Building an Internet Emulator for Cybersecurity Education

Wenliang (Kevin) Du

Syracuse University

Outline

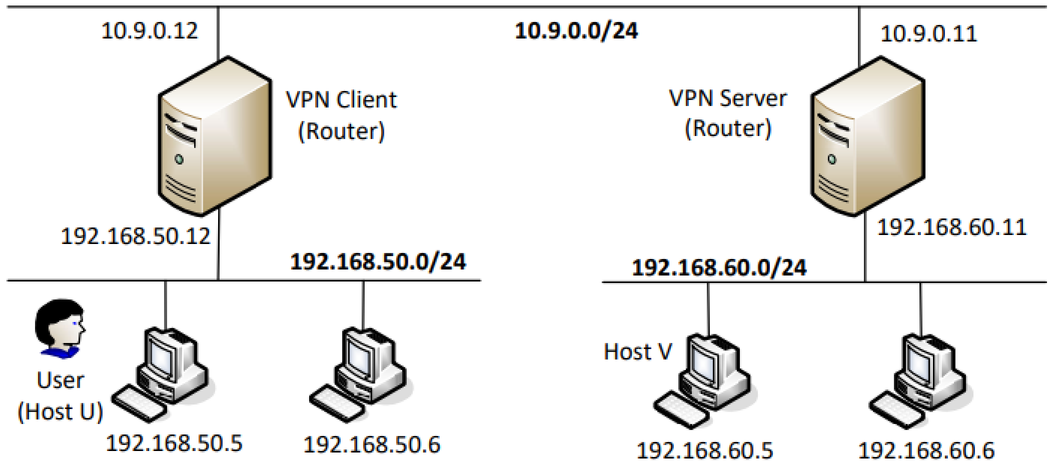
- Motivation for this project
- The design ideas
- The emulator details
- Applications: Labs
  - BGP attack lab
  - Morris worm attack lab
- Demos

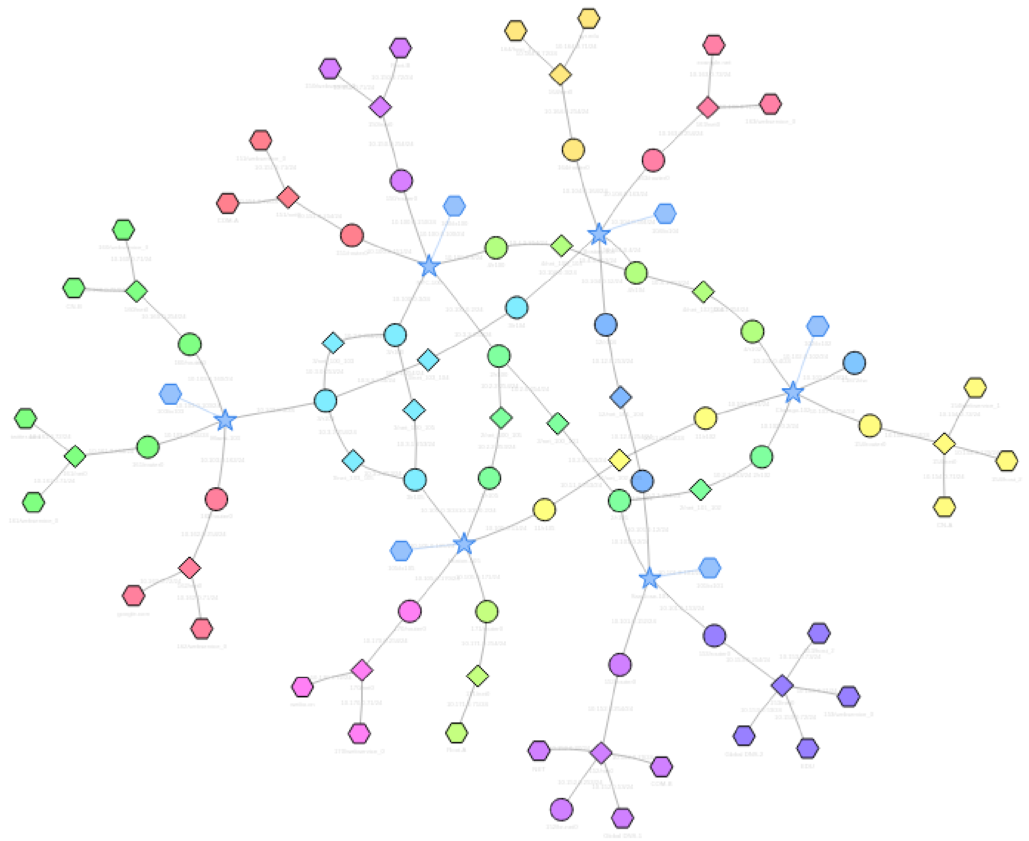
Motivation

https://seedsecuritylabs.org/

Adopted by 1000 institutes

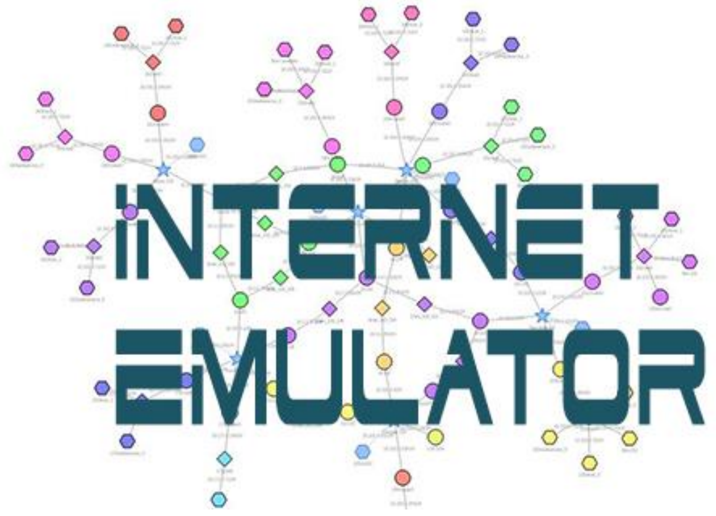
Limitations





SEED Labs 1.0: Using VMs
SEED Labs 2.0: Using docker containers

The Open-Source Project



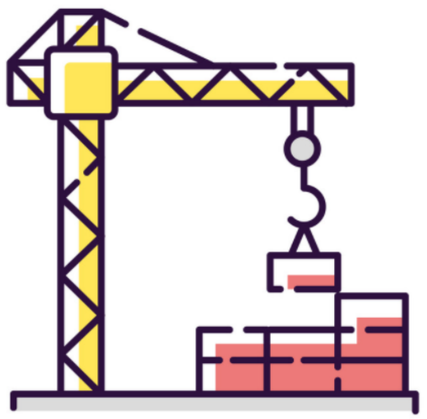
- **Founders**
  - Kevin Du
  - Honghao Zeng (MS student)

- **History**
  - 2018 – 2020: Investigation & Design
  - August 2020: Implementation
  - July 2021: First release

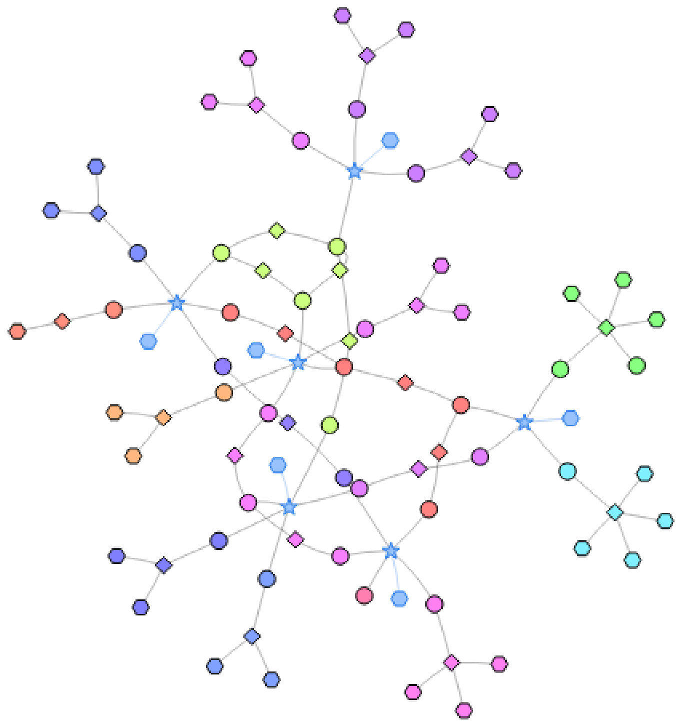
https://github.com/seed-labs/seed-emulator

The Most Important Design Decision

Building Emulation



Conducting Emulation



Existing Work

- CORE: Common Open Research Emulator
  - Based on Linux namespace
- GNS-3: Graphical Network Simulator-3
  - Focus on network emulation, not Internet
  - Good at emulating vendor-specific network devices
- NS-3
  - A simulator, not an emulator
  - Good at simulating network technologies
  - Not transparent to applications

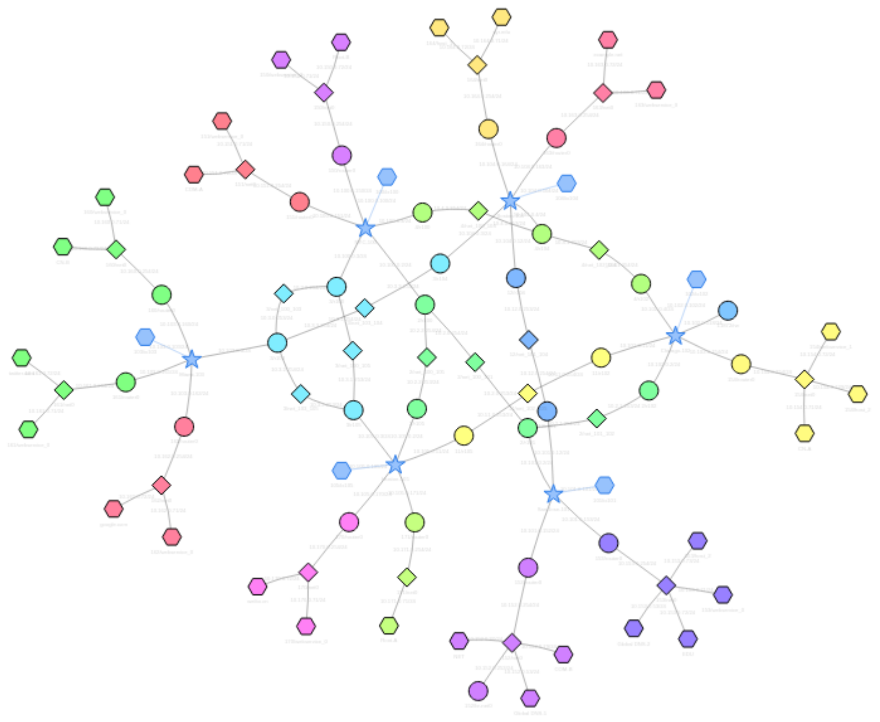
Our Approach

- Using Docker for emulation
- Our job: compose the emulation
  - Constructing docker files

Building Emulation

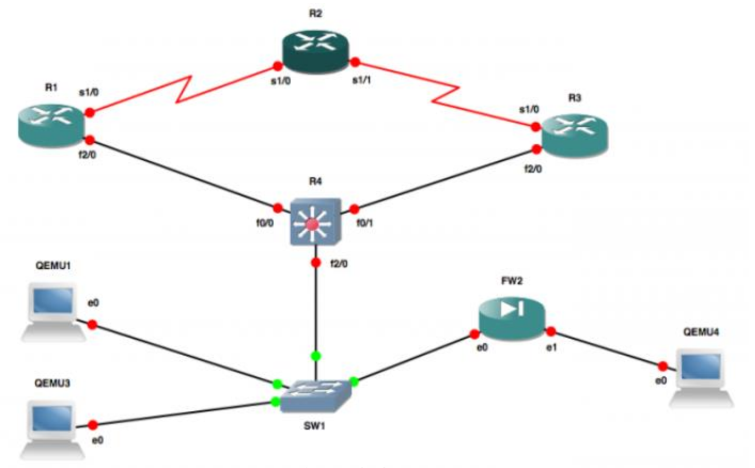
```
seed@VM:~/.../output$ ls
docker-compose.yml        hnode_160_host_1          rnode_12_r101        rnode_2_r101
dummies                   hnode_160_webservice_0    rnode_12_r104        rnode_2_r102
hnode_150_host_1          hnode_161_host_1          rnode_150_router0    rnode_2_r105
hnode_150_webservice_0    hnode_161_webservice_0    rnode_151_router0    rnode_3_r100
hnode_151_host_1          hnode_162_host_1          rnode_152_br-net0    rnode_3_r103
hnode_151_webservice_0    hnode_162_webservice_0    rnode_152_router0    rnode_3_r104
hnode_152_host_0          hnode_163_host_1          rnode_153_router0    rnode_3_r105
hnode_152_host_1          hnode_163_webservice_0    rnode_154_router0    rnode_4_r100
hnode_152_local-dns-1     hnode_164_host_0          rnode_160_router0    rnode_4_r102
hnode_153_host_1          hnode_164_host_1          rnode_161_router0    rnode_4_r104
hnode_153_host_2          hnode_170_host_1          rnode_162_router0    rs_ix_ix100
hnode_153_local-dns-2     hnode_170_webservice_0    rnode_163_router0    rs_ix_ix101
hnode_153_webservice_0    hnode_171_host_0          rnode_164_router0    rs_ix_ix102
hnode_154_host_0          rnode_11872_rw            rnode_170_router0    rs_ix_ix103
hnode_154_host_2          rnode_11_r102             rnode_171_router0    rs_ix_ix104
hnode_154_webservice_1    rnode_11_r105             rnode_2_r100         rs_ix_ix105
```

```
 429 Nov 16 20:13 082b96ec819c95ae773daebde675ef80
1072 Nov 16 20:13 17ac2d812a99a91e7f747e1defb72a29
2578 Nov 16 20:13 2b0ae038330eccd43095538618caee7d
 242 Nov 16 20:13 d18858afc6bb66ec3a19d872077acfd2
1110 Nov 16 20:13 d3d51fdf7f4bad30dc5db560a01ce629
 911 Nov 16 20:13 Dockerfile
  58 Nov 16 20:13 e01e36443f9f72c6204189260d0bd276
```



Different Approaches

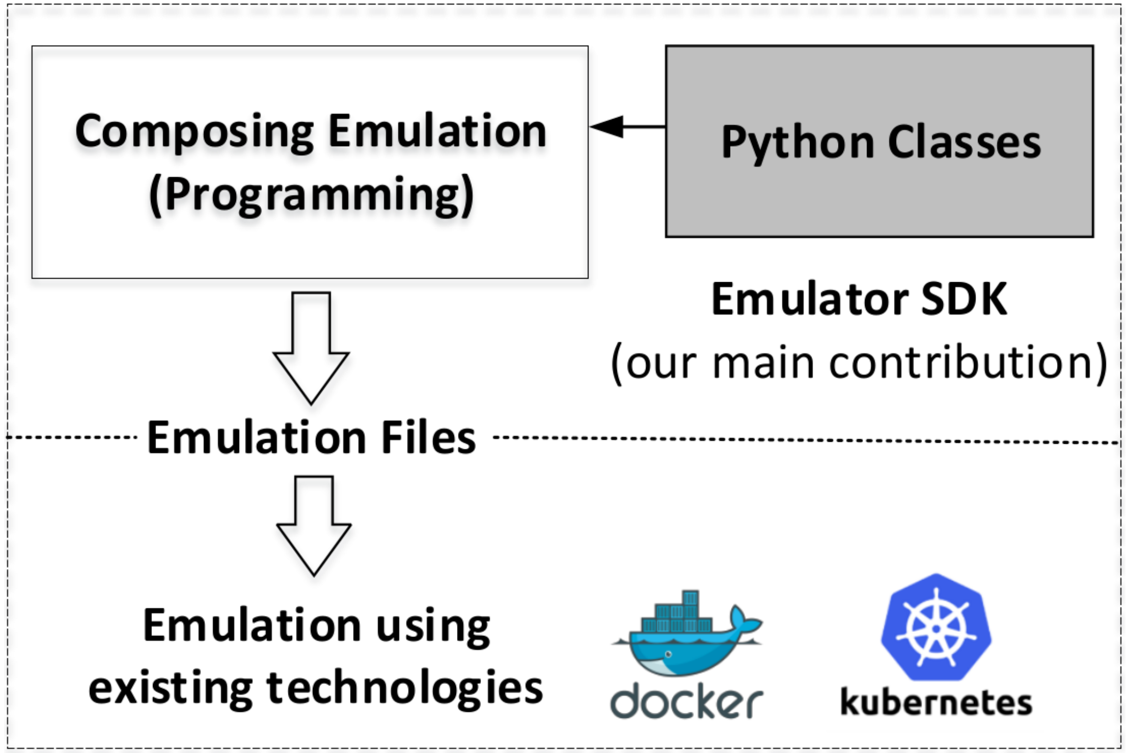
- GUI Approach
- Configuration Approach (JSON, YAML)
- Programming Approach



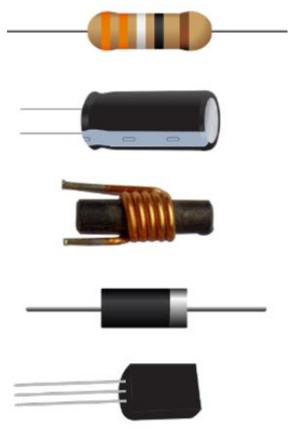
GNS3

What they have in common: **language**

Our Design

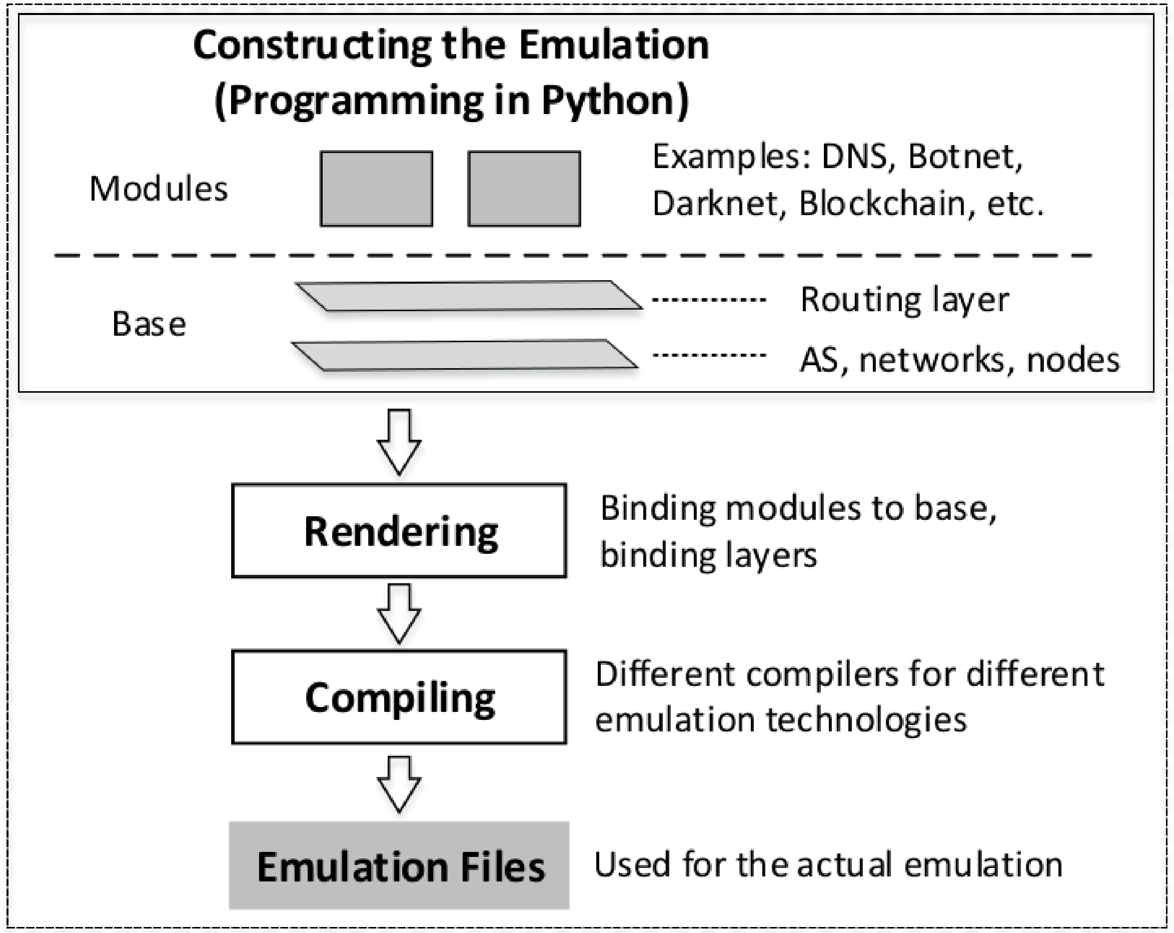


Primitives (Classes)

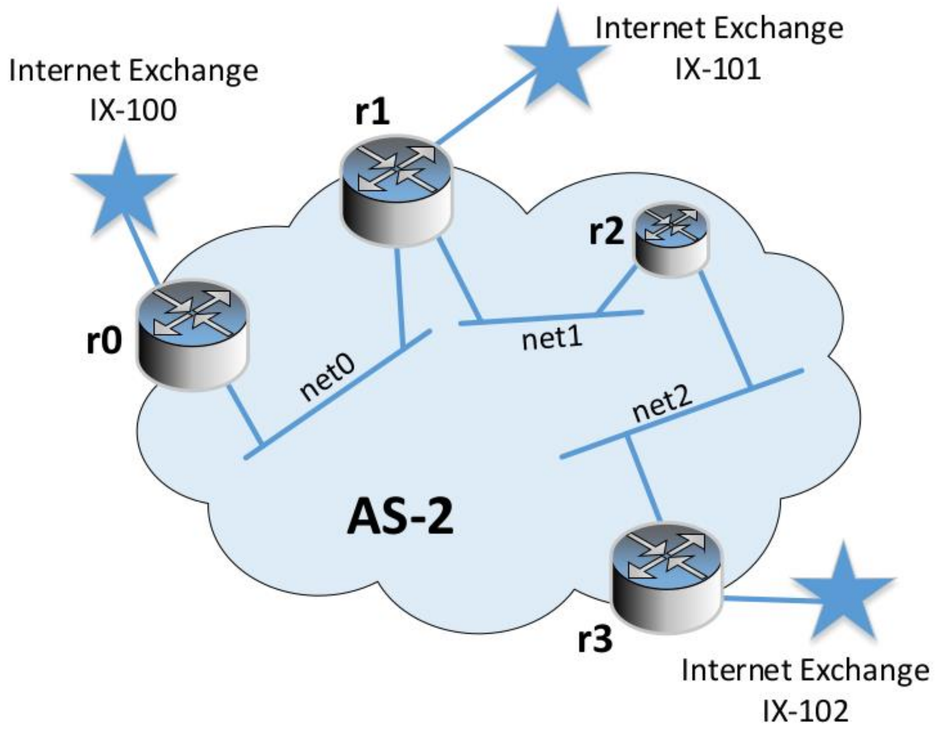


- Autonomous System
- Internet Exchange
- Network
- Router, BGP speaker
- Host
- Service
- etc.

The Architecture

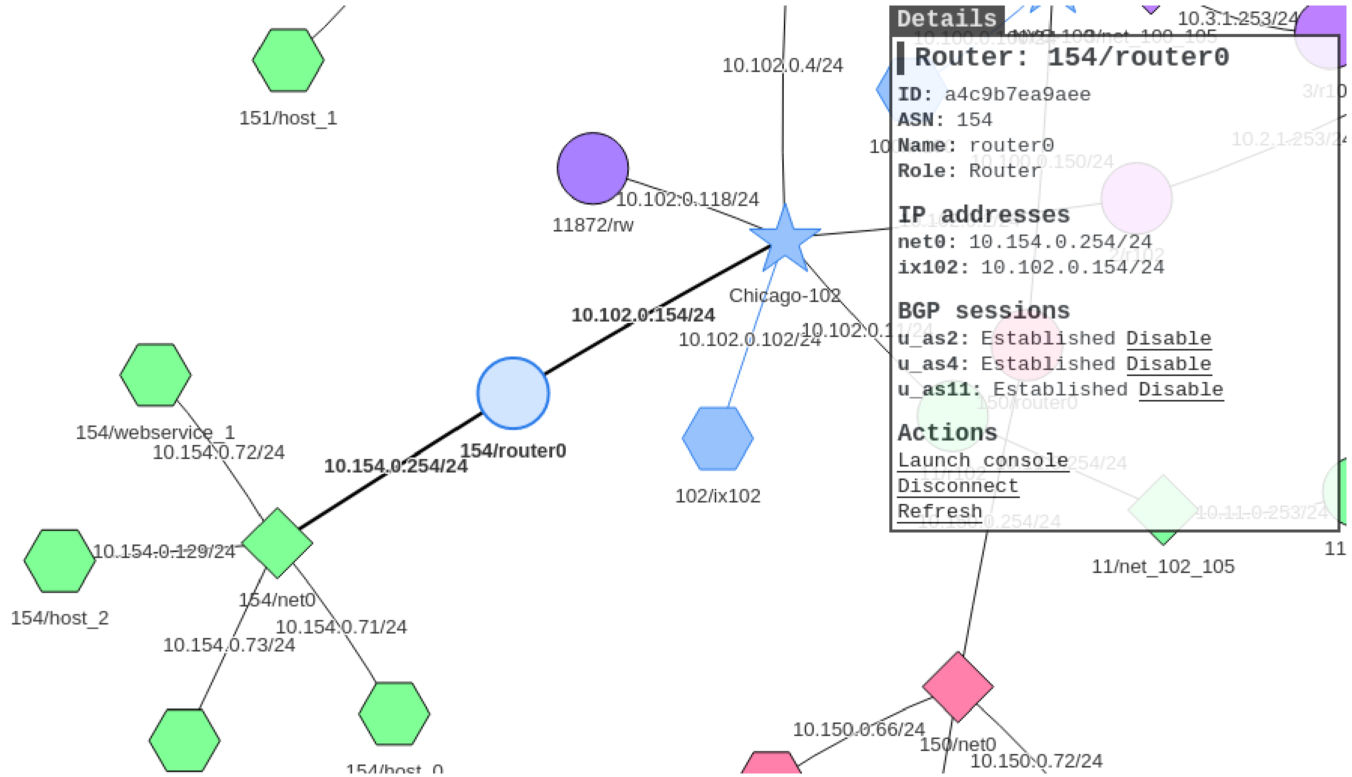

Constructing the Emulation
(Programming in Python)
Modules
Examples: DNS, Botnet, Darknet, Blockchain, etc.
Base
Routing layer
AS, networks, nodes
Rendering
Binding modules to base, binding layers
Compiling
Different compilers for different emulation technologies
Emulation Files
Used for the actual emulation

Example: Create a Transit AS



```
# Create the autonomous system (asn = 2)
as2 = base.createAutonomousSystem(2)

# Create 3 internal networks
as2.createNetwork('net0')
as2.createNetwork('net1')
as2.createNetwork('net2')


# Create 4 routers
as2.createRouter('r0').joinNetwork('ix100')
                      .joinNetwork('net0')
as2.createRouter('r1').joinNetwork('net0')
                      .joinNetwork('ix101')
                      .joinNetwork('net1')
as2.createRouter('r2').joinNetwork('net1')
                      .joinNetwork('net2')
as2.createRouter('r3').joinNetwork('net2')
                      .joinNetwork('ix102')
```

Example: BGP Peering


Details

Router: 154/router0

ID: a4c9b7ea9aee
ASN: 154
Name: router0
Role: Router

IP addresses
net0: 10.154.0.254/24
ix102: 10.102.0.154/24

BGP sessions
u_as2: Established Disable
u_as4: Established Disable
u_as11: Established Disable

Actions
Launch console
Disconnect
Refresh

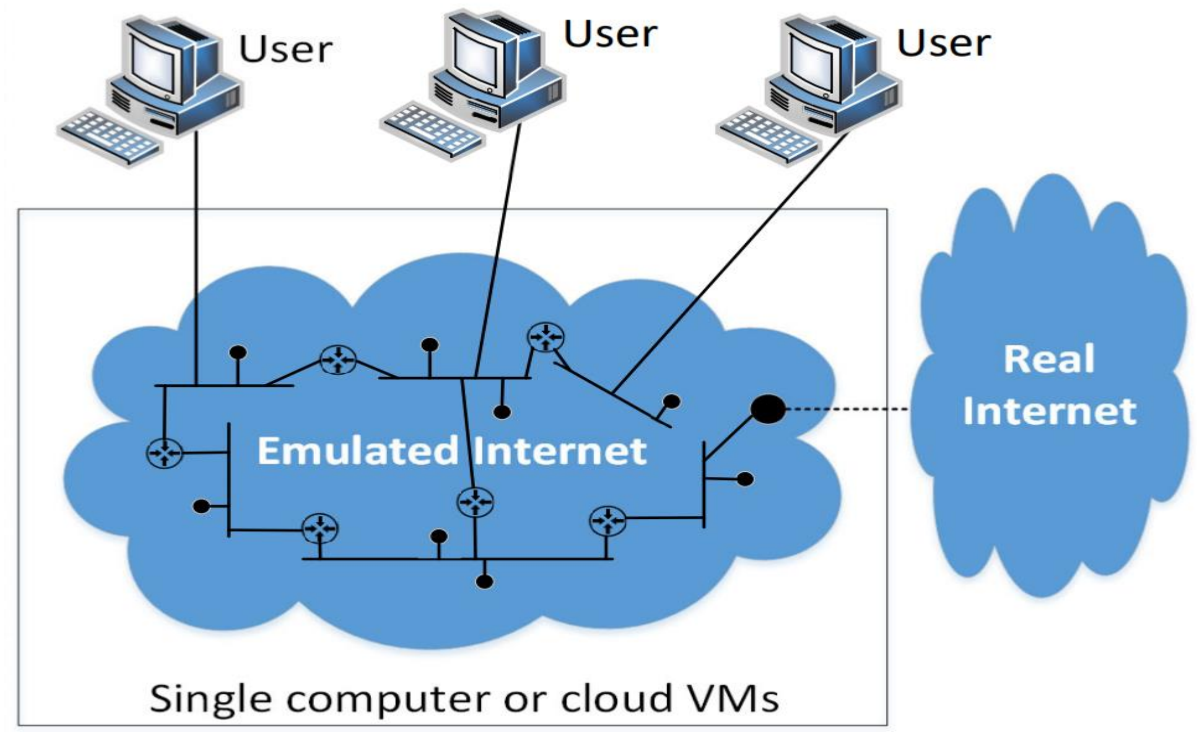
```
ebgp.addPrivatePeerings(102, [2, 4],  [11, 154], PeerRelationship.Provider)
ebgp.addPrivatePeerings(102, [11], [154, 11872], PeerRelationship.Provider)
```

Customizing Nodes

```
# Get an instance of the host from AS-151
host0 = as151.getHost('host0')

# Insteall software on the host
host0.addSoftware('telnetd').addSoftware('telnet')

# Import a file to the host
host0.importFile(hostpath="/home/seed/ddos.py",
                 nodepath="/tmp/ddos.py")
```

```
# Create a file on the host
host0.setFile(content="some content",
              path="/tmp/file.txt")

# This command is executed when the container is built
host0.addBuildCommand('useradd -m -s /bin/bash seed
                       && echo "seed:dees" | chpasswd')

# Append a command to the start script
host0.appendStartCommand('cd /bof && /bof/server &')
```

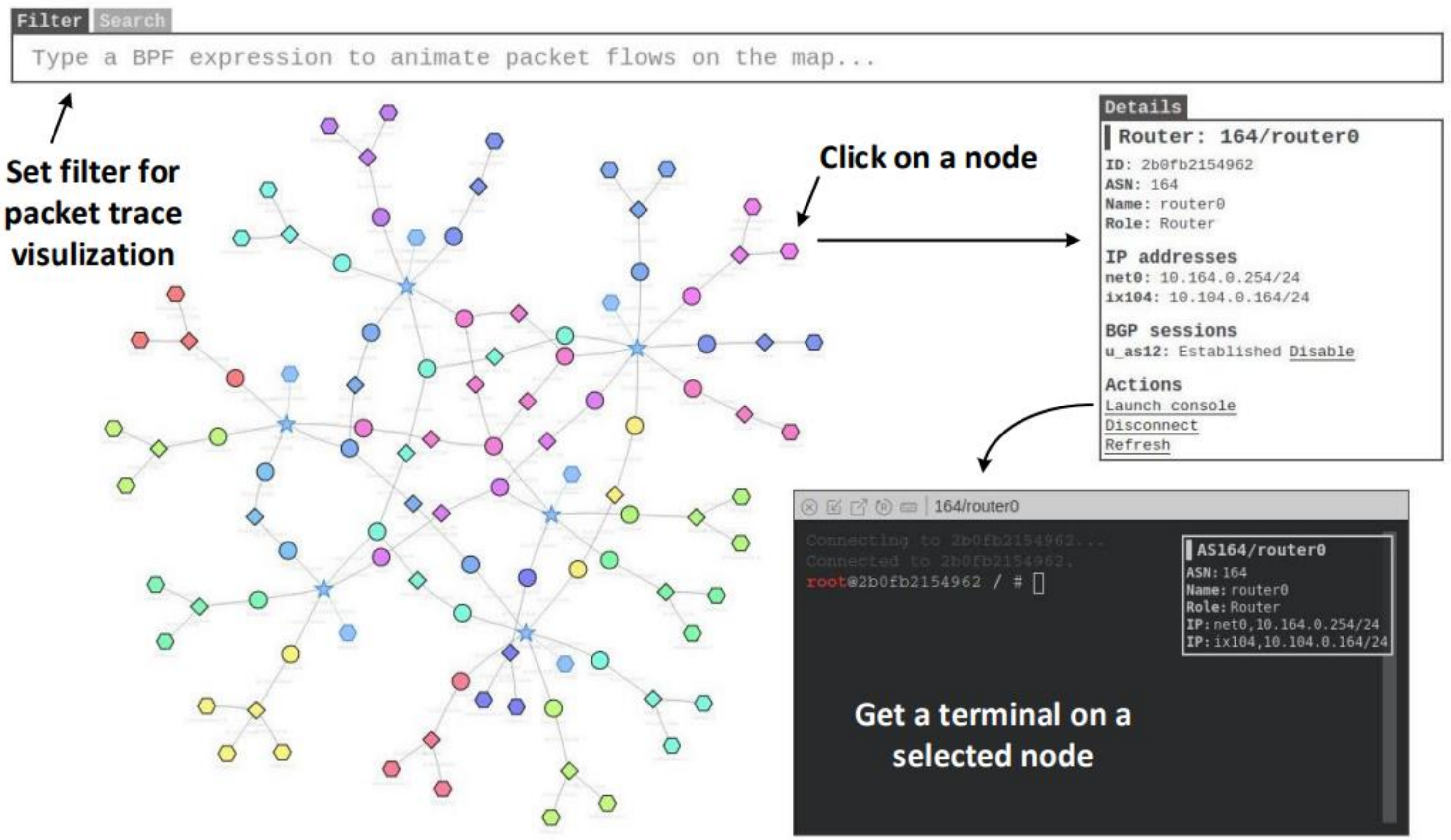
Shadow Internet



```
as152 = base.getAutonomousSystem(152)
as152.getNetwork('net0').enableRemoteAccess(ovpn)
```

```
as11872 = base.createAutonomousSystem(11872)
as11872.createRealWorldRouter('rw').
        joinNetwork('ix102', '10.102.0.118')
```

Visualization Tool: the Map



Set filter for packet trace visulization

Click on a node

Details

Router: 164/router0
ID: 2b0fb2154962
ASN: 164
Name: router0
Role: Router

IP addresses
net0: 10.164.0.254/24
ix104: 10.104.0.164/24

BGP sessions
u_as12: Established Disable

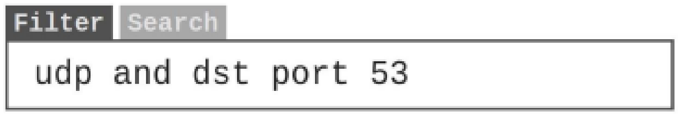
Actions
Launch console
Disconnect
Refresh

164/router0

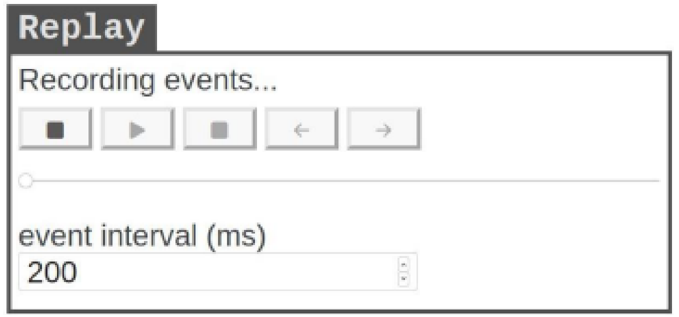
```
Connecting to 2b0fb2154962...
Connected to 2b0fb2154962.
root@2b0fb2154962 / #
```

AS164/router0
ASN: 164
Name: router0
Role: Router
IP: net0,10.164.0.254/24
IP: ix104,10.104.0.164/24

Get a terminal on a selected node



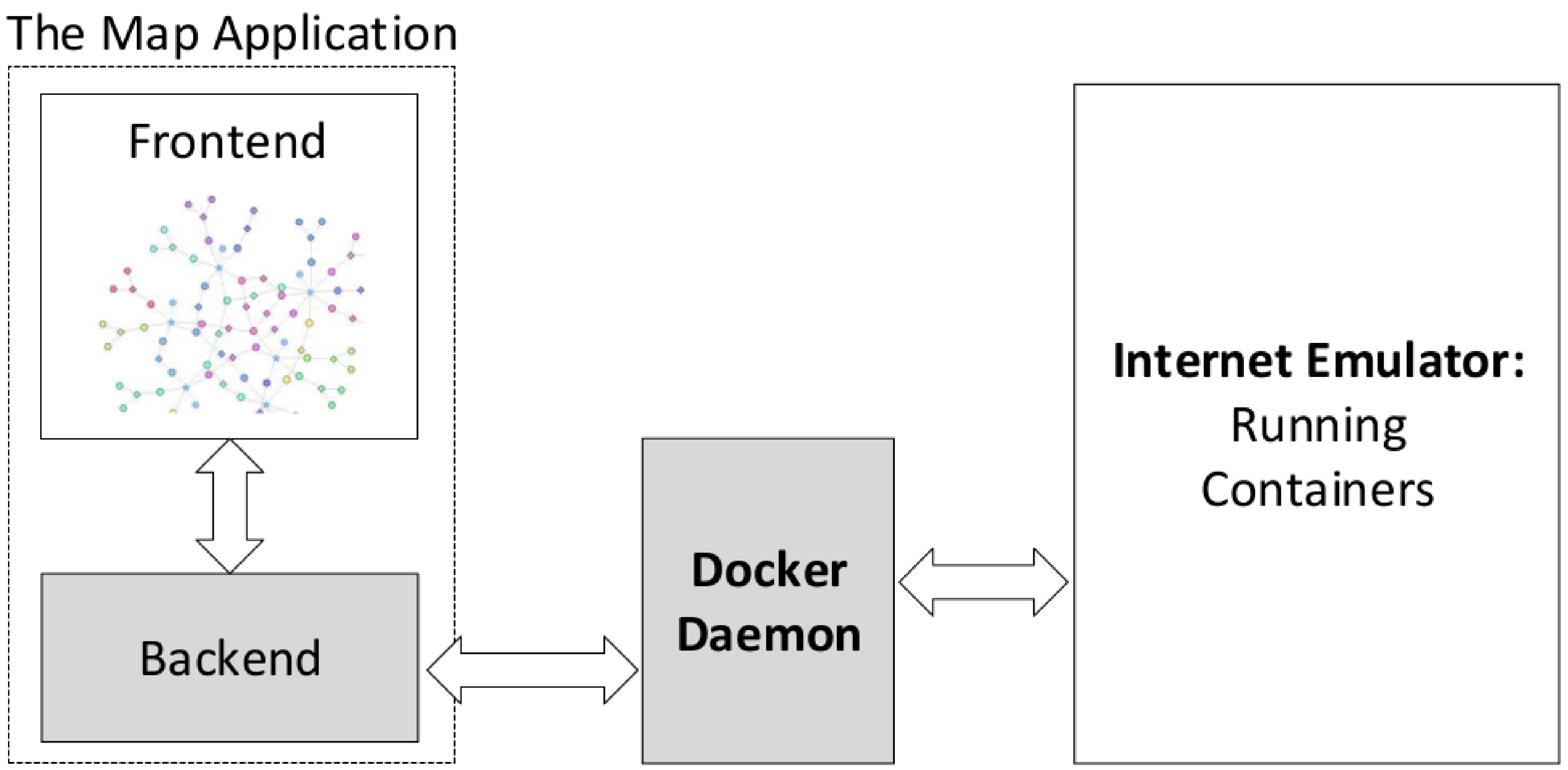
udp and dst port 53



Recording events...

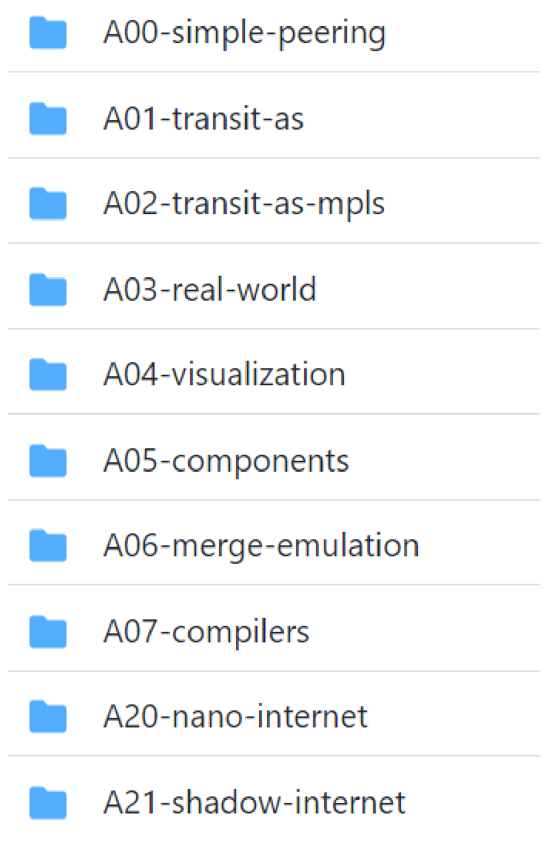
event interval (ms): 200

Visualization Tool: Design

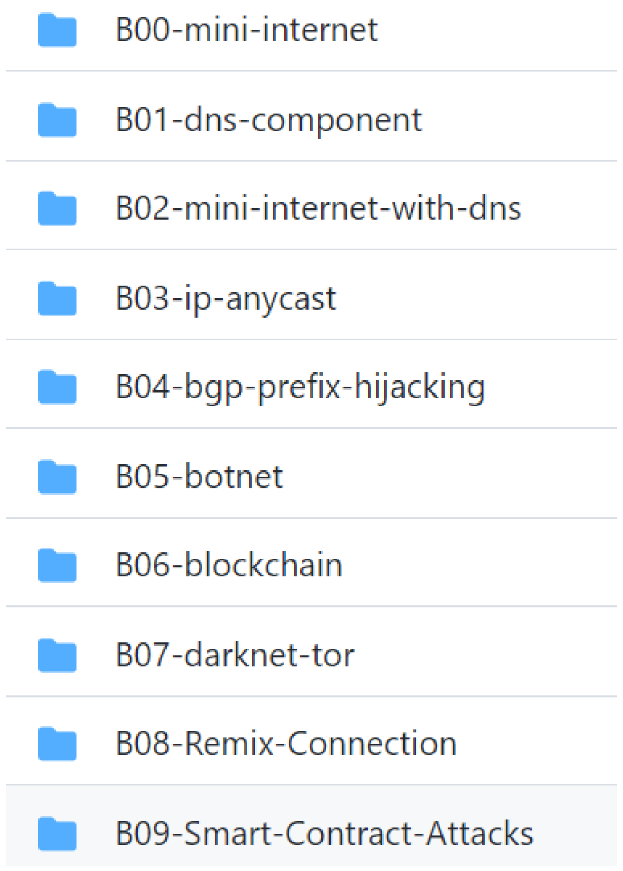


Demo: Building Internet Emulator

Code: inside the examples/ folder



- A00-simple-peering
- A01-transit-as
- A02-transit-as-mpls
- A03-real-world
- A04-visualization
- A05-components
- A06-merge-emulation
- A07-compilers
- A20-nano-internet
- A21-shadow-internet



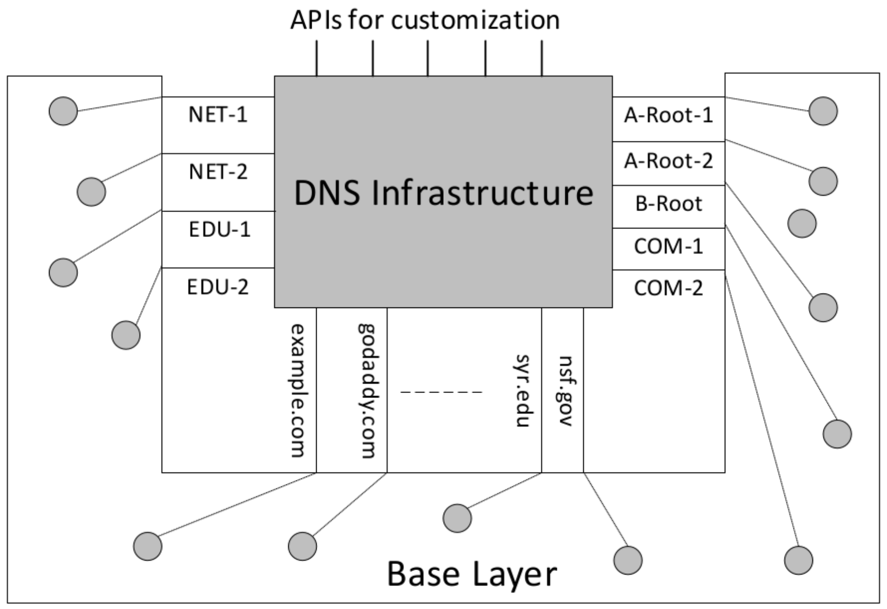
- B00-mini-internet
- B01-dns-component
- B02-mini-internet-with-dns
- B03-ip-anycast
- B04-bgp-prefix-hijacking
- B05-botnet
- B06-blockchain
- B07-darknet-tor
- B08-Remix-Connection
- B09-Smart-Contract-Attacks

Components

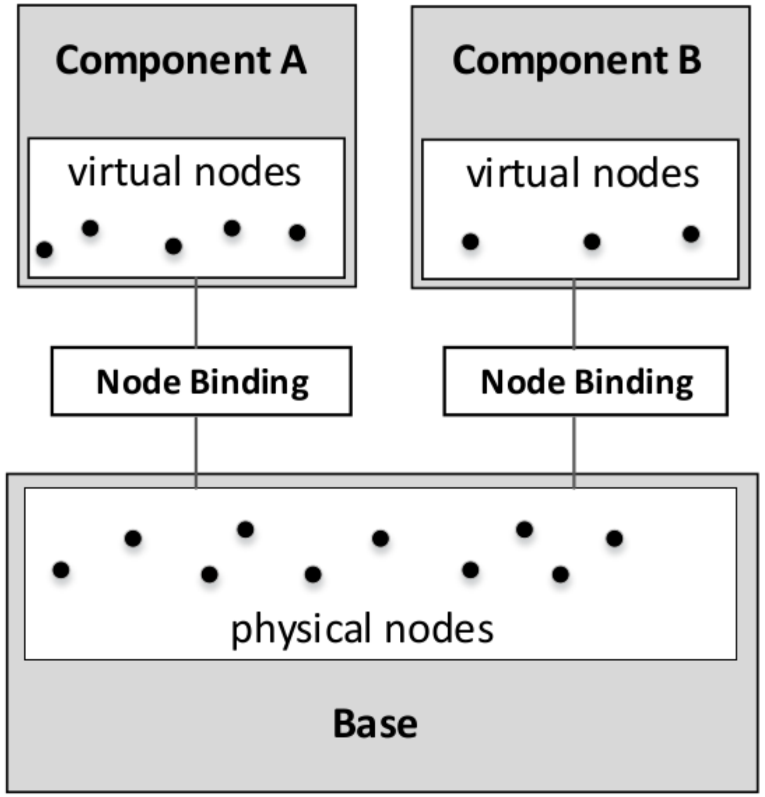
Components

Components (Class + Object)

- DNS infrastructure
- Botnet
- Darknet
- A national/state backbone
- A company's networks
- Blockchain



Extensible Design for Components

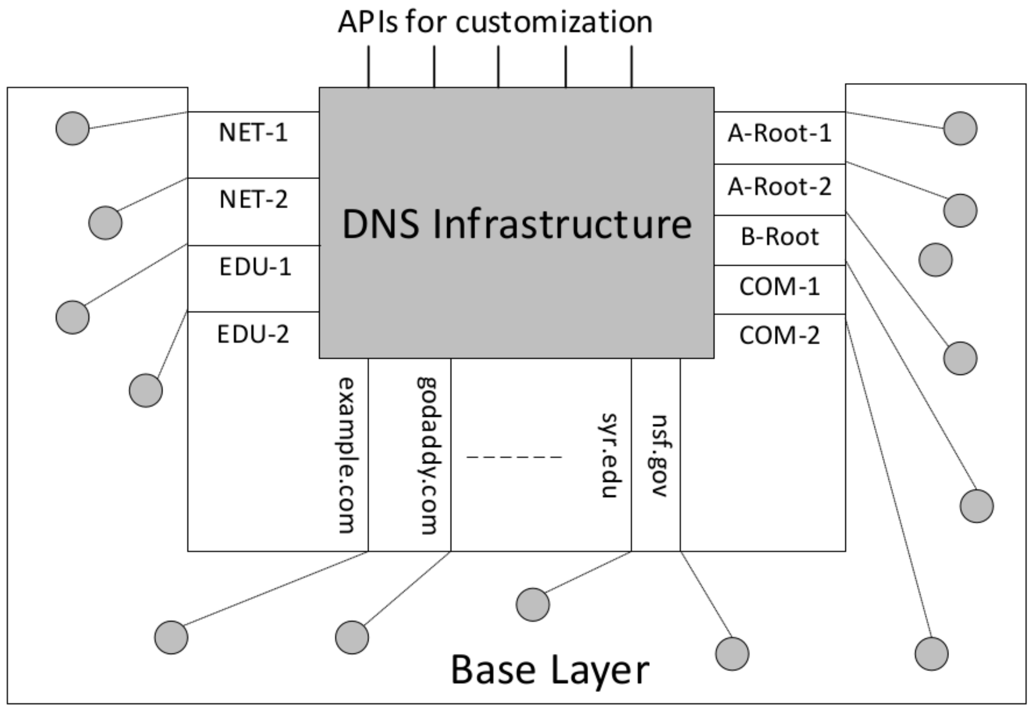


```
emu.addBinding(Binding('root-a', filter=Filter(asn=171))
emu.addBinding(Binding('root-b', filter=Filter(asn=150))
emu.addBinding(Binding('com-a',  filter=Filter(asn=151))
emu.addBinding(Binding('ns-syr-edu',
                       filter=Filter(asn=152))
```

```
as_list = [150, 151, 152, 153, 154, 160, 161, 162]
for counter in range(10):
  vname = 'bot-node-%.2d'%(counter)
  asn = random.choice(as_list)
  emu.addBinding(Binding(vname,
      filter=Filter(asn=asn), action=Action.NEW))
```

DNS: A Component Example

```
# Create a DNS layer
dns = DomainNameService()

# Create two nameservers for the root zone
dns.install('root-a').addZone('.').setMaster()
dns.install('root-b').addZone('.')

# Create nameservers for TLD zones
dns.install('com-a').addZone('com.').setMaster()
dns.install('com-b').addZone('com.')
dns.install('edu').addZone('edu.')

# Create nameservers for second-level zones
dns.install('ns-example-com').addZone('example.com.')
dns.install('ns-syr-edu').addZone('syr.edu.')

# Add records to zones
dns.getZone('example.com.').addRecord('@ A 2.2.2.2')
                           .addRecord('www A 5.5.5.5')
                           .addRecord('xyz A 5.5.5.6')
```

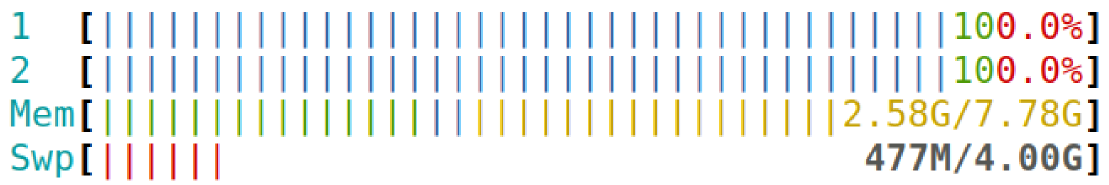


Blockchain Component

```
# Create Ethereum nodes
e1 = eth.install("eth1").startMiner()
e2 = eth.install("eth2").startMiner()
e3 = eth.install("eth3").startMiner()
e4 = eth.install("eth4").startMiner()
e5 = eth.install("eth5")
e6 = eth.install("eth6")

# Set bootnodes on e1 and e2.
# The other nodes can use these bootnodes to find peers.
e1.setBootNode(True)
e2.setBootNode(True)

# Deploy a smartcontract on e3
contract = SmartContract("./Contracts/contract.bin",
                         "./Contracts/contract.abi")
e3.deploySmartContract(contract)
```

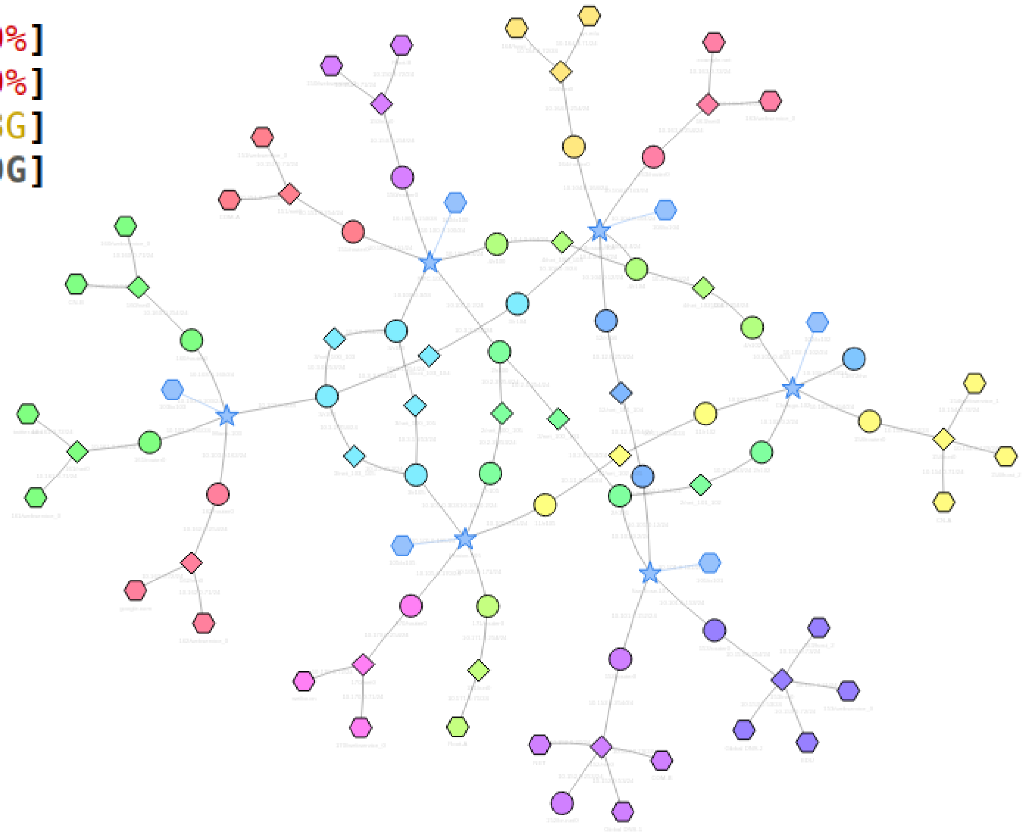
Adding Blockchain to Emulator

- Host Machine (Ubuntu 20.04 VM)



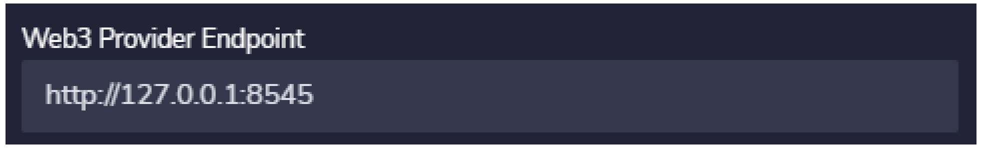
```
1   [|||||||||||||||||||||||||||||||||||||||100.0%]
2   [|||||||||||||||||||||||||||||||||||||||100.0%]
Mem[||||||||||||||||||||||||||||||||||2.58G/7.78G]
Swp[||||||                             477M/4.00G]
```

- Mini-Internet
  - **63** machines (containers)
  - **34** networks
  - **6** Internet exchanges
  - **13** stub autonomous systems
  - **5** transit autonomous systems
  - Blockchain
    - **4** mining nodes
    - **2** non-mining nodes



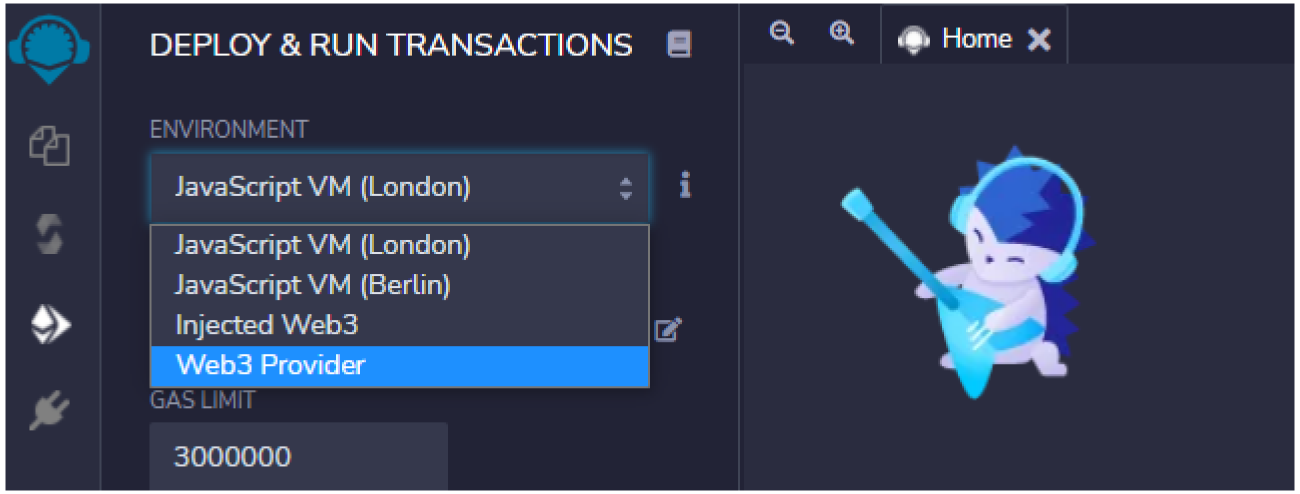
Blockchain: Integrating with Existing Tools

```
e6.startMiner().createNewAccount(2)
               .unlockAccounts()
               .enableExternalConnection()
```



Web3 Provider Endpoint

http://127.0.0.1:8545



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT

JavaScript VM (London)

- JavaScript VM (London)
- JavaScript VM (Berlin)
- Injected Web3
- Web3 Provider

GAS LIMIT

3000000

Applications: Labs

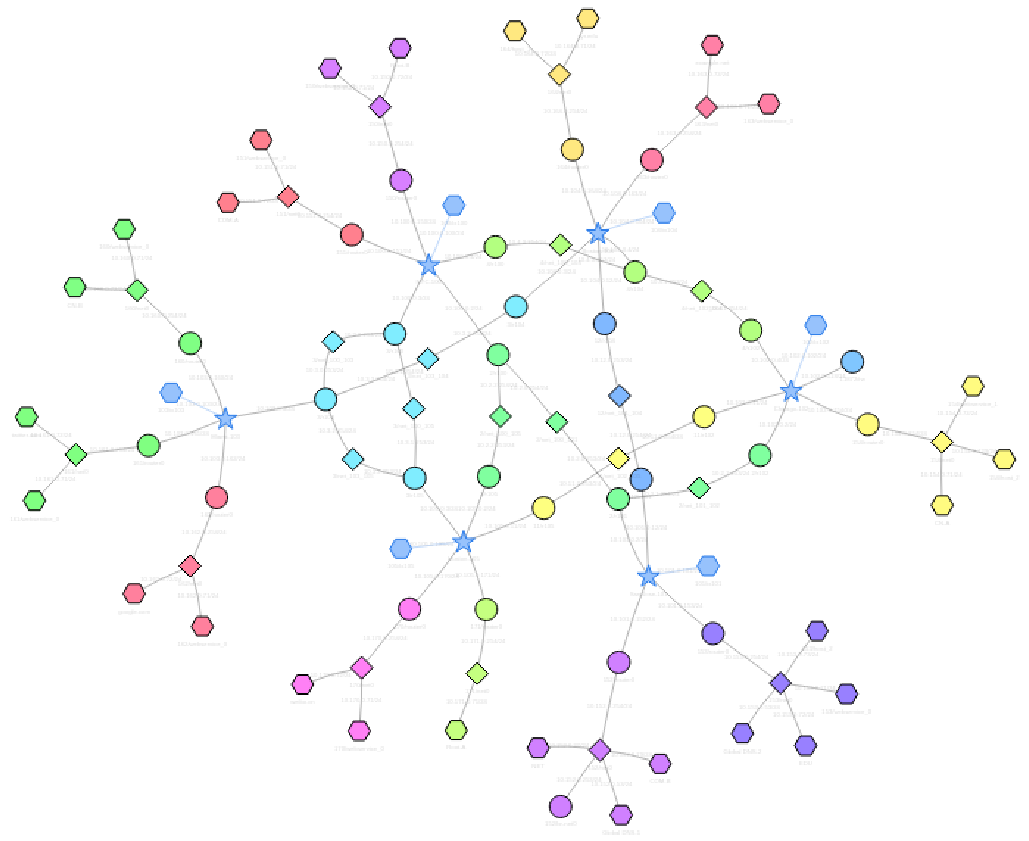
Applications

BGP & Attacks	DNS Infrastructure	Botnet	Blockchain
Yesterday Once More	Ransomware	Morris Worm	Darknet
SDN (Software Defined Network)	CDN (Content Delivery Network)	???	???

Demo: BGP Attack Lab

- 2008: Pakistan Hijacked YouTube
- **Hijack 10.154.0.0/24** (AS-154)
- Attacker: **AS-161**

```
protocol static {
  ipv4 { table t_bgp;  };
  route 10.154.0.0/25 blackhole   {
      bgp_large_community.add(LOCAL_COMM);
  };
  route 10.154.0.128/25 blackhole {
      bgp_large_community.add(LOCAL_COMM);
  };
}
```



BGP Attack: Fight Back

Fight back (by AS-154)

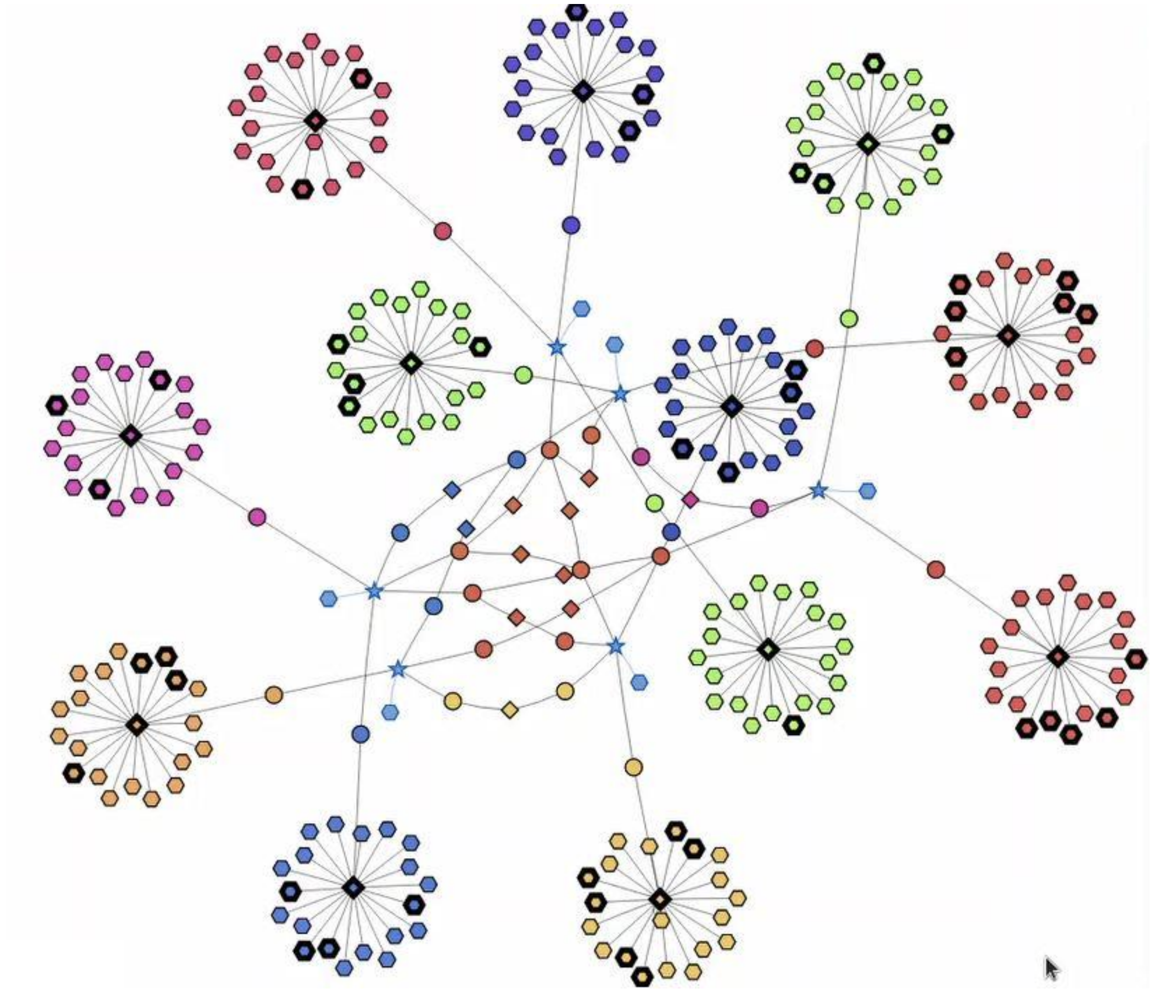
```
protocol static {
  ipv4 { table t_bgp; };
  route 10.154.0.0/26 via "net0" {
           bgp_large_community.add(LOCAL_COMM);
  };
  route 10.154.0.64/26 via "net0" {
           bgp_large_community.add(LOCAL_COMM);
  };
  route 10.154.0.128/26 via "net0" {
           bgp_large_community.add(LOCAL_COMM);
  };
  route 10.154.0.192/26 via "net0" {
           bgp_large_community.add(LOCAL_COMM);
  };
}
```

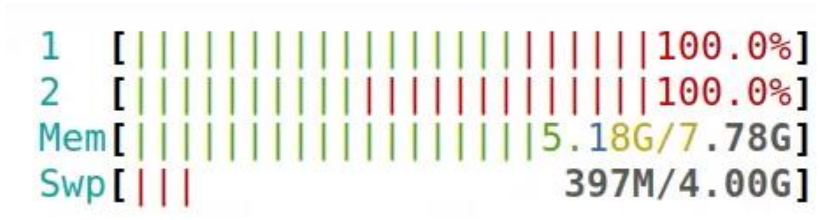
Block it on upstream ISP

```
protocol bgp c_as161 {
 ipv4 {
     table t_bgp;
     import filter {
         bgp_large_community.add(CUSTOMER_COMM);
         bgp_local_pref = 30;
         if (net != 10.161.0.0/24) then reject;
         accept;
     };
     ...
 };
 ...
}
```

Morris Worm Lab

Demo video https://youtu.be/2VZV-aFoVjk



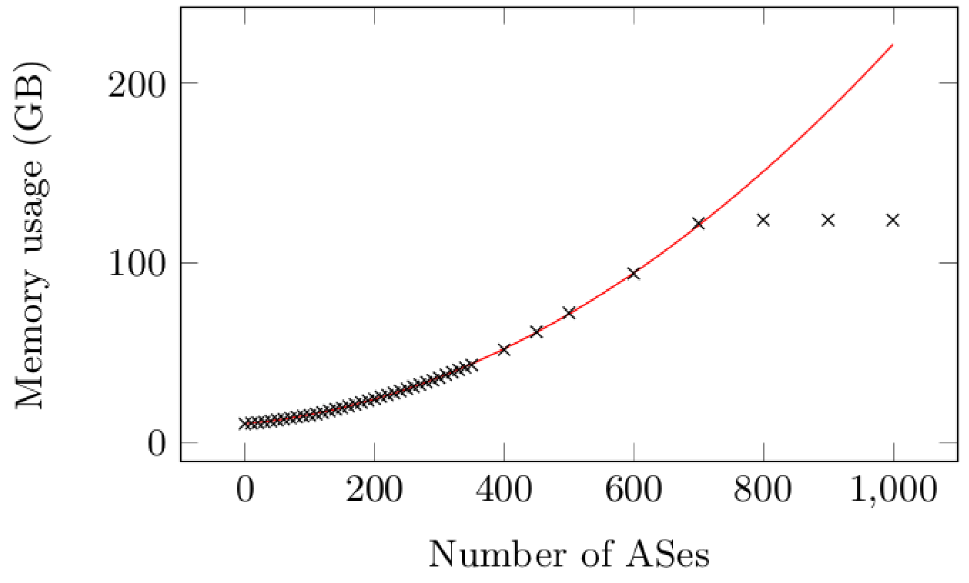


```
1   [||||||||||||||||||||||||||100.0%]
2   [||||||||||||||||||||||||||100.0%]
Mem[||||||||||||||||||||5.18G/7.78G]
Swp[|||                 397M/4.00G]
```

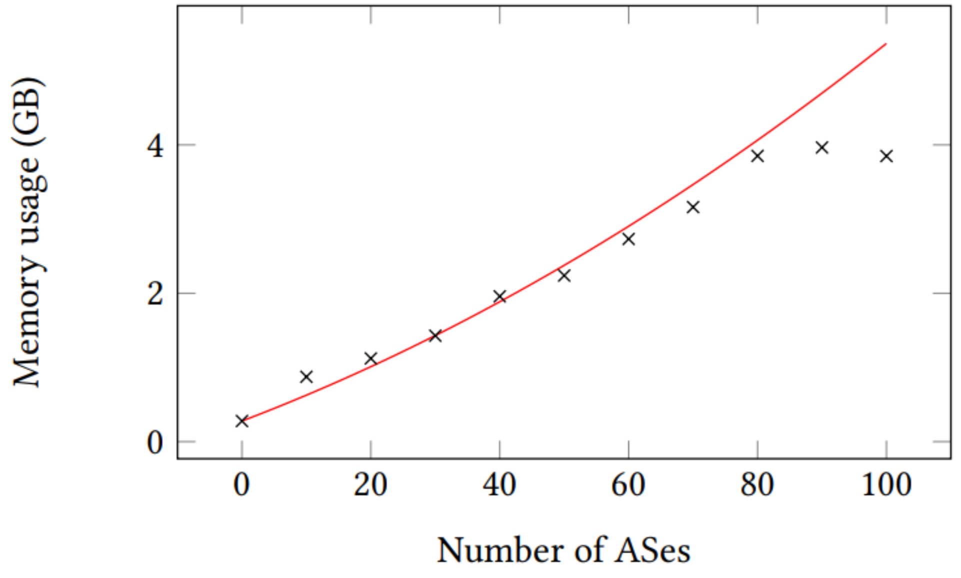
Performance

Memory Usage

Powerful Server (20 cores, 120 GB of RAM)

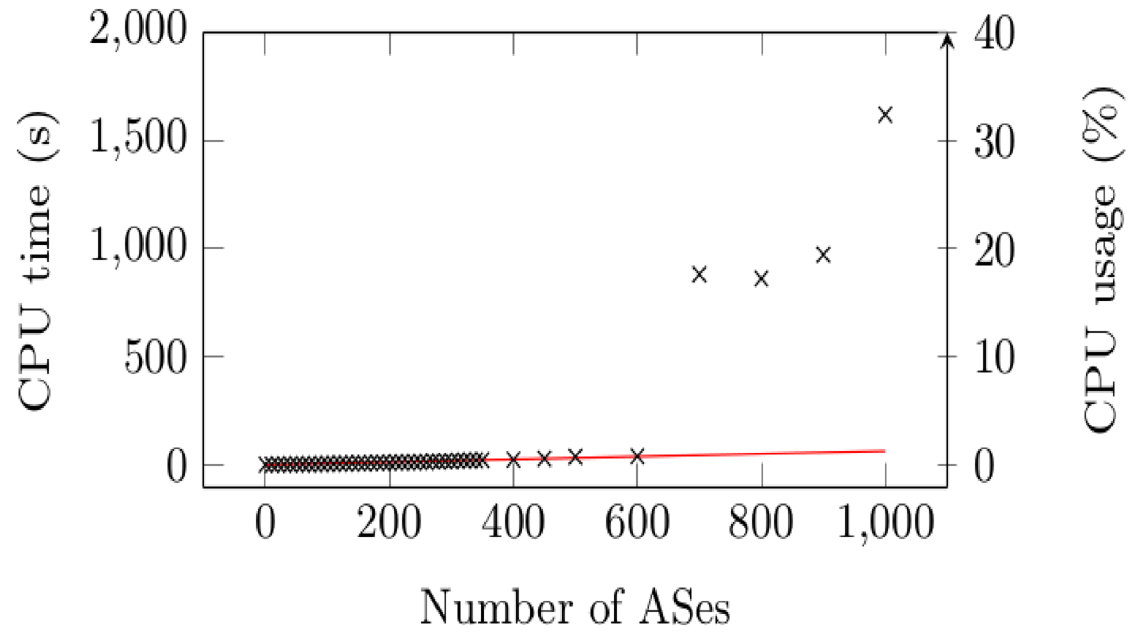


Virtual Machine (2 cores, 4 GB of RAM)



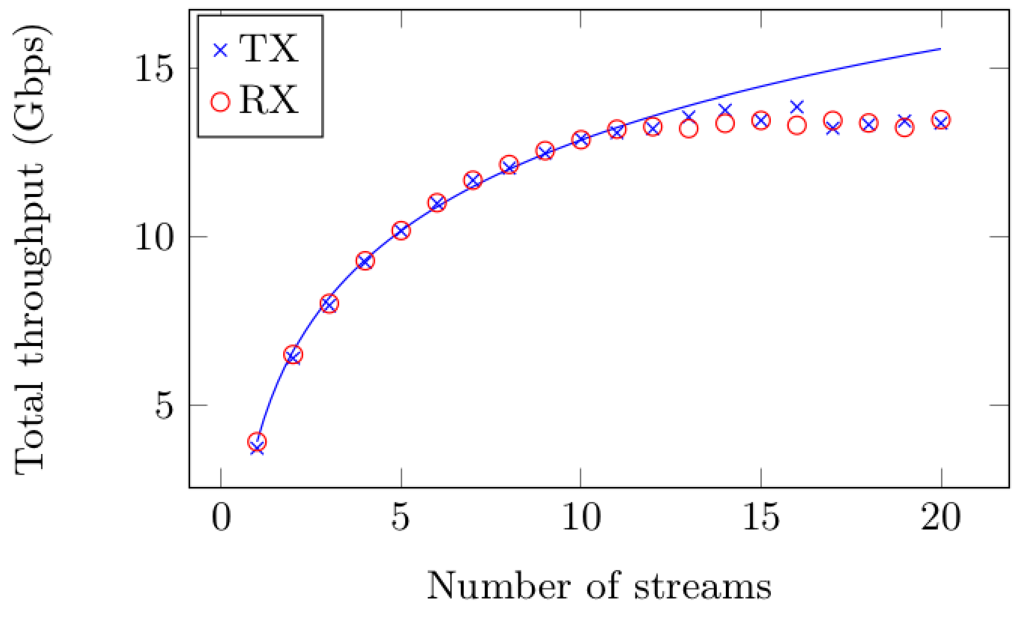
CPU

Powerful Server

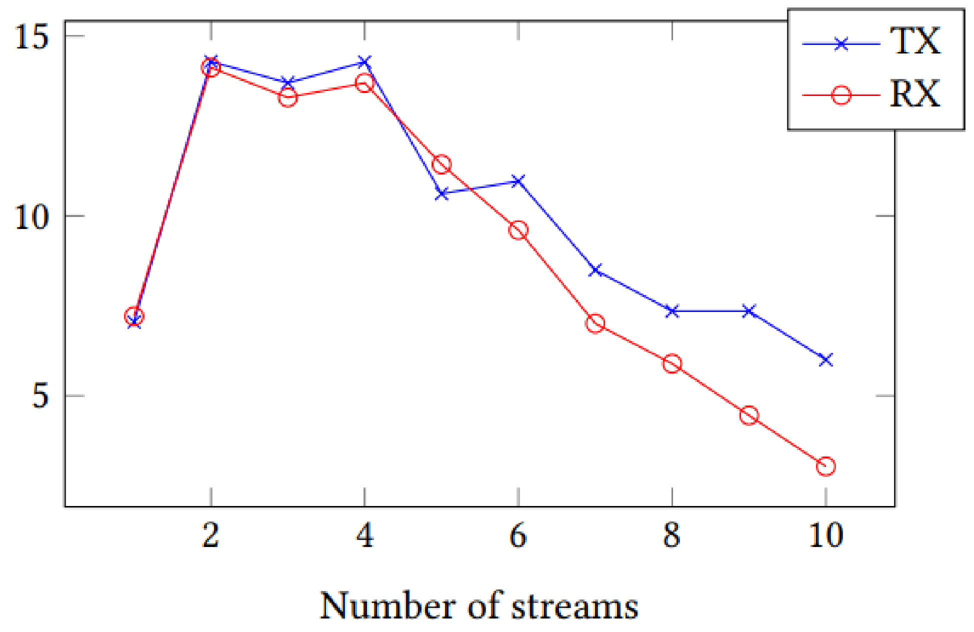


Network Throughput

Powerful Server

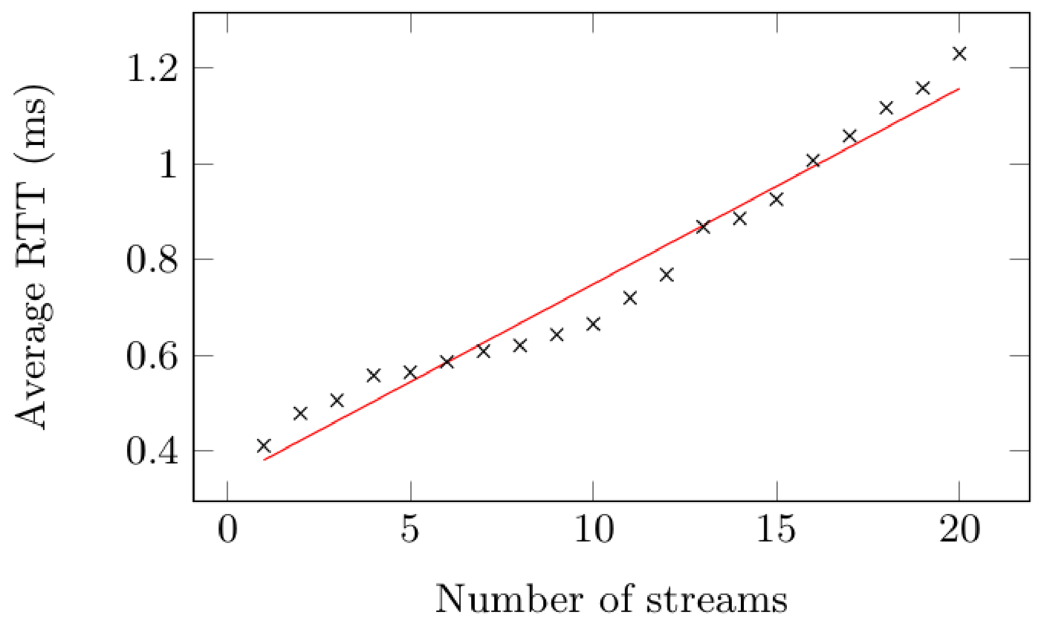


Virtual Machine



Round Trip Time

Powerful Server



Additional Information

Getting the Code

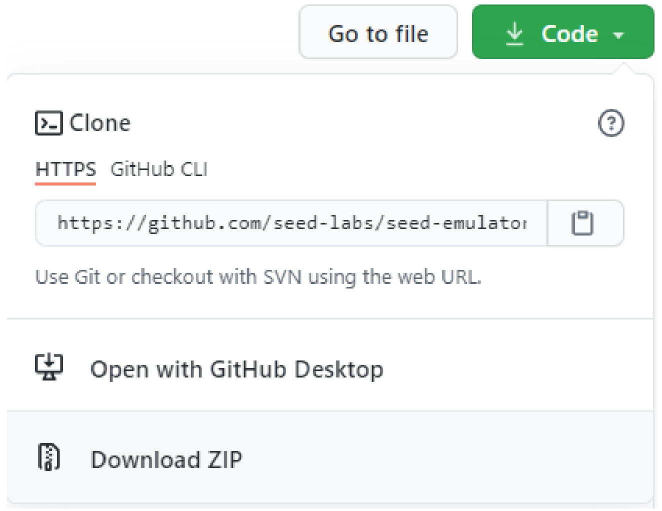
- Download the source code

  GitHub: **https://github.com/seed-labs/seed-emulator**

- Set up the development environment

```
$ source development.env
```

- The examples/ folder

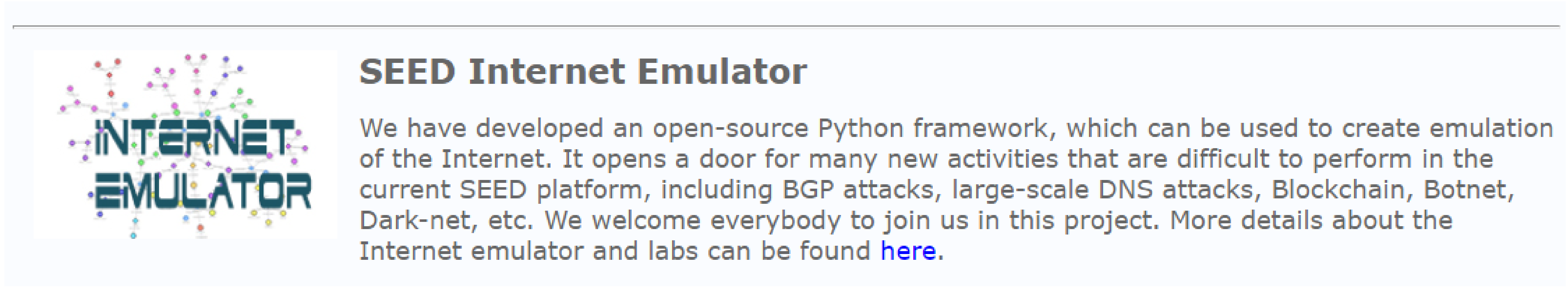


```
https://github.com/seed-labs/seed-emulator
```

Use Git or checkout with SVN using the web URL.

Additional Information

SEED Website: **https://seedsecuritylabs.org/**

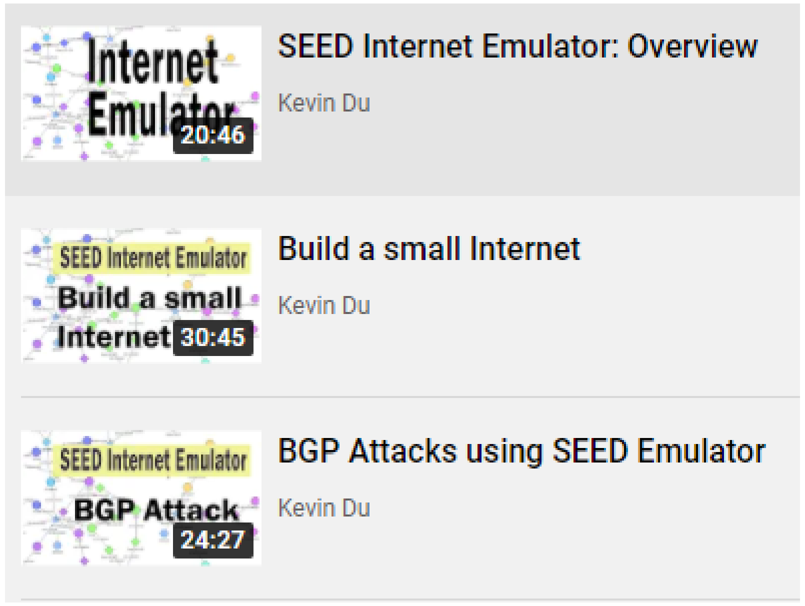


SEED Internet Emulator

We have developed an open-source Python framework, which can be used to create emulation of the Internet. It opens a door for many new activities that are difficult to perform in the current SEED platform, including BGP attacks, large-scale DNS attacks, Blockchain, Botnet, Dark-net, etc. We welcome everybody to join us in this project. More details about the Internet emulator and labs can be found here.

Emulator-Based Labs | Videos | Code and Documentation

YouTube Videos



SEED Internet Emulator: Overview

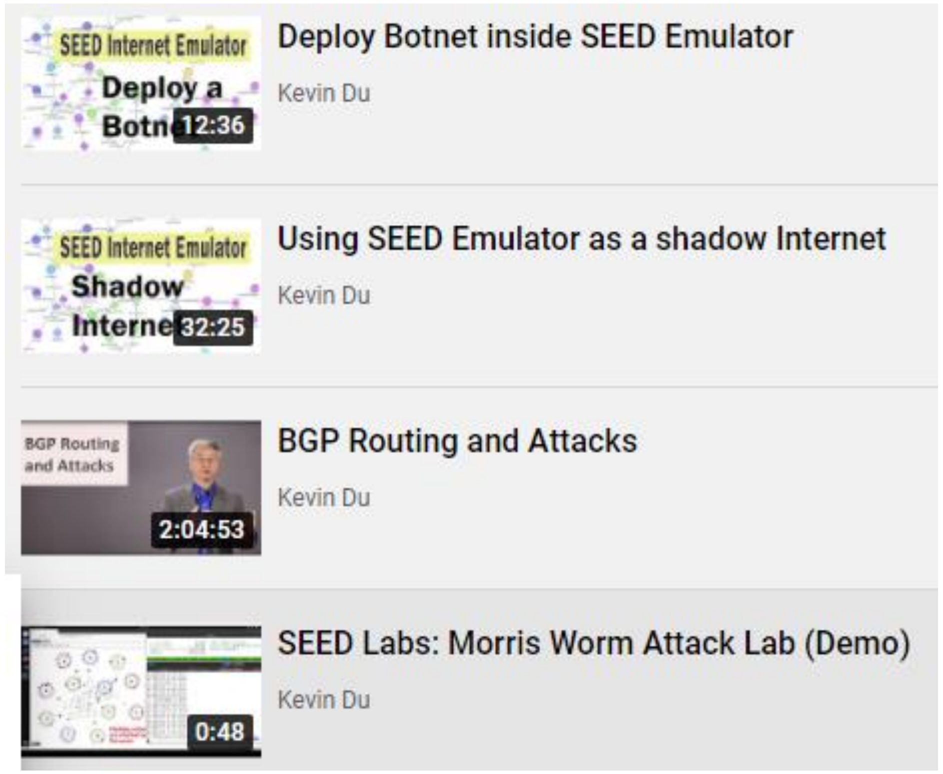
Kevin Du

Build a small Internet

Kevin Du

BGP Attacks using SEED Emulator

Kevin Du



Deploy Botnet inside SEED Emulator

Kevin Du

Using SEED Emulator as a shadow Internet

Kevin Du

BGP Routing and Attacks

Kevin Du

SEED Labs: Morris Worm Attack Lab (Demo)

Kevin Du

Summary

- The SEED Internet Emulator
  - Design
  - Applications in cybersecurity education
  - Performance
- This is an open-source project
- Questions?