Week 12

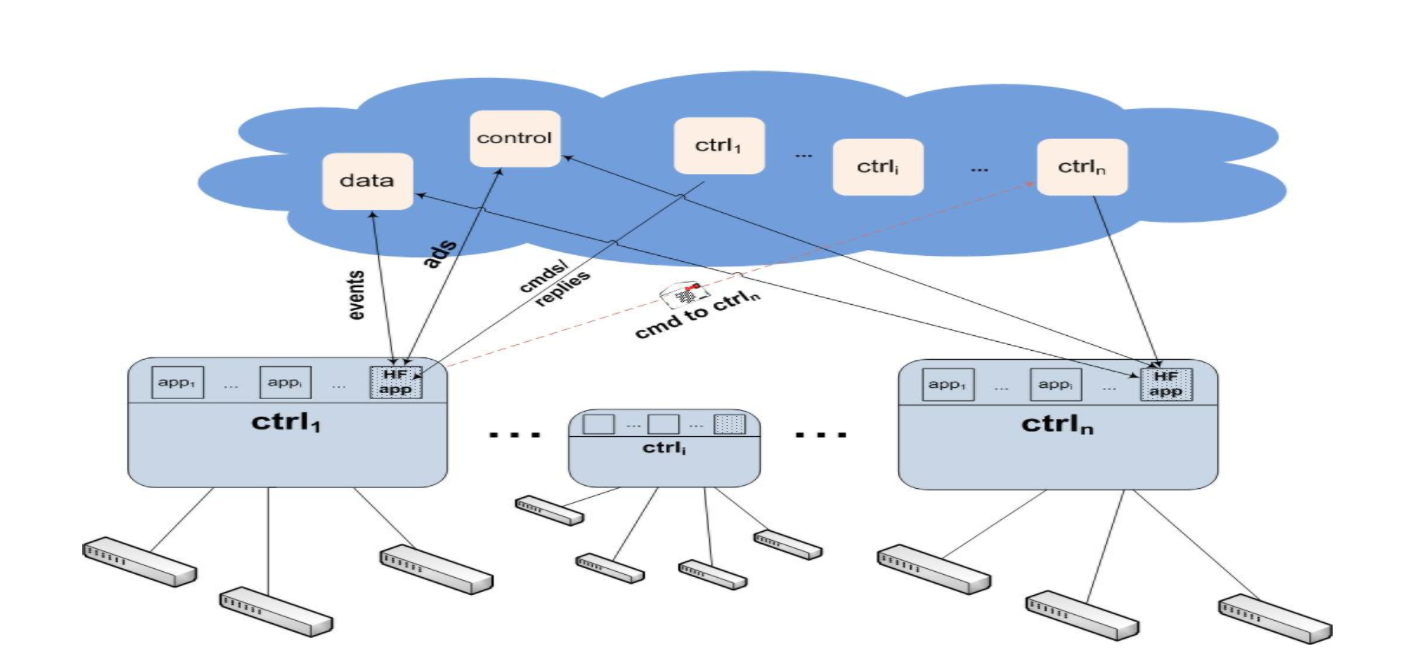


Figure 1: High-level overview of HyperFlow. Each controller runs NOX with the HyperFlow application atop, subscribes to the control, data, and its own channel in the publish/subscribe system (depicted with a cloud). Events are published to the data channel and periodic controller advertisements are sent to the control channel. Controllers directly publish the commands targeted to a controller to its channel. Replies to the commands are published in the source controller.

To propagate controller events to others, HyperFlow uses publish/subscribe messaging paradigm. The publish/subscribe system that HyperFlow uses must pro- vide persistent storage of published events (to provide guaranteed event delivery), keep the ordering of events published by the same controller, and be resilient against network partitioning (i.e., each partition must continue its operation independently and upon reconnection, partitions must synchronize). The publish/subscribe sys- tem should also minimize the cross-site2 traffic required to propagate events, i.e., controllers in a site should get most of the updates of other sites from nearby controllers to avoid congesting the cross-region links. Finally, the system should enforce access control to ensure authorized access.

WheelFS is a distributed file system designed to offer flexible wide-area storage for distributed applications.

It gives the applications control over consistency, durability, and data placement according to their requirements via semantic cues. These cues can be directly embedded in the pathnames to change the behavior of the file system. In WheelFS, we represent channels with directories and messages with files. To implement notification upon message arrival (i.e., new files in the watched directories) HyperFlow controller application periodically polls the watched directories to detect changes.