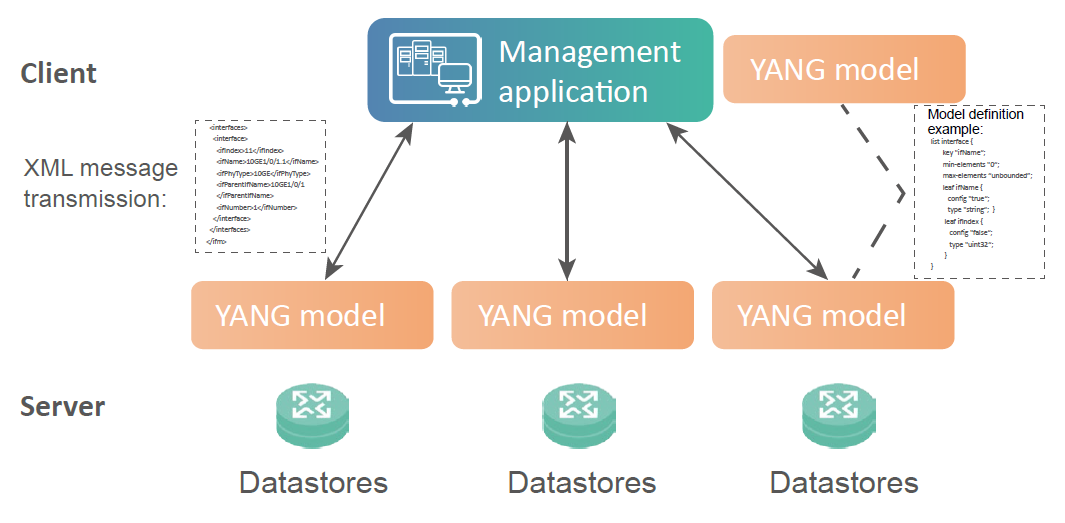
**Case Study – IoT System Management with NETCONF YANG**

**What are NETCONF and YANG:**

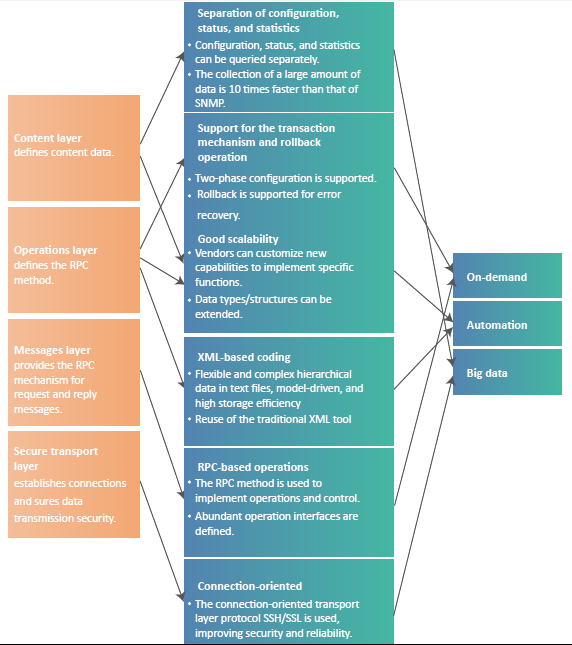
Network Configuration Protocol (NETCONF) is a network device management protocol that is similar to SNMP. NETCONF provides a framework for users to add, modify, or delete network device configurations, or query configurations, status, and statistics. Similar to SNMP that uses MIB files to model data, NETCONF uses Yet Another Next Generation (YANG) as a data modeling language to describe the interaction models between the NETCONF client and server. 

**Why are NETCONF and YANG required:**

One of the key network requirements of the cloud era is network automation, which includes quick, automatic, and on-demand service provisioning and automatic O&M. The traditional CLI mode and SNMP do not meet the requirements of cloud-based networks due to the following disadvantage:

* The traditional CLI mode is based on man-machine interfaces. The configuration is complex and varies with vendors. Therefore, the manual learning cost is high.
* The SNMP configuration efficiency is low, and the transaction mechanism is not supported. Therefore, SNMP is often used for monitoring.

1. **NETCONF uses a hierarchical protocol framework, better meeting the on-demand, automatic, and big data requirements of cloud-based networks.**



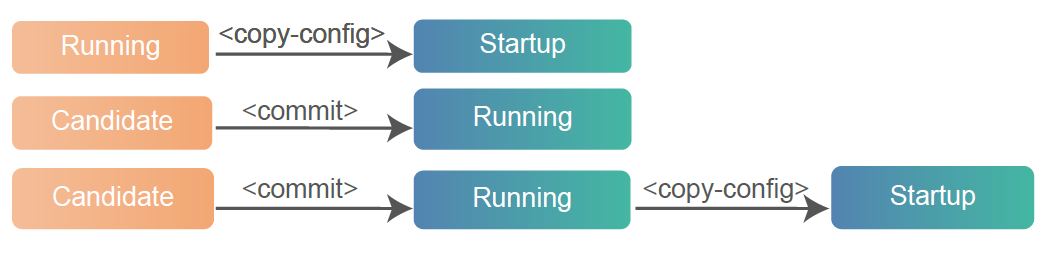
1. **NETCONF supports classified data storage and migration, phase-based committing, and configuration isolation.**

<running/>: <running/> configuration datastore that stores the complete set of active configurations of a network device.

<candidate/>: <candidate/> configuration datastore that stores various configuration data to be committed to the <running/> configuration datastore. Changes in the <candidate/> configuration datastore do not directly affect the involved device.

<startup/>: <startup/> configuration datastore that stores configuration data loaded (similar to a saved configuration file) during device startup.

Data can be migrated between configuration database.



1. **NETCONF defines abundant operation interfaces and supports extension based on capability sets.**

Basic Operations Supported by NETCONF (RFC 6241)

**<get>:** obtains part or all of the running configuration data and status data from the <running/> configuration datastore.

**<get-config>:** obtains configuration data.

**<edit-config>:** creates, modifies, or deletes configuration data.

**<copy-config>:** replaces a configuration datastore with the contents of another complete configuration datastore.

**<delete-config>:** deletes all data in a non-running configuration datastore.

**<lock>:** locks the configuration datastore of a device. A locked configuration datastore cannot be modified by other NETCONF users.

**<unlock>:** unlocks the configuration datastore of a device.

**<close-session>:** terminates a NETCONF session gracefully.

**<kill-session>:** forcibly terminates another NETCONF session.

Capabilities that can be extended:

RFC 6241: Writable-Running, Candidate Configuration, Confirmed Commit, Rollback-on-Error, Validate, Startup, URL and XPath.

RFC 5277: Notification and Interleave.

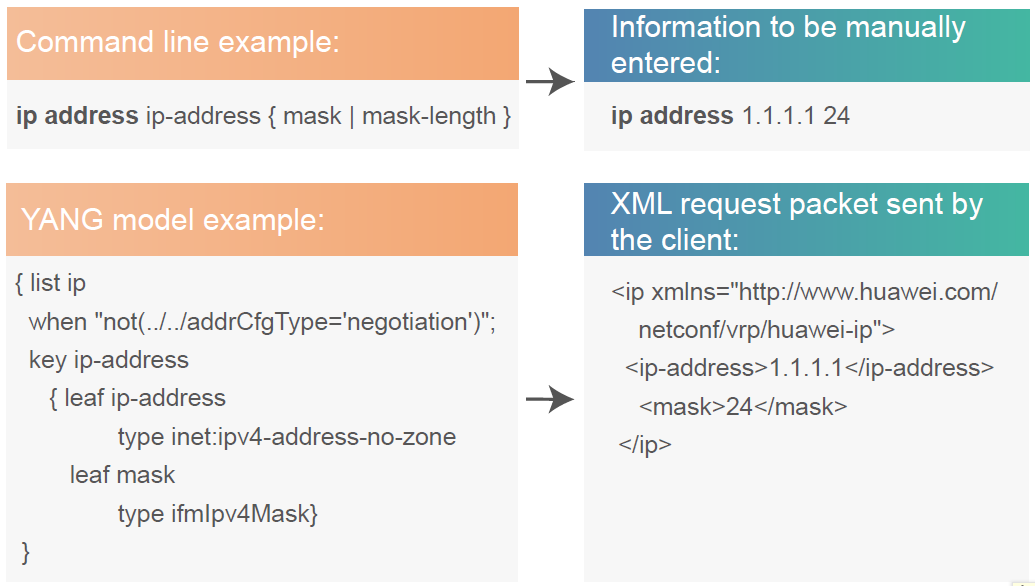
RFC 6243: with-defaults.

RFC 6022: IIetf-netconf-monitoring

1. **NETCONF operations are performed based on the YANG model.**

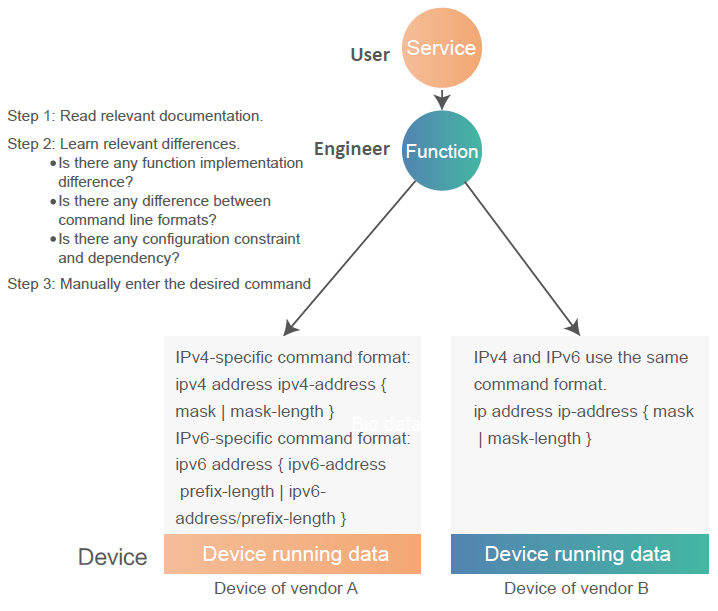
The YANG model defines device function configuration templates. Compared with the CLI mode, the YANG model has the following advantages:

* **Rich definition:** Various basic data types and data attributes can be defined.
* **Machine language:** Structured definition is adopted, allowing users to define constraints that can be directly identified by the computer without manual intervention.
* **Good scalability:** The grouping, refine, augment, and typedef statements are supported for users to extend YANG models and data types.
* **Easy integration:** The IETF defines multiple YANG models and data types for reference.

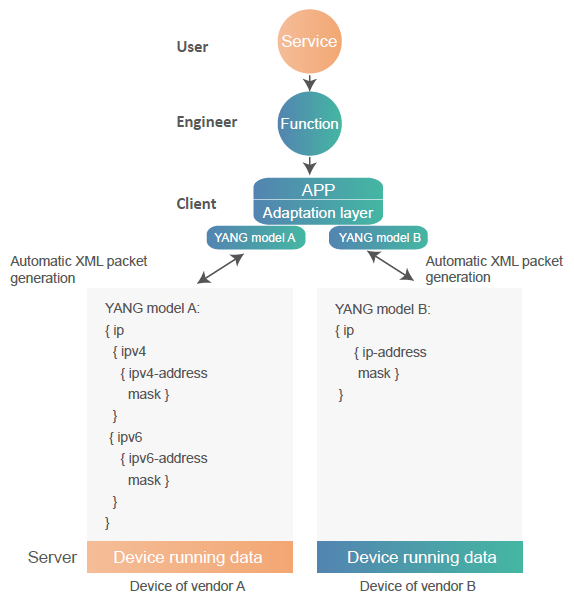


If functions are implemented in different ways on devices from different vendors:

In traditional CLI mode, engineers need to learn the differences between command lines of different vendors and manually map the command lines.



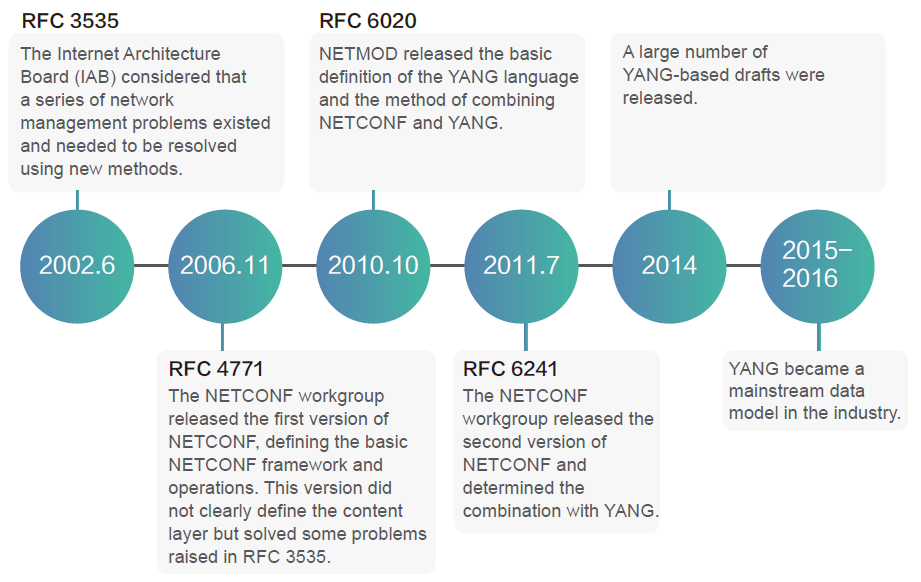
If functions are implemented based on YANG models, configuration engineers do not need to pay attention to the definitions of YANG models and the differences between the YANG models. The corresponding application automatically parses the YANG model data, shifting the focus of engineers from device and function differences to user requirements. The engineers can achieve automatic configuration only by operating the graphical application.



**NETCONF and YANG Development:**

YANG drives the development of NETCONF.

* Universal models are defined based on YANG, making a breakthrough in NETCONF implementation and laying the foundation for model unification between vendors.
* YANG-based network models are implemented on devices of different vendors, promoting NETCONF development.



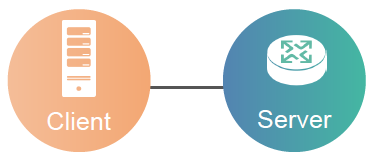
Example for the basic NETCONF session process:

**Scenario:**

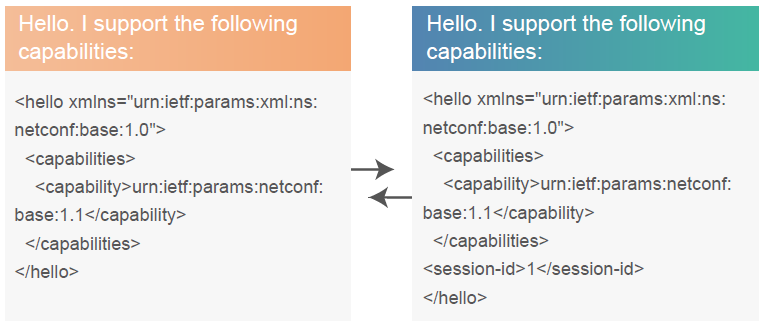
Use a NETCONF client to change the IP address of a device in two-phase validation mode.

**Prerequisites:**

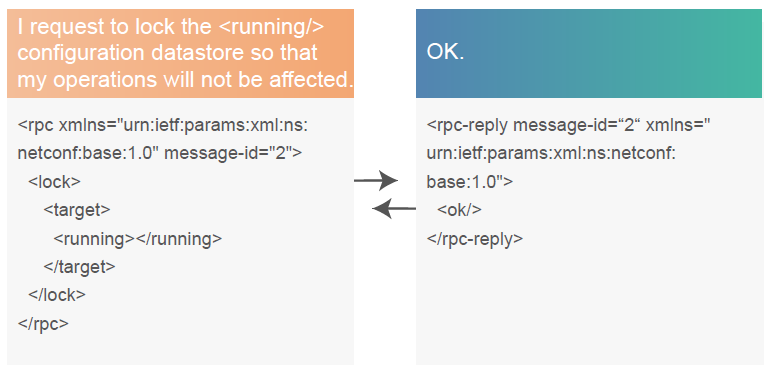
The SSH connection, authentication, and authorization are complete.



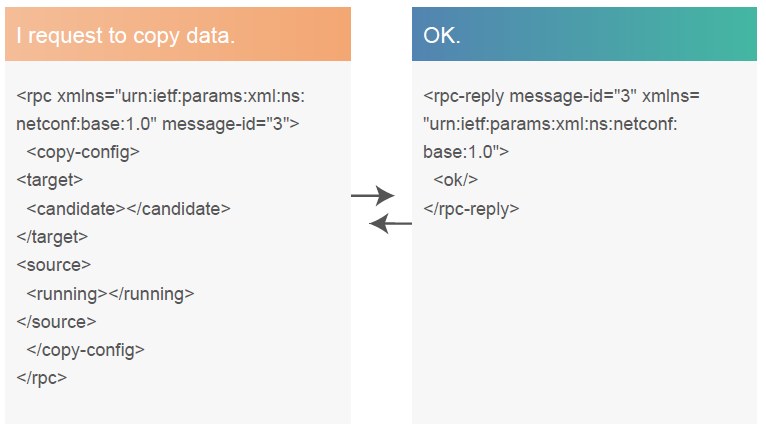
**Step 1: Establish a NETCONF session and advertise capabilities through <hello> messages.**



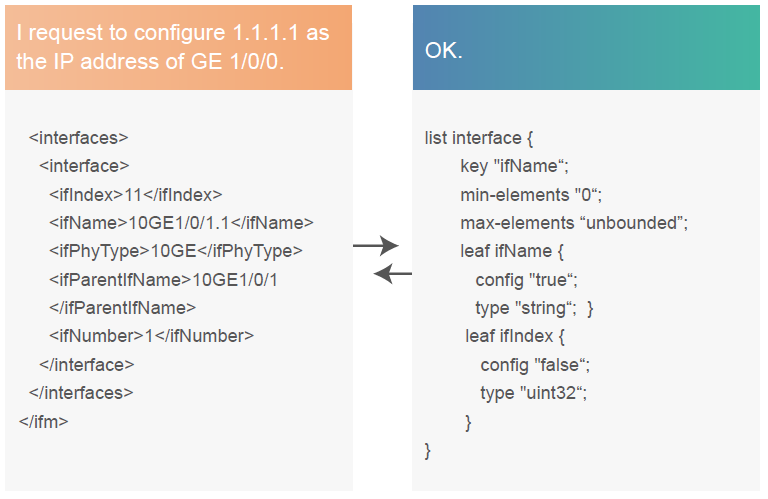
**Step 2: Lock the <running/> configuration datastore to avoid conflicts with other clients.**



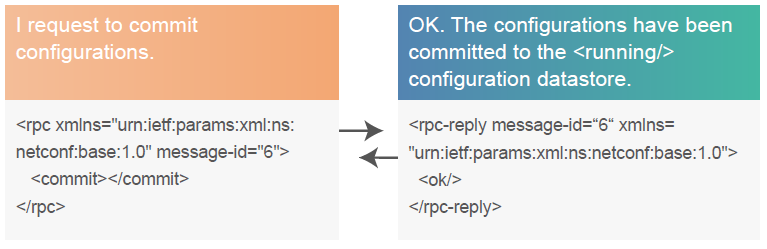
**Step 3: Copy the data in the <running/> configuration datastore to the <candidate/> configuration datastore to ensure that the configurations are the latest.**



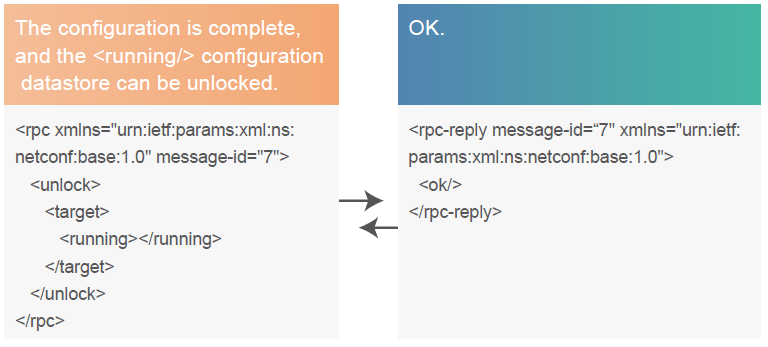
**Step 4: Edit configurations in the <candidate/> configuration datastore.**



**Step 5: Commit configurations in the <candidate/> configuration datastore to the <running/> configuration datastore.**



**Step 6: Unlock the <running/> configuration datastore.**



Follow-up procedure: Terminate the NETCONF session and tear down the SSH connection.