1. (10 points) Explain the motivation for developing COPS. What were the goals of COPS framework?

Ans:

The Common Open Policy Service (COPS) Protocol is part of the internet protocol suite as defined by the IETF's RFC 2748. COPS specifies a simple client/server model for supporting policy control over Quality of Service (QoS) signaling protocols (e.g. RSVP). Policies are stored on servers, and acted upon by Policy Decision Points (PDP), and are enforced on clients, also known as Policy Enforcement Points (PEP).

Motivation for developing COPS

The IETF started to address the problem gradually, strictly on a specific need basis. The first such need was the policy configuration in support of QoS. Network providers wanted to have a mechanism that would enable granting a resource based on a set of policy rules. The decision on whether to grant the resource takes into account information about the user, the requested service, and the network itself.

Goals of COPS Framework:
Goals of COPS framework A chief objective of this policy control protocol is to begin with a simple but extensible design. The main characteristics of the COPS includes:
• The protocol uses TCP as its transport protocol for reliable exchange of messages between policy clients and a server. Therefore, no additional mechanisms are necessary for reliable communication between a server and its clients.
• The protocol is extensible in that it is designed to leverage off self-identifying objects and can support diverse client specific information without requiring modifications to the COPS protocol itself. The protocol was created for the general administration, configuration, and enforcement of policies.

(References: https://en.wikipedia.org/wiki/Common_Open_Policy_Service &
https://www.zte.com.cn/global/about/magazine/zte-technologies/2002/2/en_328/161146.html)

2. (10 points) What feature is intrinsically new to COPs (compared to the SNMP)

Ans:

COPS employ a stateful client–server model, which is different from that of the remote procedure call. As in any client, server model, the PEP (client) sends requests to the remote PDP (server), and the PDP responds with the decisions. But all the requests from the client PEP are installed and remembered by the remote PDP until they are explicitly deleted by the PEP. The decisions can come in the form of a series of notifications to a single request.

COPS was designed to leverage self-identifying objects and therefore it is extensible. COPS also run on TCP, which ensures reliable transport. Although COPS may rely on TLS, it also has its own mechanisms for authentication, protection against replays, and message integrity. The COPS model was found very useful in telecommunications, where it was both applied and further extended for QoS support. As far as Cloud Computing is concerned, the primary application of COPS is SDN.

3. (30 points) Motivation for developing NETCONF.

Read RFC 3535, and answer the following questions (you can cite the relevant text of RFC 3535 verbatim). Each correct answer is worth 10 points.

a. Why the SNMP transactional model and the protocol constraints make it more complex to implement MIBs, as compared to the implementation of commands of a commandline interface interpreter?

b. What is the problem with the SNMP lack of support for easy retrieval and playback of Configurations?

c. List the operators' requirements for network management

Ans:

a:

- SNMP transactional model and the protocol constraints make it more complex to implement MIBs.
- A logical operation on a MIB can turn into a sequence of SNMP interactions where the implementation has to maintain state until the operation is complete, or until a failure has been determined.

b. SNMP does not support retrieval and playback. It is not easy to identify configuration objects.
c. List of the operators' required for network management are:

During breakout session, operators came to a conclusion that:
•It is necessary to make a clear distinction between configuration data, data that describes operational state and statistics.

• Some devices make it very hard to determine which parameters were administratively configured and which were obtained via other mechanisms such as routing protocols.

• It is required to be able to fetch separately configuration data, operational state data, and statistics from devices, and to be able to compare these between devices.

• There is no common database schema for network configuration, although the models used by various operators are probably very similar.

• It is desirable to extract, document, and standardize the common parts of these network wide configuration database schemas.

• It is important to distinguish between the distribution of configurations and the activation of a certain configuration. Devices should be able to hold multiple configurations.

(Reference: Cloud Computing: Business Trends and Technologies & https://mentor.ieee.org/802.11/dcn/16/11-16-1436- 01-0arc-yang-modelling-and-netconf-protocol-discussion.pptx)

4. (10 points) Does NETCONF use REST API in its Messages layer?

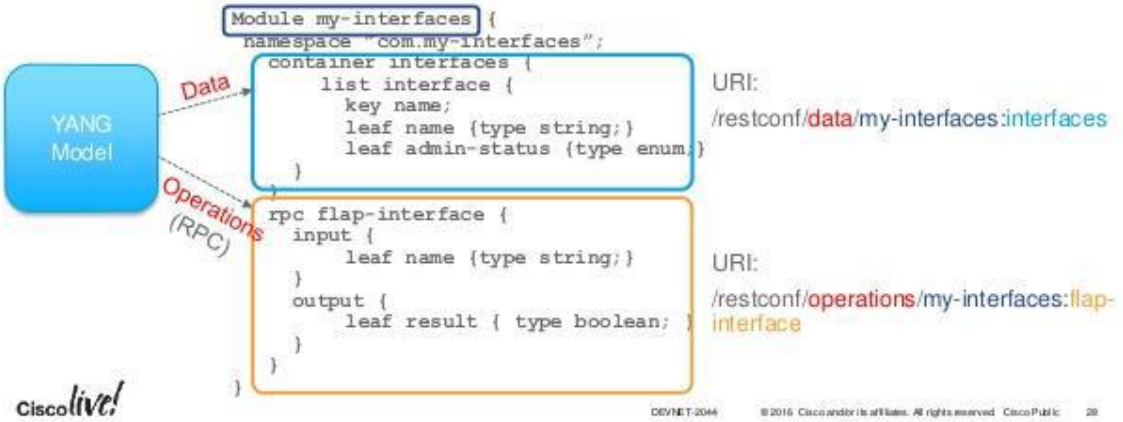
Ans:

No,A Restful protocol is used for accessing data defined in YANG using datastores defined in DATACONF.

RESTCONF API (1/4)

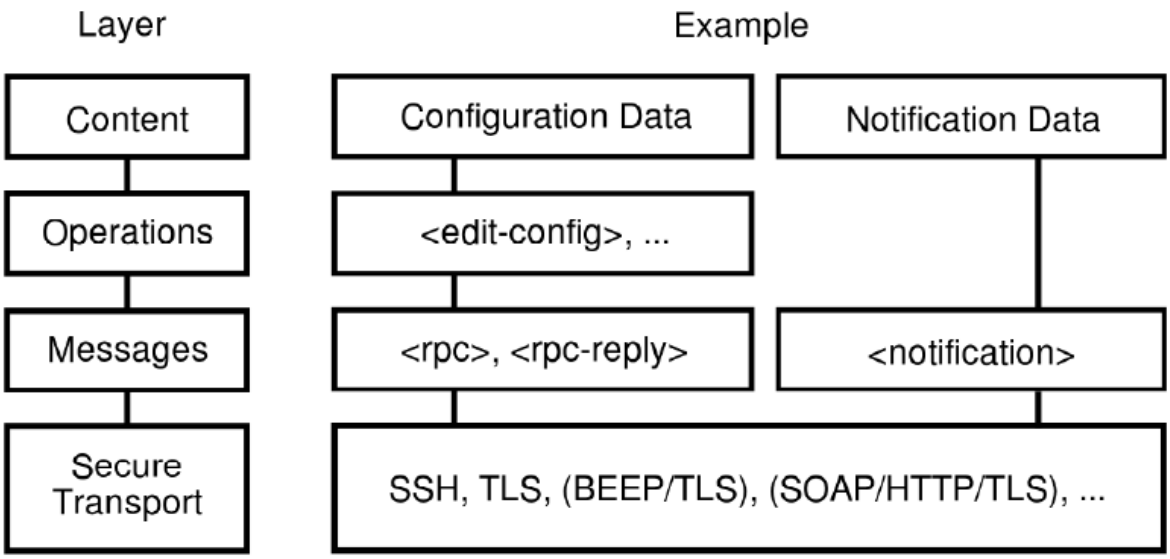
A RESTCONF URI is:

/<api-entry>/<resource-type>/<yang-module:resource>



REST vs Other Protocols

	REST	SNMP	NETCONF	SOAP
Data models		SNMP MIBs	YANG Models	
Data Modeling Language		SMI	YANG	WSDL
Management Operations	HTTP Verbs	SNMP Operations	NETCONF Operations	N/A
RPC Protocol Encoding	HTTP/XML/ JSON	BER	XML	XML
Transport Stack	SSL/HTTP/ TCP	UDP	SSH/TCP	SSL/HTTP/ TCP



From an architecture point of view, there are multiples API locations, all deduced from YANG modules. A controller typically configures network elements (routers, switches) based on the Network Element YANG modules: typically interfaces, routing, QoS, etc.). Note that, in the NETCONF and RESTCONF terminologies, the controller is the client and the network elements the server, as the controller initiates the configuration session.

(References: https://www.claise.be/2017/10/netconf-versus-restconf-capabilitity-comparisons-for-data-model-driven-management-2/)

5. (10 points) Name a de-facto modeling language for NETCONF. What document is it specified in? What is the name of this language's XML-based representation?

Ans:

A de-facto landuage for NETCONF is YANG, and it is specified in IETF 6020 under category Standard Track having ISSN 2070-1721 (https://tools.ietf.org/html/rfc6020).
The module is considered as base unit in YANG. A module defines a single data model. A module can define a complete, cohesive model, or augment an existing data model with additional nodes. The names of all standard modules and submodules MUST be unique.
Developers of enterprise modules are RECOMMENDED to choose names for their modules that will have a low probability of colliding with standard or other enterprise modules.
The name of XML-based representation of YANG is YIN Module.

(Reference: Cloud Computing: Business Trends and Technologies, https://tools.ietf.org/html/rfc6020 & https://www.ietf.org/proceedings/72/slides/netmod-5.pdf)

6. (10 points) List the steps involved in onboarding of an application.

Ans:

The actual process of onboarding ('forklifting' workloads) has seven relatively straightforward steps:
1. Defining the workload
2. Provisioning cloud resources
3. Establishing a connectivity bridge
4. Deploying the workload
5. Ensuring seamless two-way access
6. Testing and validating
7. Discontinuing the old service

(References: https://www.interxion.com/globalassets/_documents/_whitepapers/english/a-practical-guide-to-cloud-onboarding.pdf)

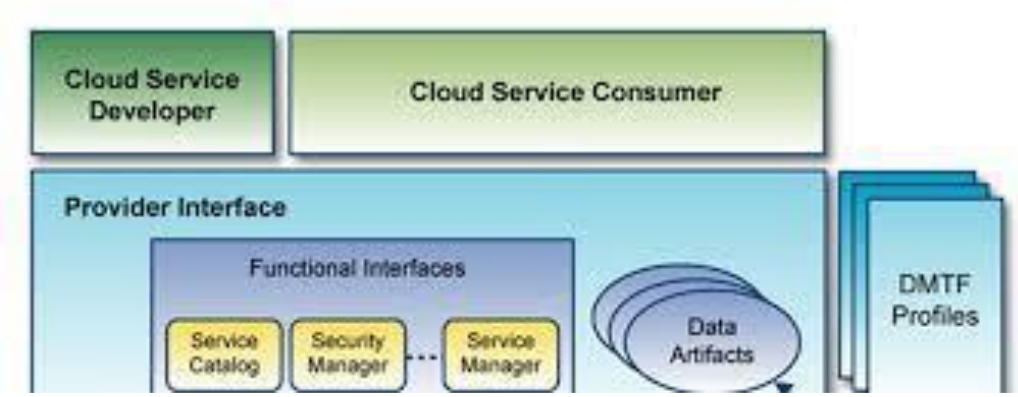
7. (10 points) List the actors involved in the service life cycle and its stages. What constitutes an offering?

Ans:

The three entities involved here are the Cloud service provider, the Cloud service developer, and the Cloud service consumer.

Suppose the instances for a load balancer and two servers have been created successfully but creating the virtual machine for the third server has failed. What should the user program do? Deleting all other instances and restarting again is hardly an efficient course of action for the following reasons. Assuming that all instances have been created, a service provider needs to support elasticity. The question is: How can this be (a) specified and (b) effected? Suppose each of the three servers has reached its threshold CPU utilization. Then a straightforward solution is to create yet another instance.

The solution adopted by the industry is to define a service in more general terms (we will clarify this with examples), so that the creation of a service is an atomic operation performed by the service provider— this is where orchestration first comes into the picture. And once the service is deployed, the orchestrator itself will then add and delete instances (or other resources) as specified in the service definition.

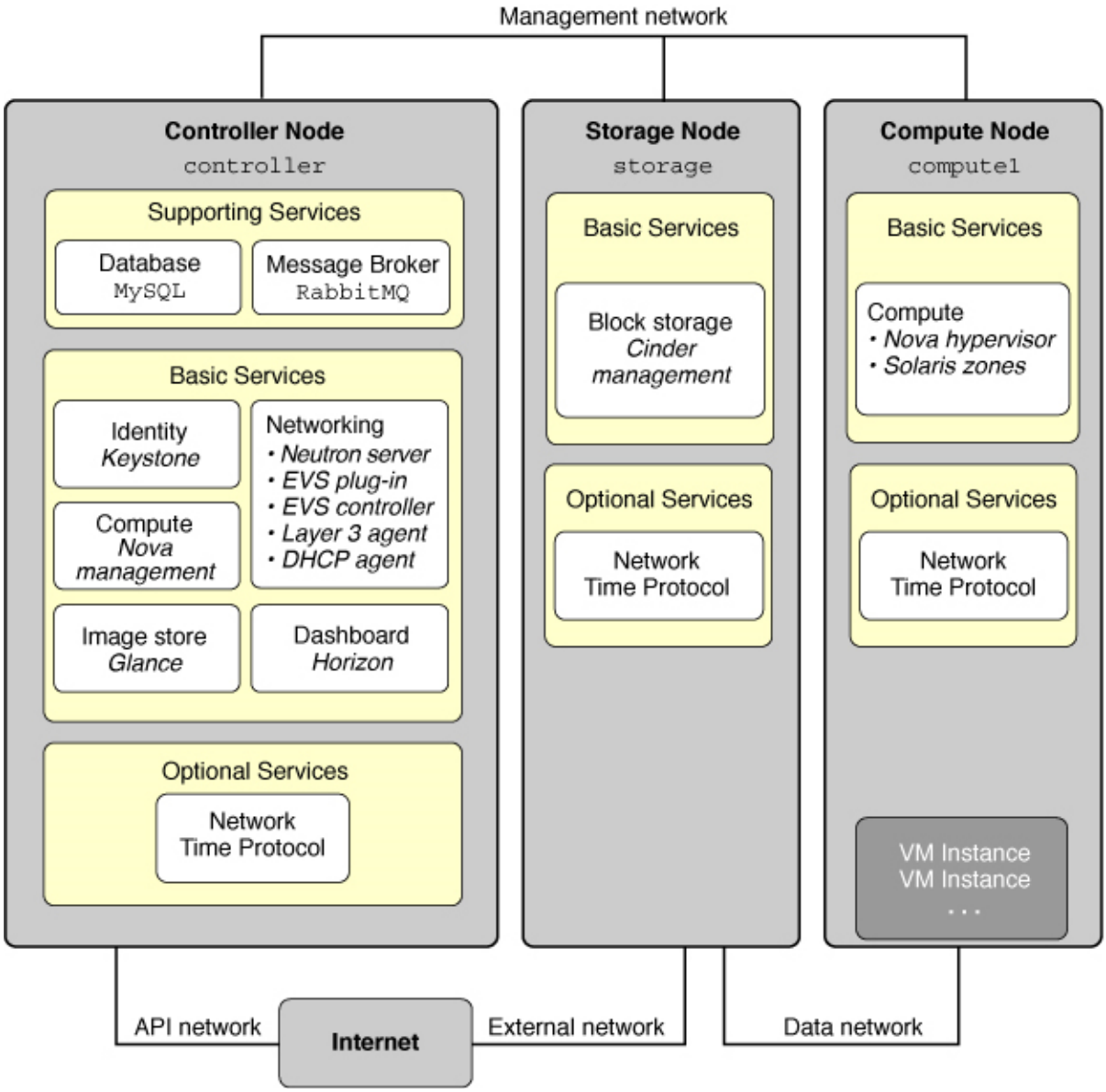


(Reference: Cloud Computing: Business Trends and Technologies)

8. (10 points) What is the name of the protocol used for communications among the OpenStack daemons?

Ans:

All OpenStack modules that have "API" in their names (i.e., nova-api) are daemons providing REST services (discussed in the Appendix). Communications among daemons are carried out via the Advanced Message Queuing Protocol (AMQP). AMQP can be initiated from either end of the pipe. On the contrary, an HTTP transaction can be initiated only by the client because HTTP is a pure client/server protocol.



(References: https://docs.oracle.com/cd/E65465_01/html/E57770/computecfg.html)