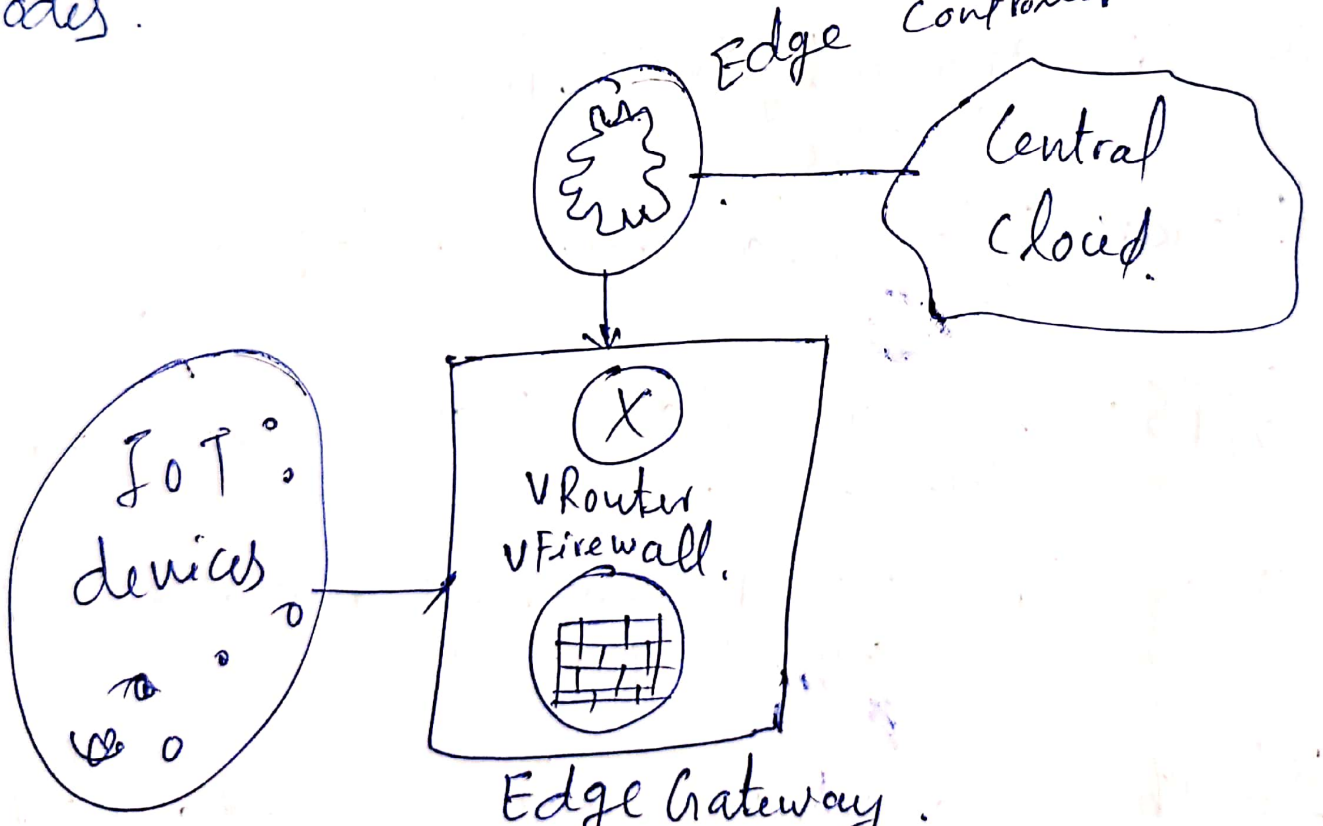
Advantages

(i) Reduce latency (ii)

Install Edge nodes / IoT Gateways (Intermediate computing Platforms)

⇒ Edge nodes are usually more powerful nodes with more computational power.

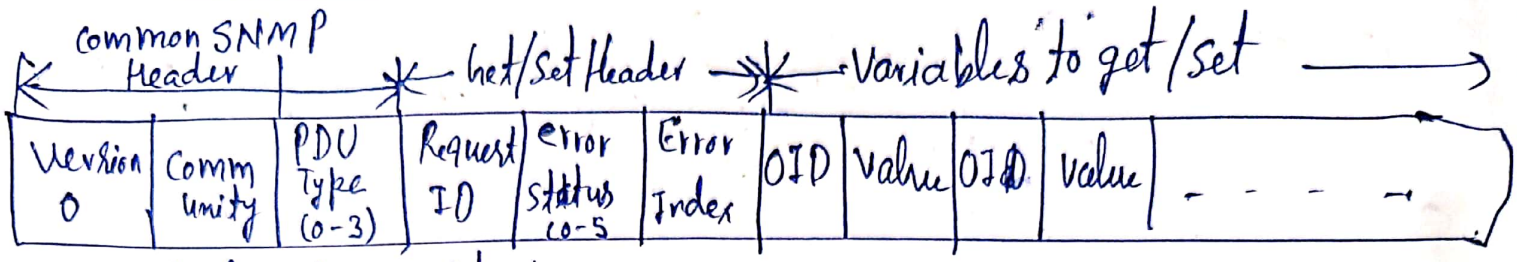
We can install the functionality of NFVs on edge nodes.



Edge Computing.

Simple Network management Protocol. (App. layer Protocol)

UDP Protocol



Centralized Monitoring:

- Standard protocol to monitor and manage the networking devices (Routers, Switches, IoT gateways & Servers) load balancer.

Real Time Fault Detection

- Allows administrators to receive alerts when fault or anomaly occurs. (device down, High CPU usages, Interface errors)

Remote configuration & Control:

- Supports configuration remotely: (change IP address, enable/disable interface)

Scalability in large Networks

- Supports Thousands of devices in enterprise or industrial IoT setup.
- lightweight & fits well in constrained environment

Supports IoT and Edge devices.

- In IoT, Helps to monitor gateways, edge devices for security availability & performance.

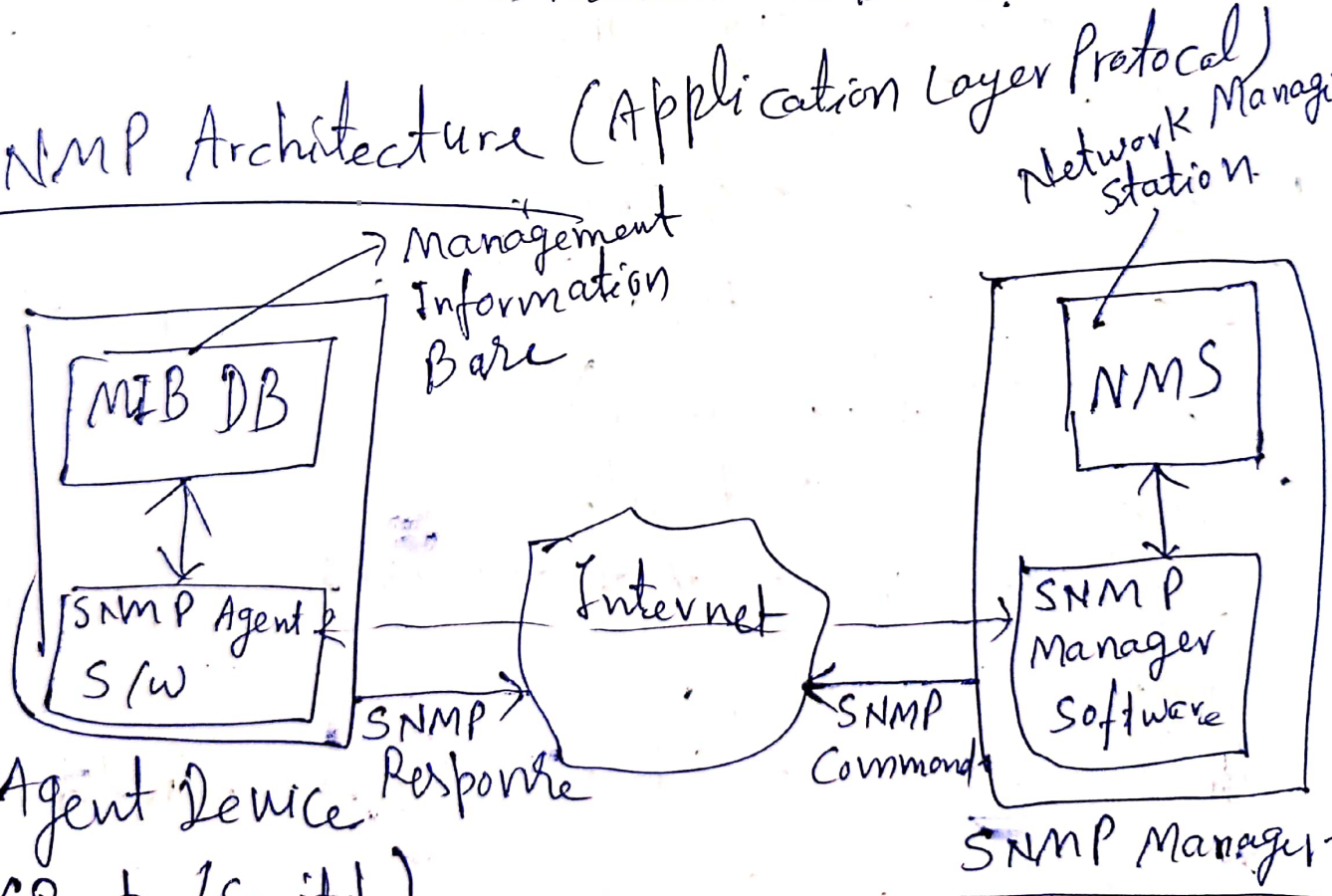
Smart Agriculture – Monitors weather stations, water
Gateways.

Smart cities – Monitors traffic lights, Smart poles, WiFi APs
Monitor Parking.

Smart Healthcare

Industrial IoT (IIoT) Programmable Logic Controllers (PLCs)
Factory devices, cooling Systems,
alert maintenance team.

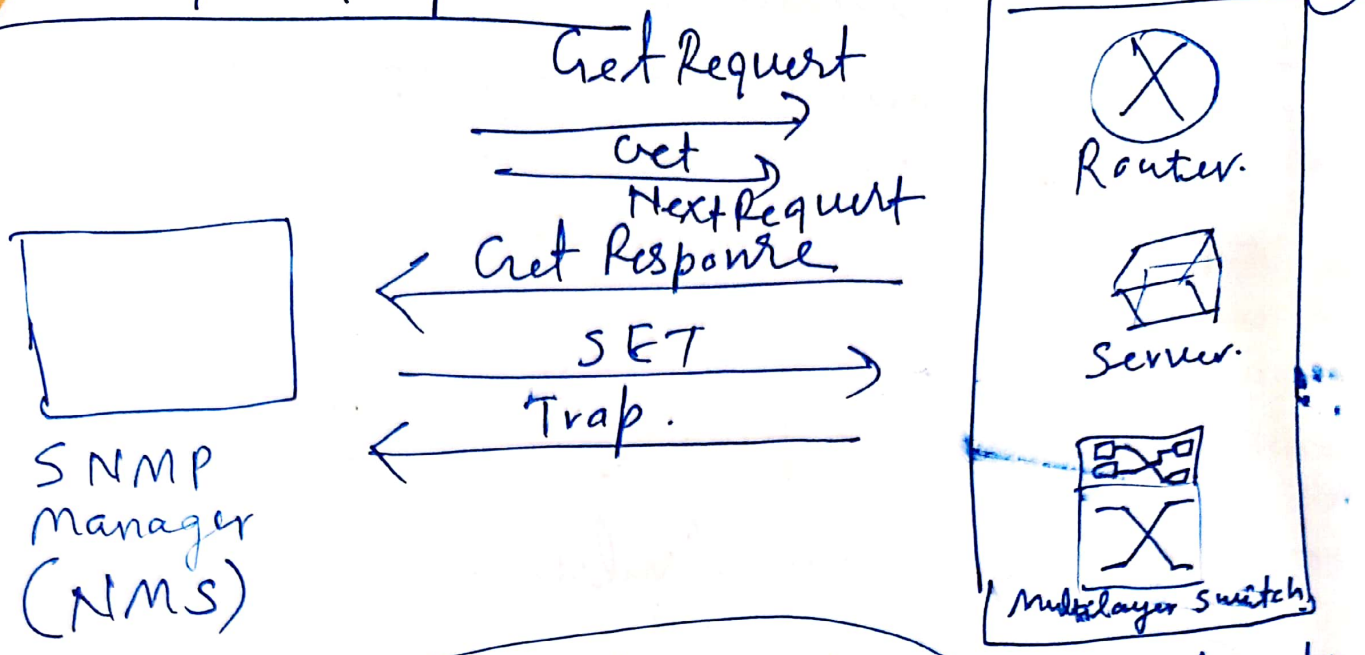
SNMP Architecture (Application Layer Protocol)



⇒ Get Request
⇒ Get Next Request
⇒ Set Request

⇒ Response
⇒ Trap
⇒ Inform Request

MP Request Response



SNMP messages (Monitoring, Alerting & reporting)

GetRequest: manager → agent
To Retrieve data from SNMP Agent

GetNextRequest: To getValue of next Variable
Manager → Agent

SetRequest: Modify A variable
Manager → agent

GetResponse: Agent → Manager
Return a variable value

Trap: consolicisted alert from agent
~~Agent~~ → ~~Manager~~
Manager ← Agent.

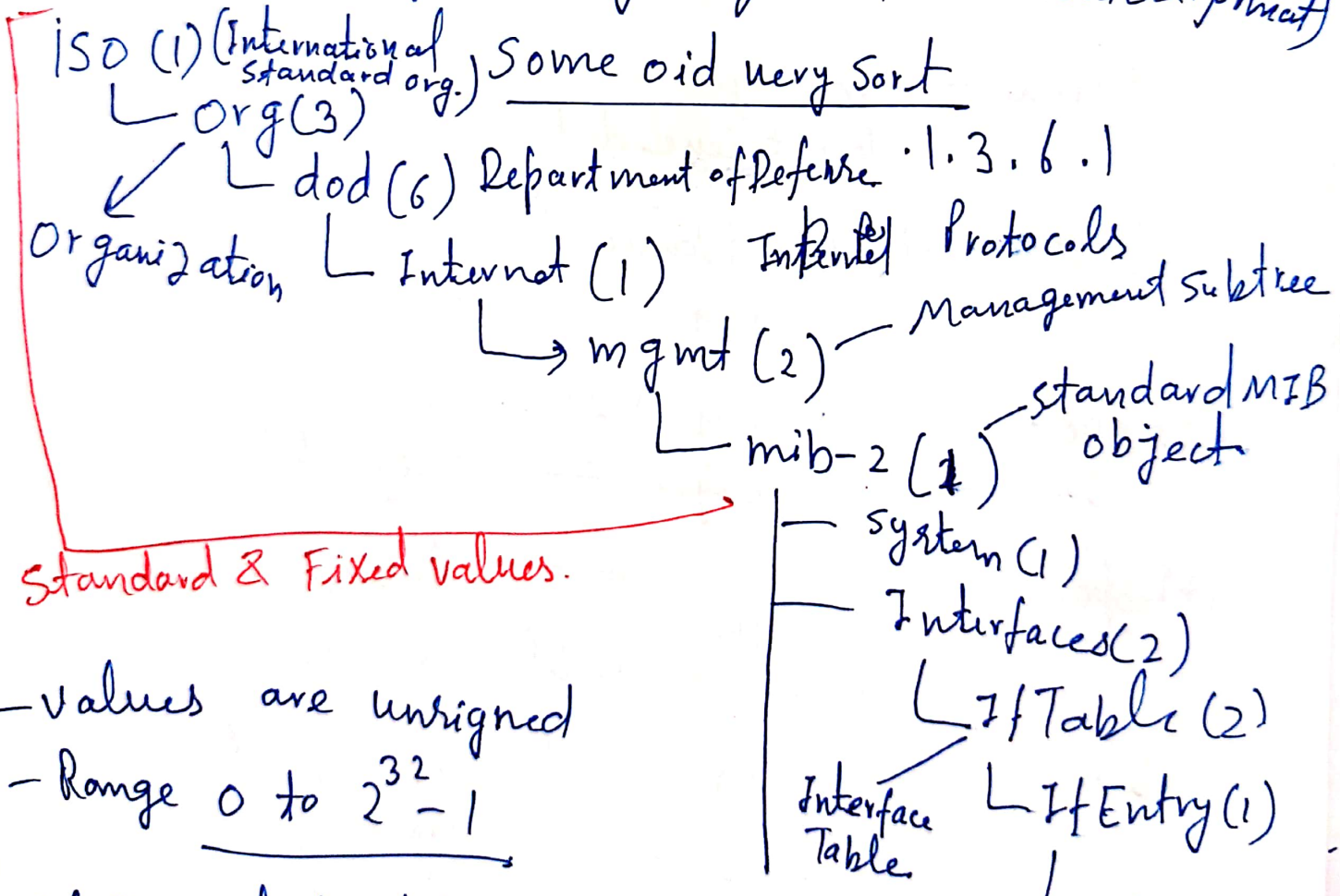
InformRequest: Agent ↔ Manager
Similar to trap but receive Ack from manag (expect)
Agent

OID: object identifier

- Globally Unique identifier for each managed object in an SNMP-enabled device.
- Represent in dotted notation.
  .1.3.6.1.2.2.1.2.1.2

Hierarchical Path from ~~OID~~ ISO to MIB

- Each Dotted numerical value represent node in hierarchical tree.
- No fixed length (Any length in hierarchical format)



- values are unsigned
- Range 0 to $2^{32}-1$

If I want to take any OID then

.1.3.6.1.4.1.<

4 — Private
1 — enterprise
My enterprise number decision

Suppose I want to create OID for my IoT agri field.

.1.3.6.1.4.1.9999 → My company Id
- 1 Devices ← Switches, Routers, load balancer
- 2 Sensors
  - 1 - Temperature Sensor1
  - 2 - Temperature Sensor2
  - 3 Humidity
  - 4 Nitrogen - - -

SNMP Architectural devices

(1) SNMP Manager

- Central controller in SNMP
- Sends queries to agents, collects data and make decisions.
- Configure device setting remotely

E.g. "Ask all routers status in every 10 minutes").

(2) SNMP Agent (Software module)

- S/w running on managed devices (router, Printer, switches, IoT devices)
- Responds on manager's requests & sends alerts.
- Store & update data in MIB
- Proactively send alerts (trap/inform) if threshold breached

"Send a trap when CPU temp. reaches 75°"

Network management system (NMS)

- App/Platform for Administrators
- Includes dashboard, visualization tools, logs.

MIB

- Structured DB of N/w object managed by SNMP
- Contains definitions of all manageable elements for a device. → each device having OID

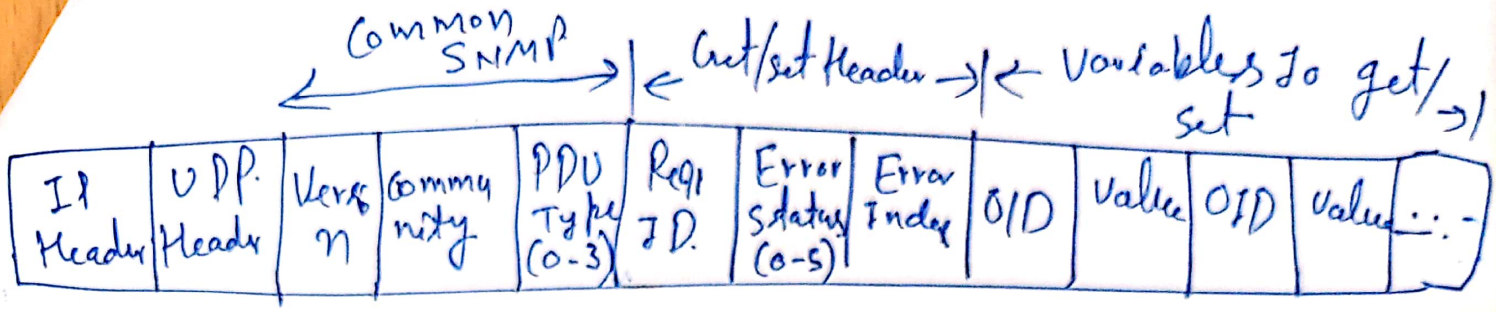
Numerical Example:

- A Green House Setup — 10 IOT Temperature sensors.
- Each sensor has an SNMP Agent
- And, A central NMS (SNMP manager) Polls them every 10 minutes
- OID for Temp. reading ⇒ .1.3.6.1.4.1.5555.1.1.1
- Acceptable temperature 18°c to 30°c
- Sensor 3 reports 35°c
- SNMP trap threshold >30°c

Now,

- NMS send getRequest for temperature to each sensor
  getRequest (.1.3.6.1.4.1.5555.1.1.1)
- Sensor 3 sends,
  getResponse (.1.3.6.1.4.1.5555.1.1.1.3, 35)

SNMP Header



PDU

Core messages

0	GetRequest	V_1, V_{2C}, V_3	Req. value of variable	M→A
1	GetNextRequest	"	Req. next variable in MIB	M→A
2	get Response	"	Response to get, getNext or set	A→M
3	Set Response	"	Set the value of variable	M→A
4	Trap	V_1		
5	GetBulkRequest	V_{2C}, V_3		
6	Inform Request	V_{2C}, V_3		
7	SNMPv2 Trap	"		
8	Report	V_3		

Error Status

0	No error	V_1, V_{2C}, V_3	
1	TooBig	"	Response too large to transport
2	No Such Name	V_1	variable does not exist
3	Bad Value	V_1	specified value not correct
4	Read only	V_1	Trying to write Read-only variable
5	genErr	V_1, V_{2C}, V_3	A general error.
16	No Access		
17	Wrong Type		
18			

Version

0	V_1	– original, minimal security (community)
1	V_2C	– Improved Performance & error handling
2	V_3	– most secure, support authentication & encryption

Community in V_1 & V_2C only. Defines Access level

Public – Read only

Private – Read-write access

custom setting.

NETCONF YANG

29.4.25 {10, 12, 15, 24, 45 (1) 58, 63, 65, 77, 81, 85, }

Network Configuration (Protocol)

Yet Another Next generation (Data Modeling language)

- → session based management Protocol
- → Manages Network devices (Routers, Switches, Printers, Servers)
- → Overcomes the limitations of SNMP

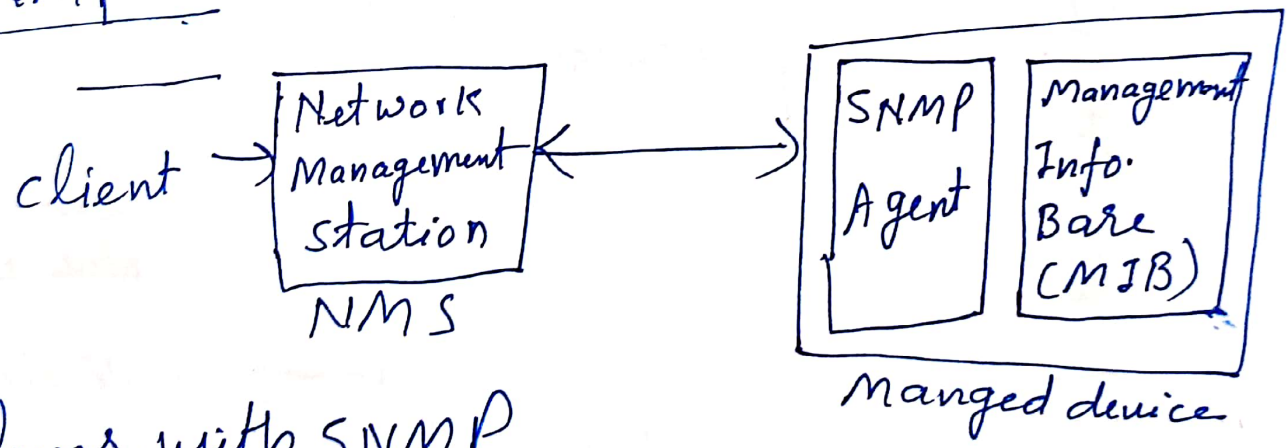
Developed by IETF to { Install, Manipulate, delete } Configurations

Revised
- Dec-2006 (RFC-4741)
- Jun 2011 (RFC-6241)

NetConf client : Network Management System (NMS) (Sends request)

NetConf Server : Devices (Routers, Switches, printers) Processes and responds.

In SNMP



Problems with SNMP

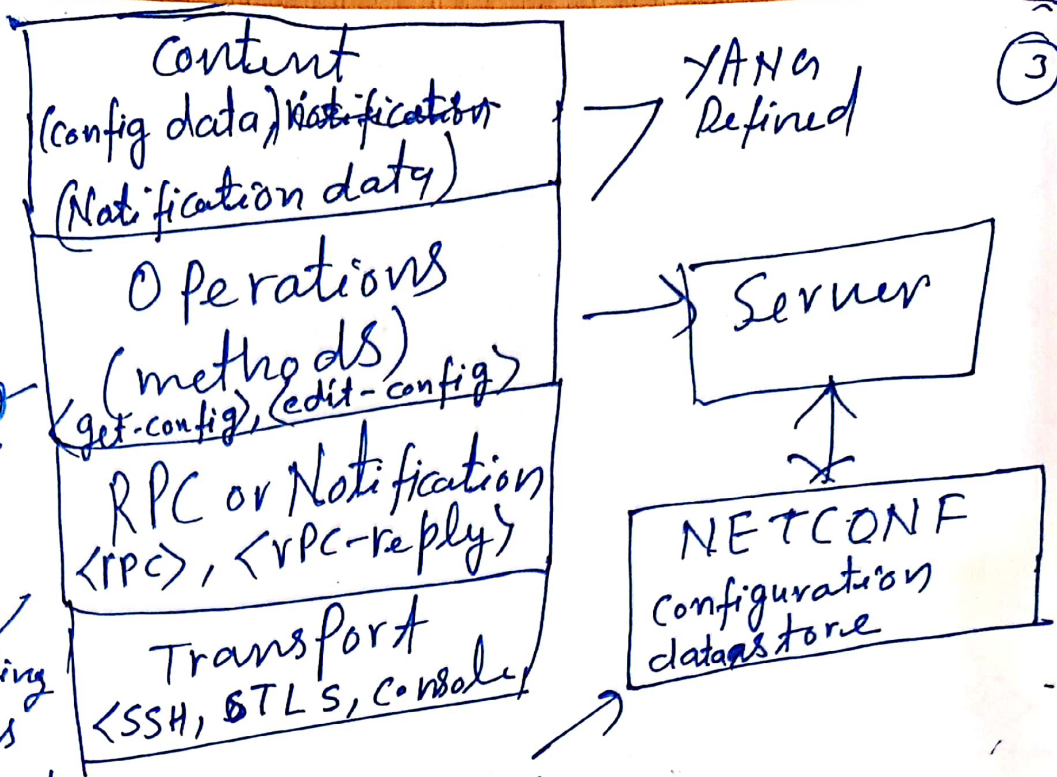
- Poor at bulk configuration changes (had to change

DataStore: - storage area inside a device (router, switch, IoT gateway) where configuration data or state information is kept

Example:	running	Startup	Candidate,	intended (Optional)
(Optional) Operational actual State	Current live Configuration	Configuration used at boot time	temp. area to Prepare configuration	Intended configuration goal

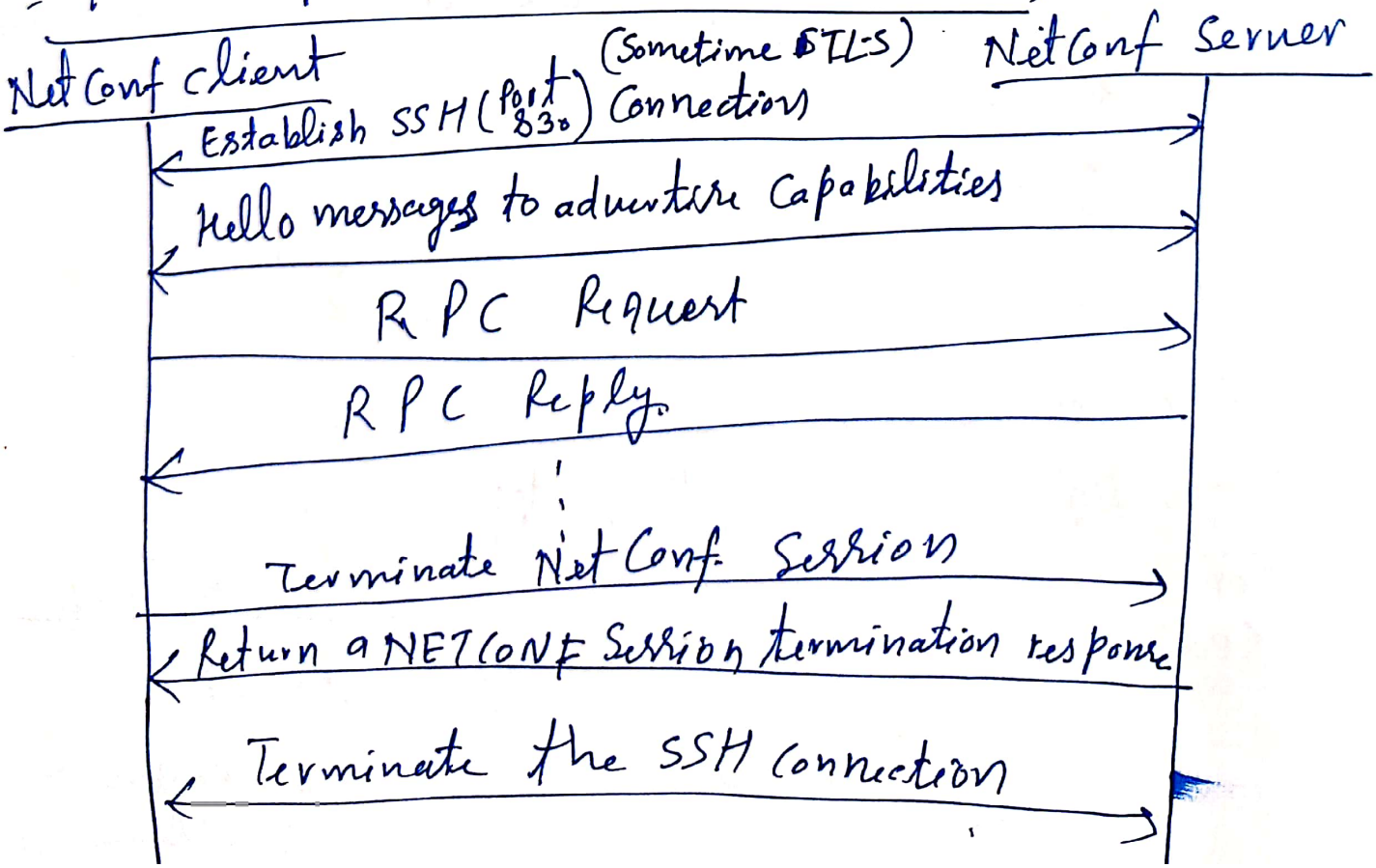
Differences between SNMP and NetConf

Aspect	NetConf	SNMP
Purpose	configuration + Monitoring	Monitoring (Primary)
Data Format	XML / JSON (structured)	OID ASN.1 BER (binary) (Basic Encoding rule) → Abstract Syntax Notation
Data Model	YANG (Flexible, extensible)	MIB (Fixed / Hierarchical)
configuration Handling	Full / Partial config set; Transaction support	object id (OID) variable by variable based
Error Handling	Atomic operation (0 or 1) (All roll back / None roll back)	Limited error Handling ~~Partial (if Batch)~~
Transport Protocal	Secure (SSH, TLS) (formerly SSL)	UDP (sometime TCP)
Security	strong (SSH/SLS) Security	weak (Authentication)
	Remote Procedure call. what failed, why fail	- Reading device statistics - Weak error Handling (basic error codes NoSuchName, tooBig)
		~~Difficult~~ No. T



4 Layer structure: (1) Content (2) Operations (3) Messages
(4) Secure Transport

Request Response and Communication



Operations

<get> Retrieve Running configuration and device State information

<get-config> Retrieve all or Part of specified configuration datastore

<edit-config> Edit a configuration datastore by creating deleting, meging or replacing content

① <get> Example

```
<rpc message-id="1001" xmlns="urn:ietf:params:xml:ns
    netconf:base:1.0>
    <get/>
</rpc>
```

②

<get-config> : From specified datastore (running, condidate

```
<rpc message-id="1002"
    <get-config>
        <Source> <running/> </Source>
    </get-config>
```

③ Retrieve current router configuration settings.

<edit-config> : New network interface, update device parameter.

```
<rpc message-id="1003"
<edit-config> <target> <running/> </target>
    <Config> <interfaces> <interface>
        <name>eth0</name> <enabled>true</enabled>
    </interface> </ > </ > </ >
```

<copy-config>
Copy configuration from one datastore to another
(running to startup)

<rpc message-id="1004" ------>
<copy-config>
<source><running/></source>
<target><start up/></target>
</copy-config>
</rpc>

(10) <commit>
↓
to commit the transaction

(5) <delete-config>

<rpc message-id="1005" ----->
<delete-config>
<target><start-up*/></target>
</delete-config></rpc>

clear startup configuration

(6) <lock> — lock datastore to Prevent from Concurrent edits

<unclock> — unlock datastore after process complation

(7) <discard-changes> cancel uncommitted changes

(8) <close-session> End the NetConf session

(9) <kill-session> Fore kill a stuck session

Q. Consider Smart Agriculture IoT Network
- Managing multiple smart irrigation controllers installed in the field.
- All supporting NetConf over SSH.

Tasks:
(i) Update watering schedule on there devices (to 4 hrs duty cycle)
(ii) Ensure Safe changes.
(iii) Monitor device status
(iv) Handle errors if something goes wrong.

(a) Monitor status

```
<rpc message-id = "1011">
  <get> <filter>
      <value-status/>
      <Soil-mairture/>
  </filter> </get> </rpc>
```

Helps in deciding new watering schedule

(b)
```
<rpc message-id = "1012">
  <get-config>
    <Source><running/><Source/>
    <filter><watering schedule></filter>
  </get config>
</rpc>.
```
⇒ gives present watering schedule.

(a) - check live field mairture/value status
(b) - Retriene existing schedule
(c) - Prevent concurrent edits
(d) - change watering interval
(e) - Save ~~reboot~~ changes permanently
(f) - Revert if mistake found
(g) - End the session.

prevent configuration from concurrent access

```
<rpc message-id = "1013">
  <lock>
    <target> <running/> </target>
  </lock> </rpc>
```

(d) change watering schedule

```
<rpc message-id = "1014">
  <edit-config>
    <target> <running/> </target>
    <config>
      <watering-schedule>
        <interval> 4 </interval>
      </watering-schedule>
    </config>
  </edit-config> </rpc>
```

(e) unlock. // Same as lock

(f) Save permanently (even after reboot)

```
<rpc> - - ->
  <copy-config>
    <source> <running/> </source>
    <target> <startup/> </target>
  </copy config> </rpc>
```

(g) Discard changes

<rpc> - - -> <discard-changes> </rpc>

(h) close session

- - - - <close-session> - - - - ->

Q.1 Retrieve the current configuration of running datastore and startup data store

Q.2 Modify the routing configuration.
- update the routing configuration to add a new static route
  - Destination: 10.10.10.0/24
  - nextHop: 192.168.1.1
  - Route matric: 10.

Q.3 Commit the changes

Q.4 validate and save configuration

Q.5 Rollback changes if necessary.

1. <get-config> <source>
   <running />
   </source> </get-config>

ference between SSH and SSL

SSH:- Secure remote command line access
- Secure file transfer between systems

SSL/TLS: Primarily used for securing web traffic (HTTPS)
- Ensuring sensitive data like passwords, credit card information

YANG

YANG 1.0 RFC 6020 (2010)

YANG 1.1 RFC 7950 Aug - 2016

Data Model Language (Schema Language)

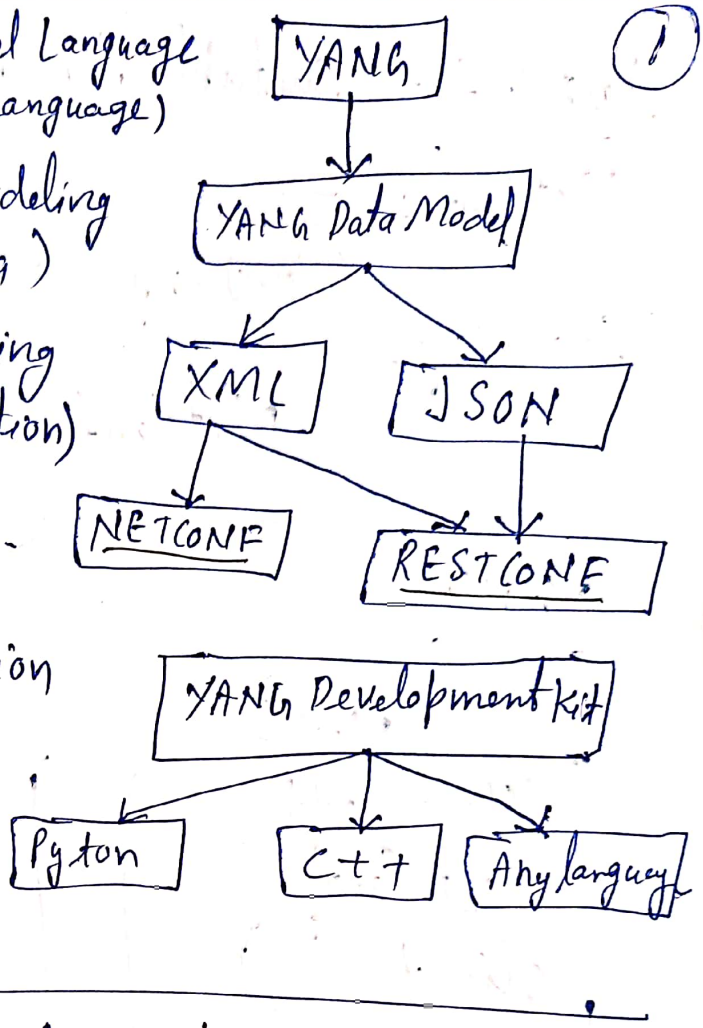
Data Modeling (Schema)

Encoding (Serialization)

Protocols

Application

Developed in Prog. language



Difference between NetConf and Restconf

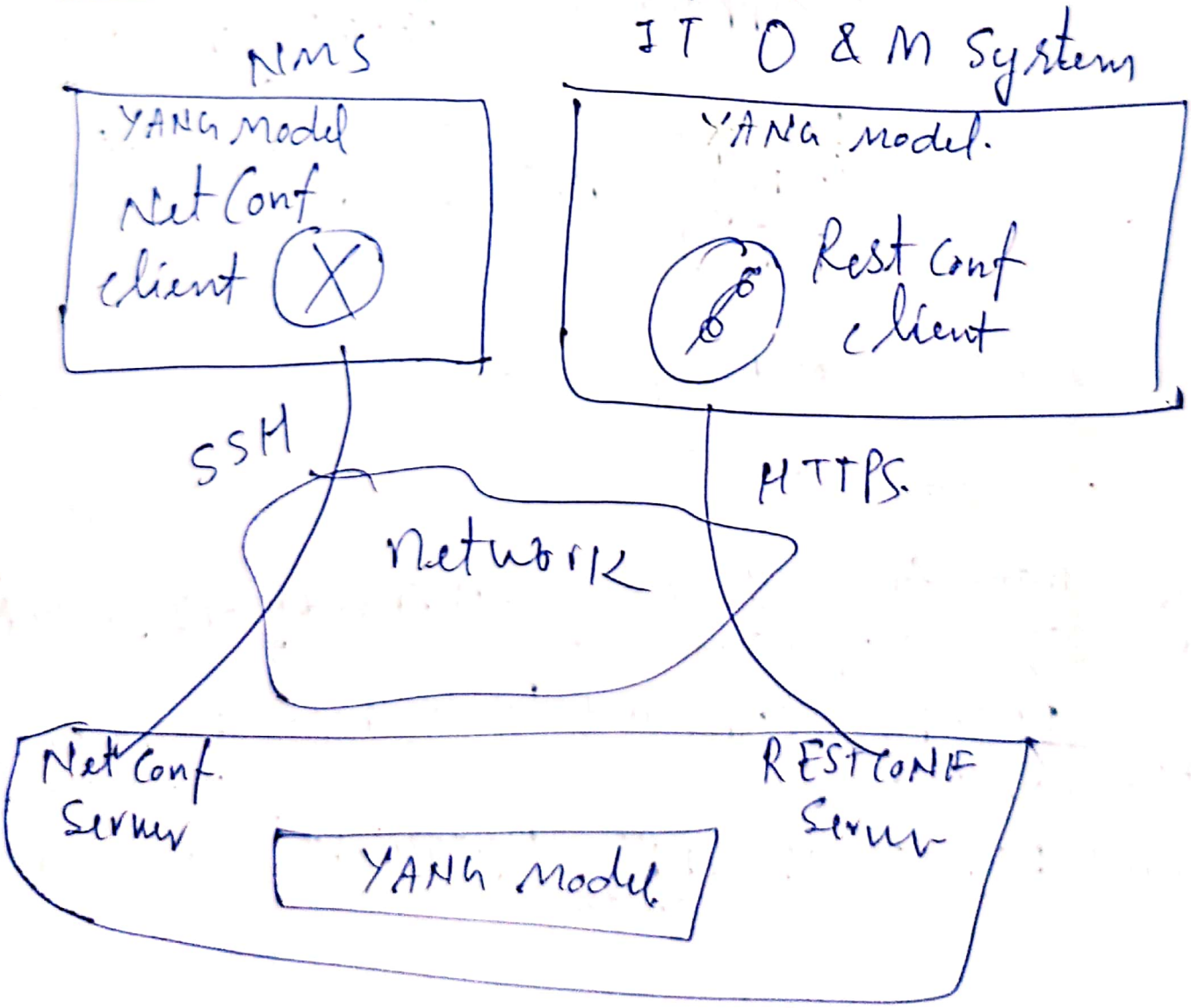
	NETCONF	RESTCONF
Design Need	Protocol for managing full device configuration.	Web friendly API for accessing YANG modeled resource
Data interaction	RPC-based (structured-XML operations)	Resource based (HTTP & RESTful API methods)
Overhead	High (SSH, XML, RPC structure)	Lightweight (HTTP, JSON/XML)
Example	A Service Provider wants to configure & manage 10,000 routers with precise roll back, locking & Configuration mechanism	An IoT Device dashboard needs to read sensor interface configuration and enable/disable certain services via API calls.

Rest Conf ⇒ Suitable for read-intensive tasks.

NetConf ⇒ ——————— configuration workflows.

Security < Restconf → & HTTPs
NetConf → SSH, TLS

YANG



30.4.25 Kumkum

10, 12, 20, 24, 63, 65, 77, 81
85, , 116

client (get-config) Request

```
<rpc message-id = "1001" xmlns="urn:ietf:params:xml:ns:
                                netconf:base:1.0">
 <get-config> <source> <running /></source>
        <filter type = "subtree">
          <interface xmlns="urn:ietf:params:ns:yang:ietf-
 </filter></get-config></rpc>.                          interfaces">
```

Server Response

```
<rpc-reply message-id="1001" xmlns="urn:ietf:params:xml:ns:
                                netconf:base:1.0">
 <data>
   <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-
                                                interfaces">
    <interface>
      <name> eth0 </name> <enabled> true </ enabled>
      <type xmlns:ianaift ="urn:ietf:params:xml:ns:yang:
                                          iana-if-type">
        ianaift:ethernetCsmacd
      </type></interface>
    <interface>
      <name> wlan0 </name><enabled> false </enabled>
    <type xmlns:ianaift= "urn:ietf:params:xml:ns:
                        yang:iana-if-type">
      ianaift:ieee80211 </type></interface>
 </interfaces></data> </rpc-reply>.
```