Intro

As part of our simulation framework we will allow an test script to be run on our server. The test script should execute the various functions on the server without the user having the manipulate them using our GUI. Things like manipulating the RDT, moving devices to and from networks, merging and dividing networks is automatically handled and performed by our test script

Plan

We plan to have the user upload a JSON file specifying the general parameters for a test script they want to run. We decided not to have the user write javascript which we would execute because it would difficult on the part of the simulation manager to have to write the various functions and calls and ensure they were compatible with our framework. Although our framework has a public API for how to perform operations such as moving devices to networks and merging partitions, it would be difficult for a new simulation manager to first learn the public API before deploying a test script. Just making them upload a JSON file which only specifies the general parameters, we can ensure that test scripts are handled in a streamlined, easy to implement and consistent manner across all simulation environments.

The test script itself uses our public API to manipulate the various elements in the simulation it is uploaded to.

Remarks and Justification

The composition of the JSON file is set up to be easy to understand and replicate for any simulation manager. A general test script is of the form:

```
{
    "name": "firstTest",
    "description": "The first automatic test script for our framework",
    "parameters": {
        "rdts" : [{"name": "incRDT", "method" : "inc"} , {"name" :
        "decRDT", "method" : "dec"}],
        "device_mobility" : "high",
        "operations" : 25
    }
}
```

The name and description are only used for identification purposes in terms of logging the events in the simulation. The parameters is what is truly important and without that component the test script would not run. The first is identifying the rdts to be manipulated. The name of the rdt and the method to be used must be specified. The rdts in this list is checked when the user uploads the test script to ensure they are available in the simulation.

The device_mobility specifies to what extent they want devices to be able to move around in the simulation. A high mobility means that while the test script is running, devices will be constant moving in and out of networks and partitions divided and merged.

The operations is the number of operation to be performed on the RDT. The script manager in our framework is smart to ensure that while it will try to perform as many operations specified.

The frequency of the operations will be function of the device mobility. That way it cannot manipulate the RDT more than it moves devices around. We think it simply makes more sense in terms of evenly dividing the operations when the script is running. The operations is an upper bound rather than a set figure that the test script manager uses.

The test script manager in our framework will ensure that it takes account the device mobility in determining how many RDT operations to perform.

Once the test script is uploaded, the parameters are extracted and it is immediately sent to the test script manager to run. After this the user can look in the simulation history to see the new state of the simulation and go back and view how the state and topology changes as the test script was manipulated.