Hyperledger Umbra: Simulating Hyperledger Blockchains using Mininet

Raphael Vicente Rosa
Mentor: David Huseby

Hyperledger Umbra

- Storyline
  - Umbra 2018 - Shadow Simulation Platform
- Proposal [1]
  - Use Mininet for Emulation

[1] https://wiki.hyperledger.org/display/INTERN/Hyperledger+Umbra%3A+Simulating+Hyperledger+Blockchains+using+Mininet

Why Blockchain Networks with Mininet?

- Issues with Shadow
  - Not (enough) kernel libraries available
  - i.e., patching environment for each project -> hard for fast prototyping
- Mininet
  - Created for simple prototypes of Software Defined Networks concepts
  - Can emulate hosts and programmable switches in a laptop
    - Easy/Fast prototype/test cycles
  - Scale to hundreds of nodes -> Included multiples servers (i.e., thousands of nodes)
  - Extract granular networking features
  - Well documented
  - Active mailing list

Why Blockchain Networks with Mininet?

Your project fully reproducible/comparable!

i.e., chaincodes, transactions, consensus, footprints, traffic analysis, etc...

Debugging blockchain projects in a laptop!

A focus on research oriented test cases (e.g., investigating consensus algorithms)

Software Defined Networking (SDN)

- Started in 2008 - Stanford
- Nowadays, widely adopted (standards, open source projects, etc)
- In essence, splits network control and data planes
- i.e., programmable networks!
- Open interfaces for network controllers

Mininet: Quick Intro

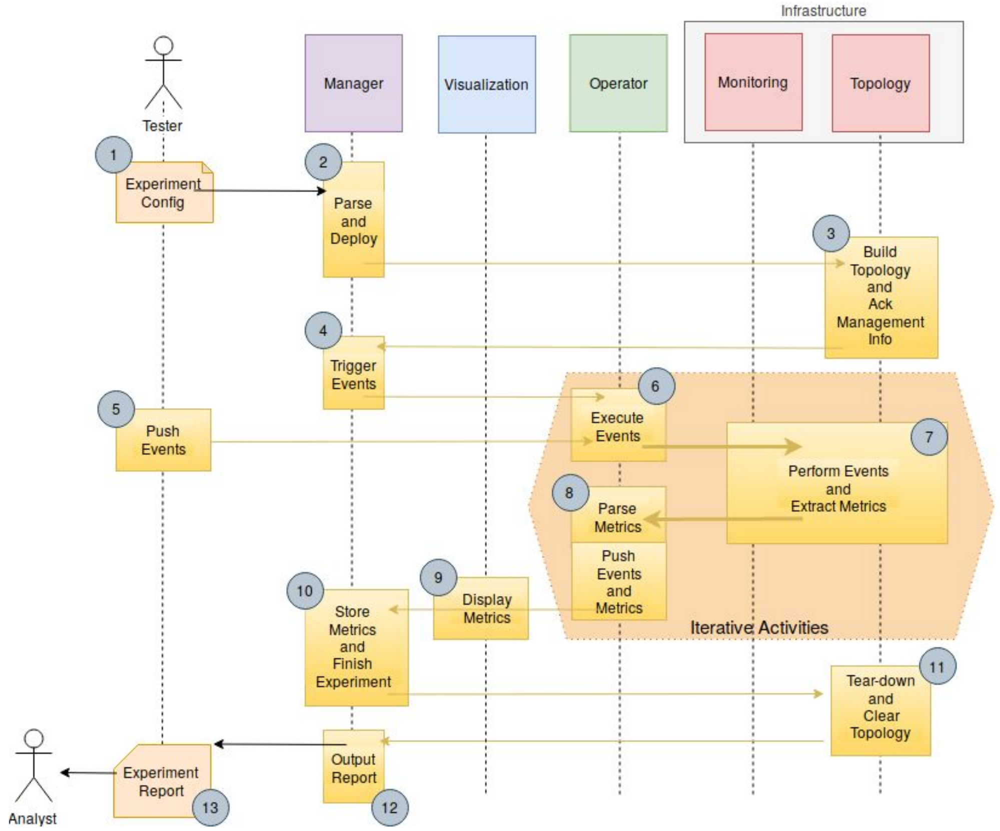
- http://mininet.org/
- Create nodes and switches -> extensible and flexible network
- Create dynamic links
- Set nodes/links resources (e.g., bandwidth, latency, cpu, memory, etc)
- Plug and play networks
- Easy to start coding networks (http://mininet.org/walkthrough/)
- Extended:
    - Maxinet
    - Containernet
    - Mininet-wifi/[IoT]

Using Containernet to Emulate Blockchain Projects

- Ideas:
  - Plug and play blockchain projects
  - Unit test cases in a laptop
  - Inherits all the abstractions from SDNs
  - Grabs containers as host entities
    - i.e., any available version of HL projects in Docker images will be just fine

Hyperledger Umbra - Internship 2019

- Written in python 3 and Yaml config files
- Test deployed in one (or more) servers
- Test descriptor and report -> reproducible and comparable
- Event oriented:
    - Nodes/Links lifecycle
- Architecture
    - Microservices with REST interfaces
- Views:
    - Use Case: Tester/Analyst
    - Logical: Modules/Classes



Process View: Big Picture of Test Workflow

Activities

1. Build reference architecture (Logical View and Process View)
2. Evaluate a Hyperledger project (e.g., Fabric) with reference architecture
3. Build Stimulus
4. Build Monitoring
5. Build Dynamics
6. Build Analysis
7. Scale to cluster of servers
8. Extend architecture to other Hyperledger project(s)
9. Document the project

Expected Outcomes

- Proof of concept configurations, scripts, and documentation for setting up a Mininet network that runs one of the Hyperledger blockchain frameworks under simulation.
  - A proposal for a blockchain experiment that can be executed on the Mininet simulation setup.
- Stretch goals:
  - Coordinate with the maintainers for a Hyperledger framework project to execute a useful experiment (e.g. running at different scales 10, 100, 1000 nodes) and providing an analysis report and/or blog post describing the findings.
  - Investigate packet data on how blockchain projects realize consensus

Future Plans

The Hyperledger Umbra Lab already exists.

Once we achieve the ability to run Hyperledger blockchains under simulation, future work will be around designing and conducting empirical computer science research experiments to help direct improvements in design and architecture of future versions of the Hyperledger blockchain platforms.

Joining Umbra

- Github - https://github.com/hyperledger-labs/umbra
- Mailing list - #umbra in labs@lists.hyperledger.org

Demo

Reproducing Hyperledger Fabric

- Run 2 orgs with 2 peers each, and another org with orderer
- All files (topology, events, crypto-keys, genesis, etc) generated from a single build_configs.py script
  - Based on module umbra-configs
- Abstracts each peer as a docker container
- Joins peer in orgs by a single network (i.e., single virtual switch)

Demo

- Containers deployed according to topology built
- Events called upon schedules
- In summary,
  - Deploys peers/orderer of orgs
  - Creates channel
  - Peers to join channel
  - Info about channel, network, etc
  - Install, Instantiate, Invoke chaincodes in orgs
  - Query chaincode in orgs
  - Clean the environment

Beyond the Basics

- Requires all the setup of cryptogen and configtxgen config files
  - Built upon topology definition
- fabric-python-sdk was crucial to perform all the events
- Docker and Open vSwtiches can enable sFlow metrics to be collected in a single database
  - Measures host and networking metrics during execution
- Still, optimization can be applied for config files (transparent for users)