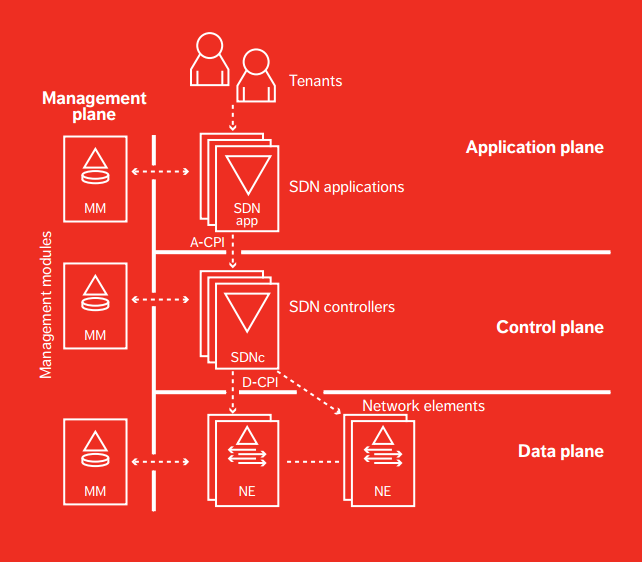
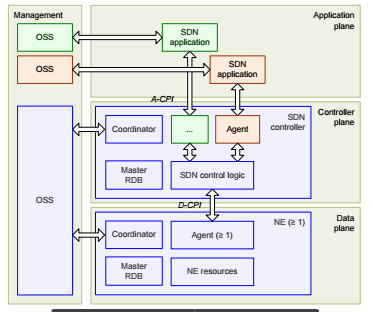
**Software Defined Networks**

**(Addressing security Concerns in SDN’s)**

**SDN Architecture:**

The SDN model proposed by the Architecture and Framework working group is composed of the application plane, the controller plane and the data plane. A fundamental concept of the SDN architecture is the separation of the controller plane from the data plane. Network switches become simple forwarding devices and the control logic is implemented in a logically centralized controller (in practical implementation, the control function is distributed for resilience). The SDN controller controls data plane resources via D-CPI (Data-controller plane interface). A-CPI (Application-controller plane interface) is used to realize communication between applications and controllers, and management functions are orchestrated through the management interface. With programmability and flexibility, new algorithms and applications can be implemented and verified efficiently. This configuration also supports higher-layer applications that deal with multi-tenant issues.

**What does controller stores and why:**

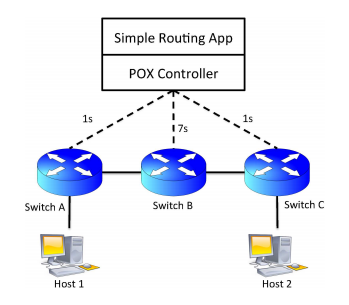
There are many invariants that have to be check in SDN’s like:

1. Race conditions and inconsistent ordering of rule updates in a traffic engineering application,
2. A security vulnerability in a multiple application interaction environment, and
3. A security issue due to rule expiration and rule shadowing.
4. How to deals with network outages that require rerouting?
5. How to debug software and hardware problems?

For all of the above mentioned problems, some data must be stored and shared on the SDN typically following data is stored on SDN controller:

1. Configurations.
2. Log files.
3. Control Logic.
4. Network Topology.

All of the data stored on the SDN controller is very critical and must be secured as many attacks can be done on SDN controller and this information can be revealed.

**Scenarios where Checking these Invariants is Necessary:**

1. **Scenario 1 : Incorrect Ordering of Updates:**

The scenario premise is that a developer has coded an application to install rules in switches in the following order: switch B and C are updated simultaneously (barrier messages are not used), then switch A is updated. The developer has not considered latency differences between the controller and switches, leading to a potential race condition. Specifically, rules may be installed at switches C and A, and data packets may start flowing before the appropriate rules are installed at switch B. This inconsistent state will lead to poor performance for traffic flows until switch B is updated, and could result in e.g., initial discontinuity in a VoIP call or high jitter at the start of a video stream.

1. **Scenario 2 : Bad Multi-app Interaction:**

SDN controllers typically permit multiple, logically separate control programs to be run simultaneously, each of which can receive network events and modify switch state. It is well known that inconsistent network state can arise when SDN programs are not composed correctly. We call this problem Multi-app Interaction Inconsistency (MAII).

1. **Scenario 3 : Unexpected Rule Expiration:**

It is well known that unexpected interactions can occur when wild-carded rules overlap, or specific micro-flow rules are shadowed by wildcard rules. If the actions associated with the rules differ, unexpected behavior can occur if the more specific rule is removed while traffic is still flowing.

**Security Concerns in SDN’s:**

1. **Vulnerability of Central Control:**

* SDN Applications and Controller have complete control of the network

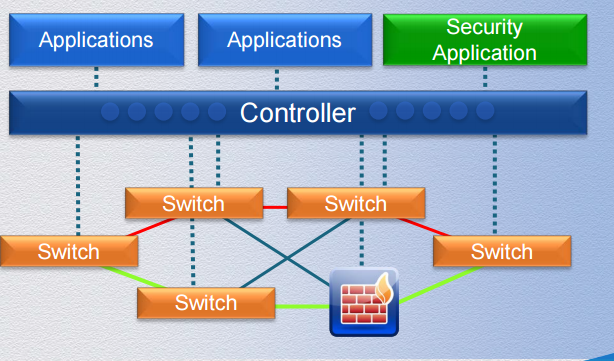
• Controllers/Applications are built on general purpose computing platforms, we all know all about the vulnerabilities of these platforms

* If Controller or Application is compromised, the whole Network is compromised.

1. **Possible Attacks:**

* Route flows around security devices
* Controller subverts new flows
* Send traffic to compromised nodes
* “Man in the Middle” attacks
* Modify content
* Insert malware
* Monitor traffic
* Subvert DNS responses

1. **Programmability:**
   * Traffic and resource isolation
   * Trust between third party applications and the controller
   * Interface Security protection on A-CPI and I-CPI
2. **Cross Domain Connection:**
   * An additional requirement of SDN implementation requires that infrastructure of different domains can be connected.
   * The mechanisms to establish trust relationships, to determine authorization level in order to prevent abuse and secure channel setup should all be considered.

**Current Solutions vs Our Solution:**

Generally, one application or the fire-wall is introduced in the system to ensure the security and the policies all over the networks. Also another solution is devised, which is to divide the controller into multiple controllers which can work together to form the network. Now these solutions have great overheads and reduce the efficiency of the network. Our solution is to find an encryptic algorithm and store the data on SDN in that way so that the information is secured.