Report for November

Policy based Docker Container Migration

Goal: Build a flexible Migration System based on different policy, providing multiple Network functionality from different aspects to the system user.

Policy defined:

- **Random**: Migration source host and destination host are selected by the system randomly, regardless traffic situation.
- **Bandwidth**: Migration decision made according to the network traffic situation, in particular, according to the path bandwidth, system select the migration destination host which has the maximum bandwidth available.
- **Shortest Path**: Migration decision made according to the network traffic situation, in particular according to the shortest path,system select the migration destination host which has the fast path.

Policy implementation:

Migration Source Host selection(Algorithm 1):

1) Access Database checking from AggregateStatistics table the switches which both aggregate and aggregate difference greater than the threshold. Then get the heavy switch set,then use customized mode select one which will be potentially heavy in the future.**-------> heavy switch selected**
2) Access Database checking from BandwidthStatistics table the host which has the maximum Bandwidth consumption(suppose that every host connected with the same switch has the same maximum bandwidth.**--------> migration source host selected**

Random: One random number generator is used, source host is selected from the hosts set which are running the Docker container.

Bandwidth: Migration Source Host selection(Algorithm 1)
Migration Destination Host selection(Alorithm2):
Send REST API requests to floodlight controller to get the paths to each free switch,then use the function Max{mix(Bandwidth)} to select the switch which has the maximum available Path bandwidth.Then select the host which has the maximum available bandwidth.------>migration destination host selected

Shortest Path: Migration Source Host selection(Algorithm 1)
Migration Destination Host selection(Alorithm2):
Send REST API requests to floodlight to get the fast path to each free switch,and then select one has the fastest path with least latency as the target destination switch. From the target destination switch select the first free host. ------->migration destination host selected

Policy analysis:

Random:

Pros: no computation, less CPU usage, can be used to defence server attack
no interact with SDN controller

Cons: high possibility of making bad decision
--select light host as migration source

--select heavy host as migration host

Migration time is not predictable

Bandwidth:

Pros: more compatible with the network traffic situation

Guarantee server have better performance after migration.

Cons: More CPU usage(executing algorithm to make migration decision)

Migration time is not predictable

Shortest Path:

Pros: more compatible with the network traffic situation

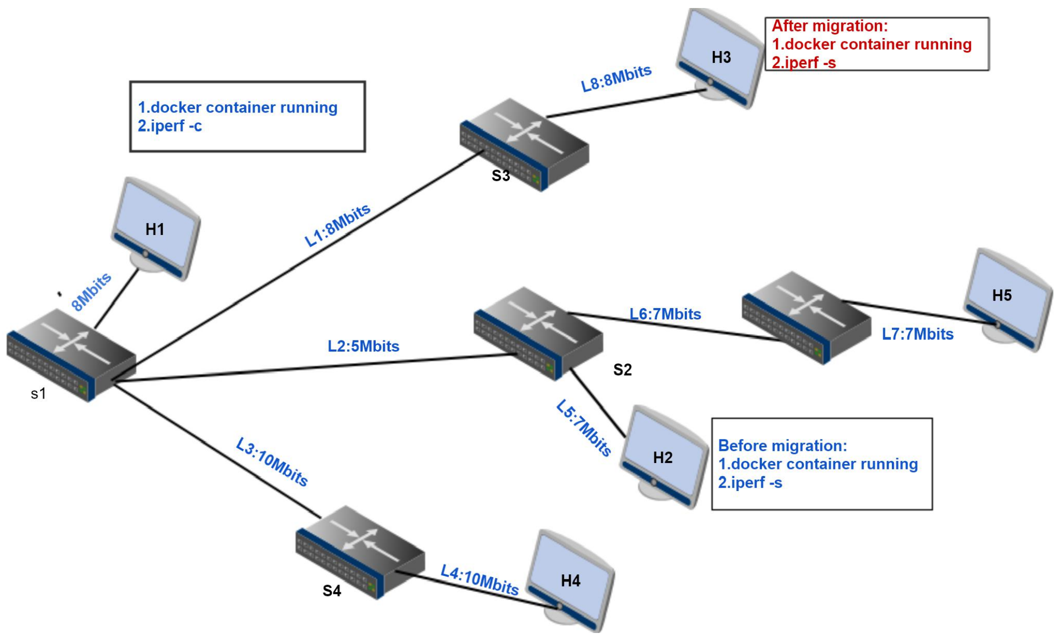
Migration time is the shortest

Cons: More CPU usage(executing algorithm to make migration decision)

Server performance are not guaranteed

Experiment topology use case:

For random and bandwidth policy:



How to do the experiment:

Bandwidth policy:

--H1 is as iperf client, H2 is as iperf Server.

--L2 is the bottleneck link

Expect result:

System will choose H5 as the migration host instead of H3 and H4, because the H5 has the biggest path available bandwidth.

Random policy:

Initial condition:

--H1 is as iperf client

--H2 is as iperf server

Expect result:

System will randomly choose one migration destination host in each time run

Note:Experiment result comparison will show in next report.