Emulating an Infrastructure with EASE

ARISTA

Arup R. Roy

Shihabur R. Chowdhury, Md. Faizul Bari, Reaz Ahmed, and Raouf Boutaba
EASE: Emulation as a Service

- A multi-tenant, distributed and virtualized shared emulation platform with built-in SDN
- EASE can emulate infrastructures consisting of compute, network and storage resources
- Evolution of our prior work DOT* to a cloud-based service.

---

Why EASE?

- Quickly deployable SDN emulators (e.g., Mininet) cannot scale to network size and traffic volume*.
- Large-scale SDN emulators (e.g., DOT*, Maxinet**) require up-front investment and time consuming setup.
- Shared testbeds (e.g., EmuLab, GENI etc.) do not provide all the desired features

---

** P. Wette et al. “Maxinet: Distributed Emulation for Software-Defined Networks”, IFIP Networking 201
Desired Features of a Shared Testbed

- Performance isolation between testbed users
- Resource guarantee to support reproducible emulation
- Fault-tolerance for seamlessly resuming emulations
- Maximize underlying infrastructure utilization
  - To support more users
A Case Study: EmuLab

- Emulab is a shared emulation platform for network emulations.
- We deployed Internet2 (12 nodes, 15 links) topology on Emulab.
- We measured link utilization of a selected link.
- We varied traffic on links other than the one we selected for measurement.
A Case Study: Emulab (contd...)

- Emulab provides *isolation between the users* by hard-partitioning resources.
  - This reduces the *number of simultaneous users*
- No Resource Guarantee
  - Limits *experiment reproducibility*
- No Fault-tolerance
EASE: Challenges

- How to guarantee resources (CPU, network bandwidth, storage) for reproducible experiments while maximizing the number of simultaneous users?
  - We use time dilation

- How to implement time dilation across all resources?

- How to provide transparency to the users, i.e., hide distributed nature of deployment?

- How to ensure fault-tolerance for seamless execution of emulations?
Challenge-1: Resource Guarantee while Maximizing No. of Users

- Solution: Time Dilation
- Time dilation slows down the progression of time
- Time dilation can stretch the perceived limits of the infrastructure
  - A link with 50Mbps remaining capacity can appear as 100Mbps if the speed of time progression is halved
Challenge-1: Resource Guarantee while Maximizing No. of Users (contd...)

- Heuristic Algorithm for emulation provisioning
  - Binary search on TDF to determine the minimum TDF that yields a feasible embedding.
  - Emulation request is partitioned in clusters.
    - Cluster of virtual switches are deployed in a single VM with the required resources.
  - Each cluster is placed on a different machine.
  - First-fit algorithm for embedding.
Challenge-1: Resource Guarantee while Maximizing No. of Users (contd...)
Challenge 2: Implementation of Time Dilation.

- Modify timer management in each subsystem
  - Compute, Network, Memory
- We modified timer management in KVM hypervisor for Intel processors
  - Intercept `rdtsc` instruction that reads time stamp counter register from CPU
  - Modify time-stamp computation to slow down time
Challenge 2: Implementation of Time Dilation (contd...)

- Timer management is architecture specific.
- Non-uniform methods to dilate all resources.
  - We place the switches inside dilated VMs to dilate switching.
  - Still open problem to dilate memory access time
- Time dilation synchronization across multiple machines.
  - All resources deployed on different machines should be identically dilated.
Preliminary Performance Evaluation

![Graph showing throughput vs. background traffic with Emulab and EASE lines, emphasizing a section requiring further investigation.](image-url)
Conclusion

- EASE is a proposal for a distributed testbed that provides emulation as a service to the users.
- Full-fledged implementation of EASE is yet to be done.
- We leverage time dilation to maximize the number of users admitted in the system.
  - Some challenges pertaining to time dilation are still open.
Questions?