FireSim

FireSim Multi-FPGA Networked Simulation

https://fires.im

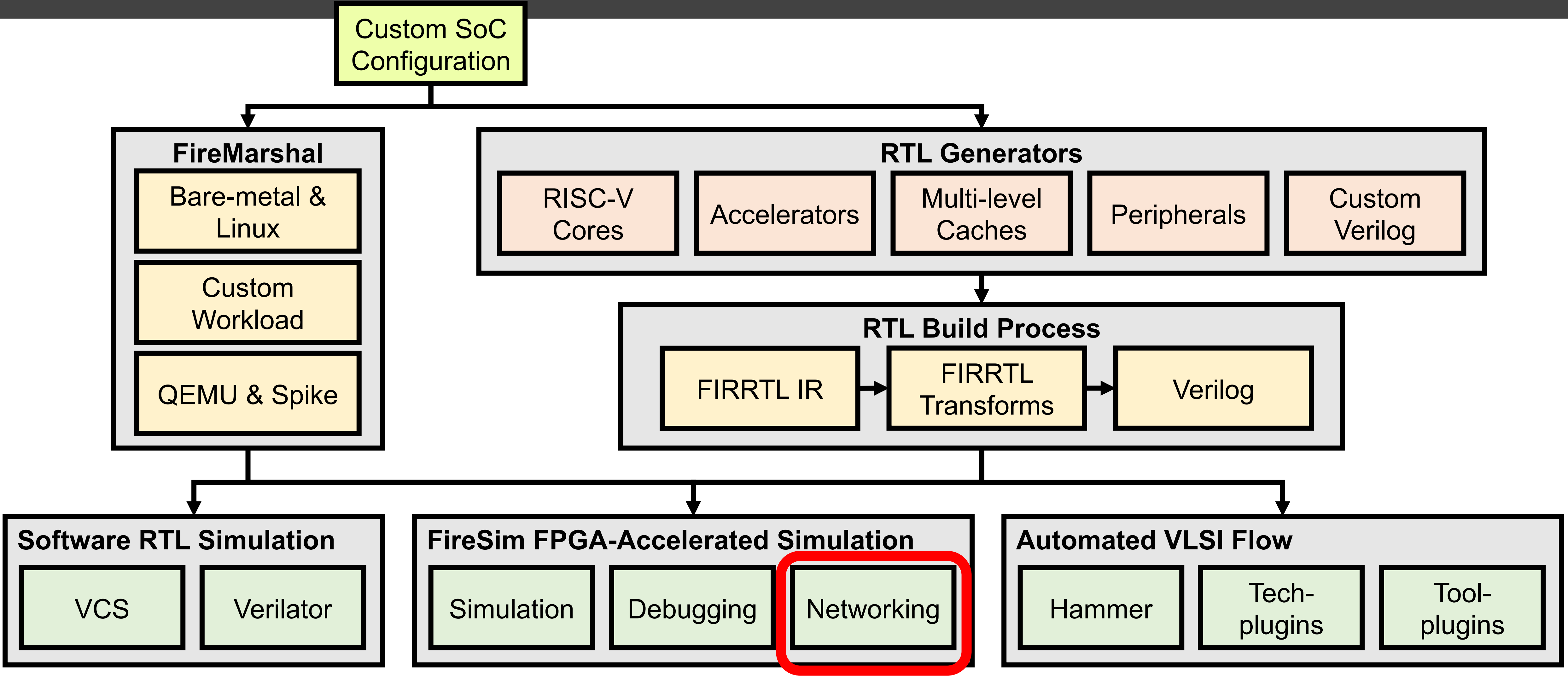
@firesimproject

ASPLOS 2022 Tutorial

Speaker: Abraham Gonzalez

Berkeley Architecture Research

Tutorial Roadmap



Agenda

- Configuring Network Parameters
- Setting Up a Network Topology
- Network topology examples
- Hand-on example with a heterogenous 2-node network.

Network Parameters

- Network parameters are defined in `$FDIR/deploy/config_runtime.ini`
- Network parameters
  - `linklatency` – link latency (measured in cycles). Default is 6405
  - `switchlatency` – minimum port-to-port packet switching latency within a switch (measured in cycles). Default is 10
  - `netbandwidth` – maximum output network bandwidth of each switch (measured in integer Gbit/s). Default is 200

```
[targetconfig]
topology=example_8config
no_net_num_nodes=2
linklatency=6405
switchinglatency=10
netbandwidth=200
profileinterval=-1
```

Writing a Network Topology

- Network topology definitions found in: `$FDIR/deploy/runtools/user_topology.py`
- Basic Elements:
  - `FireSimServerNode()`
  - `FireSimSwitchNode()`
  - `<some_node>.add_downlinks(<list_of_downstream_nodes>)`
- Compose a network topology in a hierarchical fashion

Example (Using a single f1.4xlarge)

- Smallest Network example
- 2-node configuration (with a single switch)

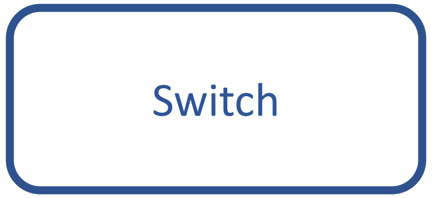
```
def example_2config(self):
    self.roots = [FireSimSwitchNode()]
    servers = [FireSimServerNode() for y in range(2)]
    self.roots[0].add_downlinks(servers)
```

Example (Using a single f1.4xlarge)

```
def example_2config(self):
```

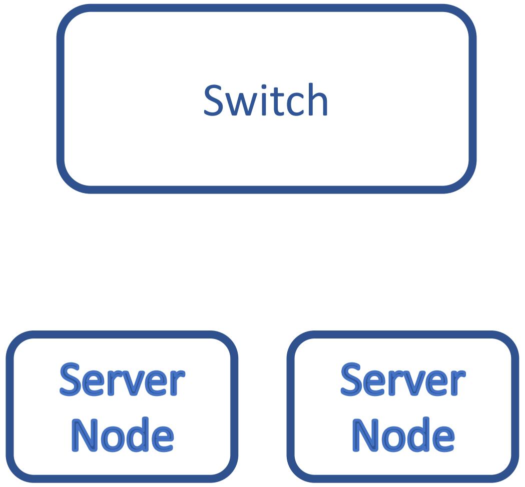
Example (Using a single f1.4xlarge)

```
def example_2config(self):
    self.roots = [FireSimSwitchNode()]
```



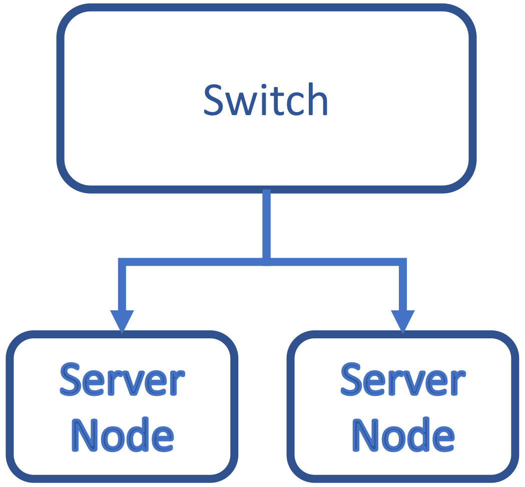
Example (Using a single f1.4xlarge)

```
def example_2config(self):
    self.roots = [FireSimSwitchNode()]
    servers = [FireSimServerNode() for y in range(2)]
```



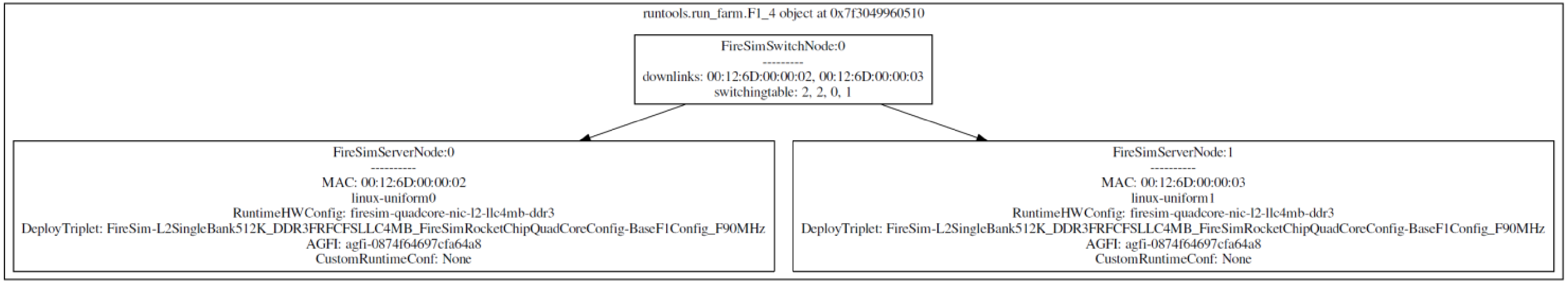
Example (Using a single f1.4xlarge)

```
def example_2config(self):
    self.roots = [FireSimSwitchNode()]
    servers = [FireSimServerNode() for y in range(2)]
    self.roots[0].add_downlinks(servers)
```



Verify The Topology

- The firesim command `firesim runcheck` will generate a visualization of the network topology that is currently defined in `config_runtime.ini`
  - Including assigned HW configuration, IP and MAC
- The outputted diagram will be located in `$FDIR/deploy/generated-topology-diagrams/`


runtools.run_farm.F1_4 object at 0x7f3049960510
FireSimSwitchNode:0
----
downlinks: 00:12:6D:00:00:02, 00:12:6D:00:00:03
switchingtable: 2, 2, 0, 1
FireSimServerNode:0

MAC: 00:12:6D:00:00:02
linux-uniform0
RuntimeHWConfig: firesim-quadcore-nic-l2-llc4mb-ddr3
DeployTriplet: FireSim-L2SingleBank512K_DDR3FRFCFSLLC4MB_FireSimRocketChipQuadCoreConfig-BaseF1Config_F90MHz
AGFI: agfi-0874f64697cfa64a8
CustomRuntimeConf: None
FireSimServerNode:1

MAC: 00:12:6D:00:00:03
linux-uniform1
RuntimeHWConfig: firesim-quadcore-nic-l2-llc4mb-ddr3
DeployTriplet: FireSim-L2SingleBank512K_DDR3FRFCFSLLC4MB_FireSimRocketChipQuadCoreConfig-BaseF1Config_F90MHz
AGFI: agfi-0874f64697cfa64a8
CustomRuntimeConf: None

example_2config topology diagram

Heterogenous Topology Example

- `FireSimServerNode()` can take an argument called `server_hardware_config` with the AFI descriptor name
- If we want to create a topology with 2 nodes, one with the SHA3 accelerator and one with BOOM we will describe it as follows:

```
def example_sha3hetero_2config(self):
  self.roots = [FireSimSwitchNode()]
  servers = [FireSimServerNode(server_hardware_config=
                               "firesim-boom-singlecore-nic-l2-llc4mb-ddr3"),
            FireSimServerNode(server_hardware_config=
                               "firesim-rocket-singlecore-sha3-nic-l2-llc4mb-ddr3")]
  self.roots[0].add_downlinks(servers)
```

Heterogenous Topology Example: Hands-on

- Add/Un-comment the `example_sha3hetero_2config` at the bottom of your `$FDIR/deploy/runtools/user_topology.py`

```
def example_sha3hetero_2config(self):
  self.roots = [FireSimSwitchNode()]
  servers = [FireSimServerNode(server_hardware_config=
                               "firesim-boom-singlecore-nic-l2-llc4mb-ddr3"),
             FireSimServerNode(server_hardware_config=
                               "firesim-rocket-singlecore-sha3-nic-l2-llc4mb-ddr3")]
  self.roots[0].add_downlinks(servers)
```

Heterogenous Topology Example: Hands-on

- Update `$FDIR/deploy/config_runtime.ini` with the appropriate resources and topology
  - `vim $FDIR/deploy/config_runtime.ini`
  - One `f1.4xlarge` instance is sufficient for a 2-node simulation since it includes 2 FPGAs

```
f1_16xlarges=0
m4_16xlarges=0
f1_4xlarges=1
f1_2xlarges=0

runinstancemarket=ondemand
spotinterruptionbehavior=terminate
spotmaxprice=ondemand

[targetconfig]
topology=example_sha3hetero_2config
no_net_num_nodes=2
linklatency=6405
switchinglatency=10
netbandwidth=200
profileinterval=-1

[workload]
workloadname=linux-uniform.json
terminateoncompletion=no
```

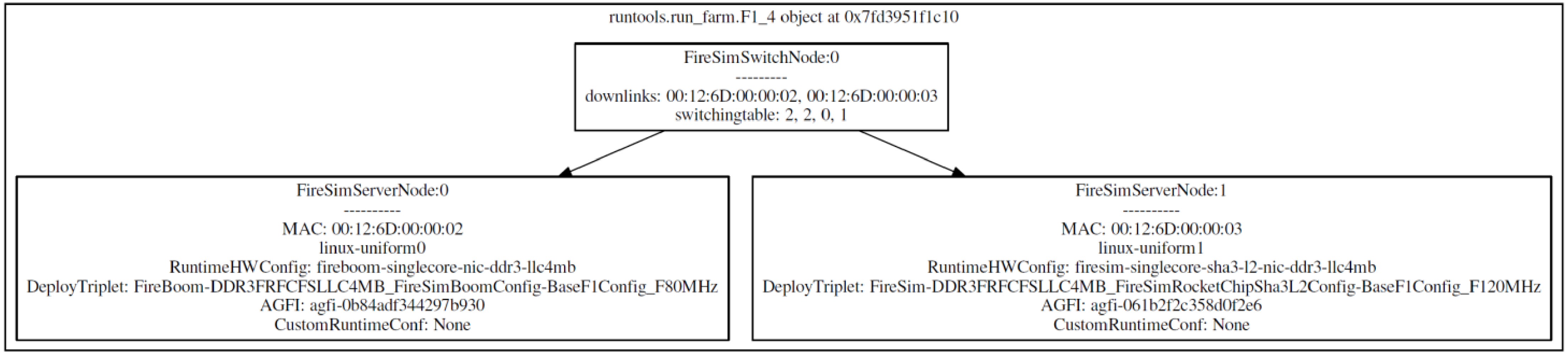
Heterogenous Topology Example: Hands-on

- Verify your topology by running

```
$ firesim runcheck
```

- If you have GUI/X enabled, you can view it at `$FDIR/deploy/generated-topology-diagrams/`

  it should look as follows:



runtools.run_farm.F1_4 object at 0x7fd3951f1c10

FireSimSwitchNode:0
----
downlinks: 00:12:6D:00:00:02, 00:12:6D:00:00:03
switchingtable: 2, 2, 0, 1

FireSimServerNode:0

MAC: 00:12:6D:00:00:02
linux-uniform0
RuntimeHWConfig: fireboom-singlecore-nic-ddr3-llc4mb
DeployTriplet: FireBoom-DDR3FRFCFSLLC4MB_FireSimBoomConfig-BaseF1Config_F80MHz
AGFI: agfi-0b84adf344297b930
CustomRuntimeConf: None

FireSimServerNode:1

MAC: 00:12:6D:00:00:03
linux-uniform1
RuntimeHWConfig: firesim-singlecore-sha3-l2-nic-ddr3-llc4mb
DeployTriplet: FireSim-DDR3FRFCFSLLC4MB_FireSimRocketChipSha3L2Config-BaseF1Config_F120MHz
AGFI: agfi-061b2f2c358d0f2e6
CustomRuntimeConf: None

Heterogenous Topology Example: Hands-on

- Boot the simulation by running the following sequence of commands:
  -
    ```
    $ firesim launchrunfarm && firesim
      infrasetup
    ```
    - This should take about 10 minutes
  -
    ```
    $ firesim runworkload
    ```
    - This should take about 2 minutes

While The Simulation is Booting….

- We can have a look at a few other useful examples:
  - Network config using a single f1.16xlarge instance
  - Network config using multiple f1.16xlarge instances
  - Network config using Supernode
  - More complex network configurations

Example (Using a single f1.16xlarge)

- 8-node configuration (with a single switch, 8 server nodes)
- Requires a single f1.16xlarge instance in your runfarm

```
def example_8config(self):
    self.roots = [FireSimSwitchNode()]
    servers = [FireSimServerNode() for y in range(8)]
    self.roots[0].add_downlinks(servers)
```

Example (Using a single f1.16xlarge)

```
def example_8config(self):
```

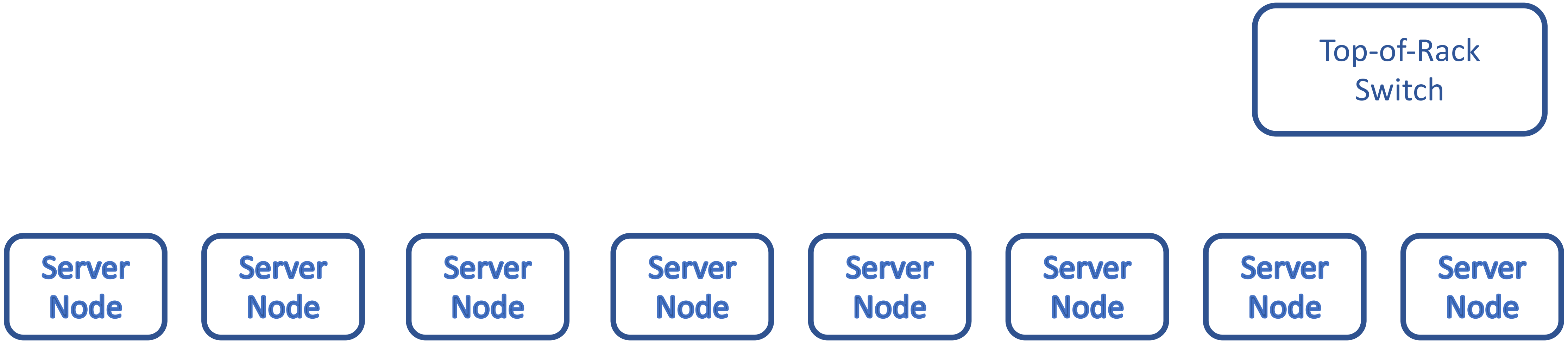
Example (Using a single f1.16xlarge)

```
def example_8config(self):
    self.roots = [FireSimSwitchNode()]
```

Top-of-Rack Switch

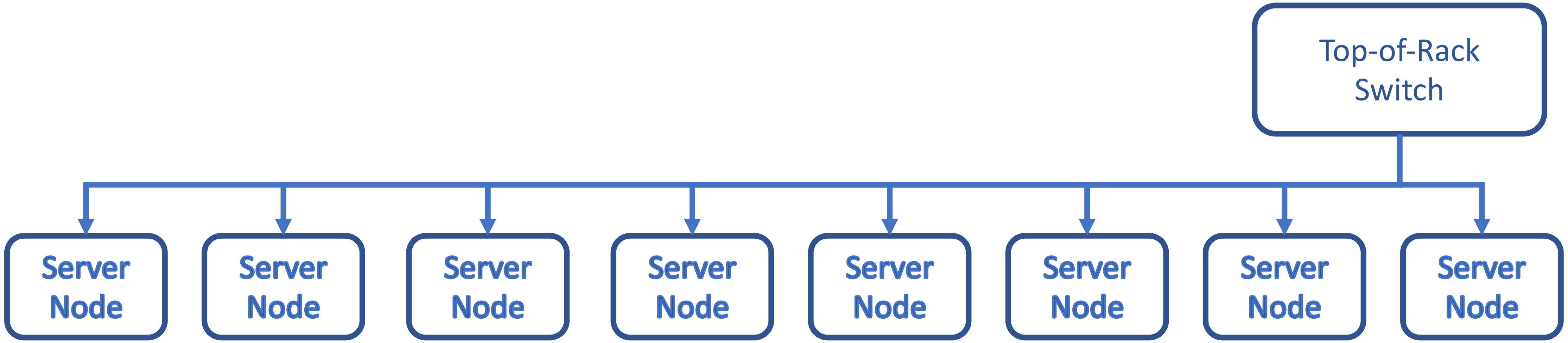
Example (Using a single f1.16xlarge)

```
def example_8config(self):
    self.roots = [FireSimSwitchNode()]
    servers = [FireSimServerNode() for y in range(8)]
```



Example (Using a single f1.16xlarge)

```
def example_8config(self):
    self.roots = [FireSimSwitchNode()]
    servers = [FireSimServerNode() for y in range(8)]
    self.roots[0].add_downlinks(servers)
```



Example (Using multiple f1.16xlarge)

- 64-node configuration (1 aggregation switch, 8 ToR switches, 64 server nodes)
- Requires 8 f1.16xlarge instances, 1 m4.16xlarge instance in your runfarm

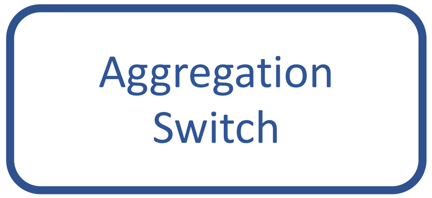
```
def example_64config(self):
    self.roots = [FireSimSwitchNode()]
    level2switches = [FireSimSwitchNode() for x in range(8)]
    servers = [[FireSimServerNode() for y in range(8)] for x in range(8)]

    for root in self.roots:
        root.add_downlinks(level2switches)

    for l2switchNo in range(len(level2switches)):
        level2switches[l2switchNo].add_downlinks(servers[l2switchNo])
```

Example (Using multiple f1.16xlarge)

```
def example_64config(self):
```

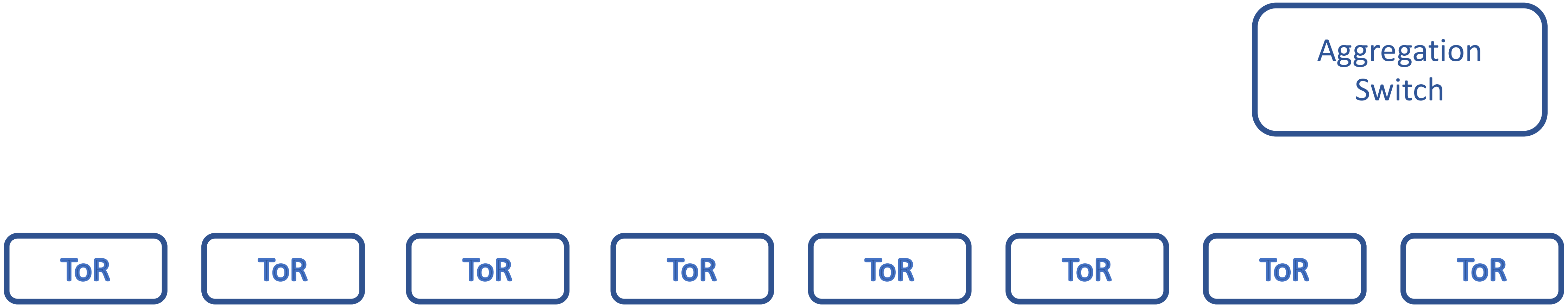
Example (Using multiple f1.16xlarge)

```
def example_64config(self):
    self.roots = [FireSimSwitchNode()]
```



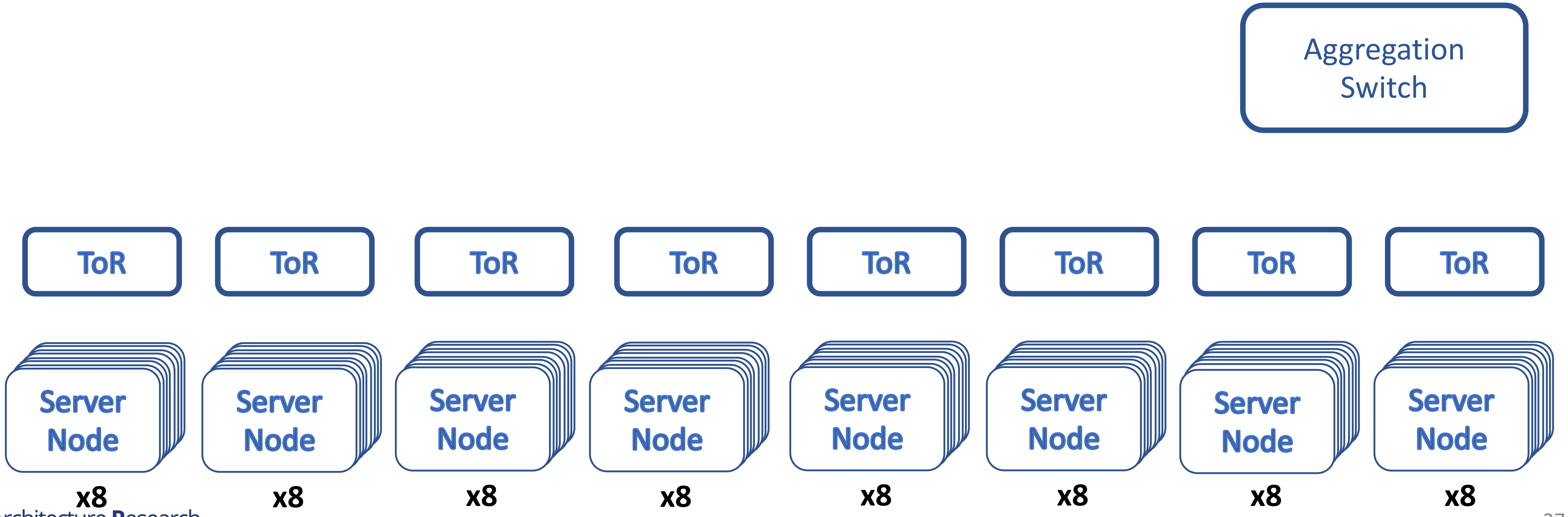
Example (Using multiple f1.16xlarge)

```
def example_64config(self):
    self.roots = [FireSimSwitchNode()]
    level2switches = [FireSimSwitchNode() for x in range(8)]
```



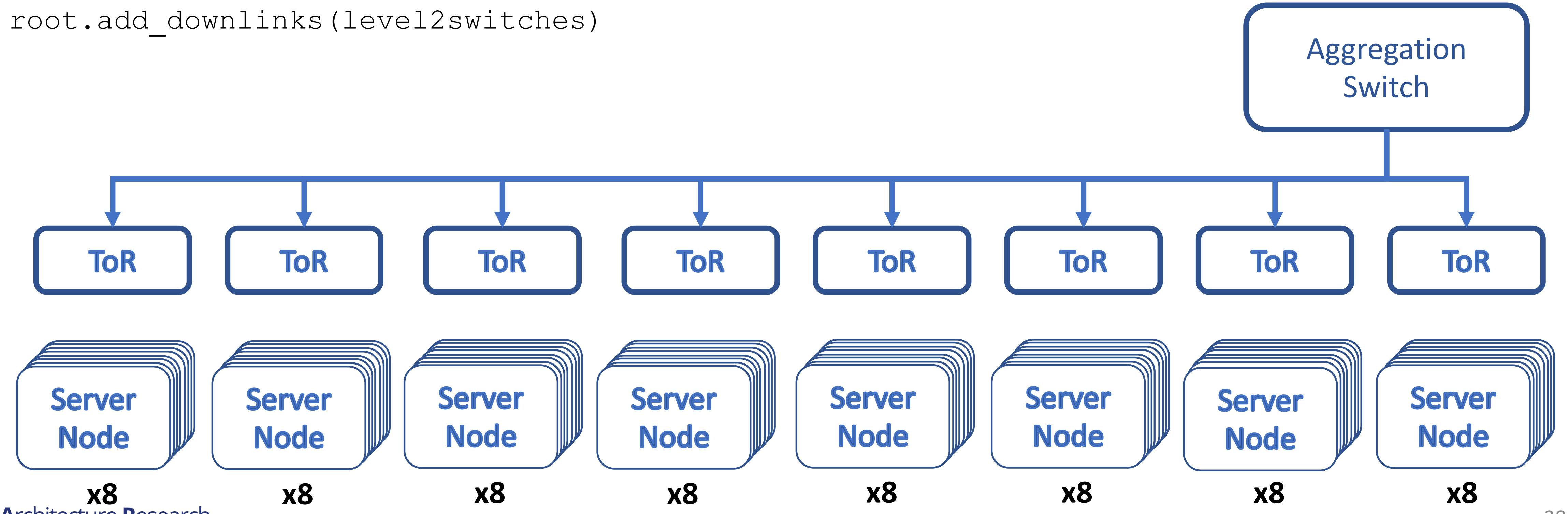
Example (Using multiple f1.16xlarge)

```
def example_64config(self):
    self.roots = [FireSimSwitchNode()]
    level2switches = [FireSimSwitchNode() for x in range(8)]
    servers = [[FireSimServerNode() for y in range(8)] for x in range(8)]
```



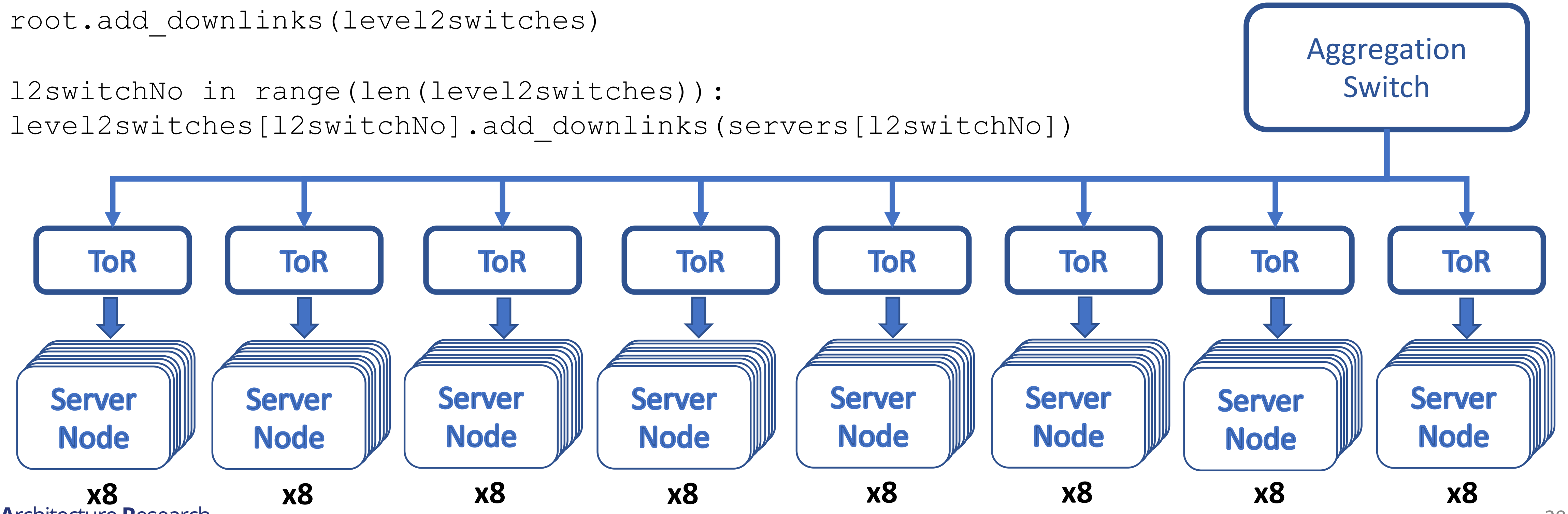
Example (Using multiple f1.16xlarge)

```
def example_64config(self):
    self.roots = [FireSimSwitchNode()]
    level2switches = [FireSimSwitchNode() for x in range(8)]
    servers = [[FireSimServerNode() for y in range(8)] for x in range(8)]

    for root in self.roots:
        root.add_downlinks(level2switches)
```



Example (Using multiple f1.16xlarge)

```
def example_64config(self):
    self.roots = [FireSimSwitchNode()]
    level2switches = [FireSimSwitchNode() for x in range(8)]
    servers = [[FireSimServerNode() for y in range(8)] for x in range(8)]

    for root in self.roots:
        root.add_downlinks(level2switches)

    for l2switchNo in range(len(level2switches)):
        level2switches[l2switchNo].add_downlinks(servers[l2switchNo])
```



Example (Using multiple f1.16xlarge)

- Update `config_runtime.ini` with the appropriate resources and topology
  - Need 8 `f1.16xlarge` instances, since each of them has 8 FPGAs
  - Need one `m4.16xlarge` instance for the aggregation switch

```
[runfarm]
runfarmtag=mainrunfarm

f1_16xlarges=8
m4_16xlarges=1
f1_4xlarges=0
f1_2xlarges=0

runinstancemarket=ondemand
spotinterruptionbehavior=terminate
spotmaxprice=ondemand

[targetconfig]
topology=example_64config
no_net_num_nodes=2
linklatency=6405
switchinglatency=10
netbandwidth=200
profileinterval=-1
```

Network Topologies Using SuperNode

- Supernode packs n server nodes (commonly n=4) onto a single FPGA
  - By generating a pseudo-target design that wraps n server node simulation
  - This is an advanced-user feature, and therefore currently support only a single target design configuration
- Supernode allows simulation of more realistic network topologies, such as a 32-node rack
  - 8 FPGAs on a `f1.16xlarge` instance, with 4 server nodes simulated on each FPGA
- Supernode requires special handling in network topologies

Supernode Example (Using single f1.16xlarge)

- 32-node configuration (1 ToR switches, 32 server nodes)
- Requires a single f1.16xlarge instance in your runfarm

```
def supernode_example_32config(self):
    self.roots = [FireSimSwitchNode()]
    servers = UserTopologies.supernode_flatten([[FireSimSuperNodeServerNode(),
                                                    FireSimDummyServerNode(),
                                                    FireSimDummyServerNode(),
                                                    FireSimDummyServerNode()] for y in range(8)])
    self.roots[0].add_downlinks(servers)
```

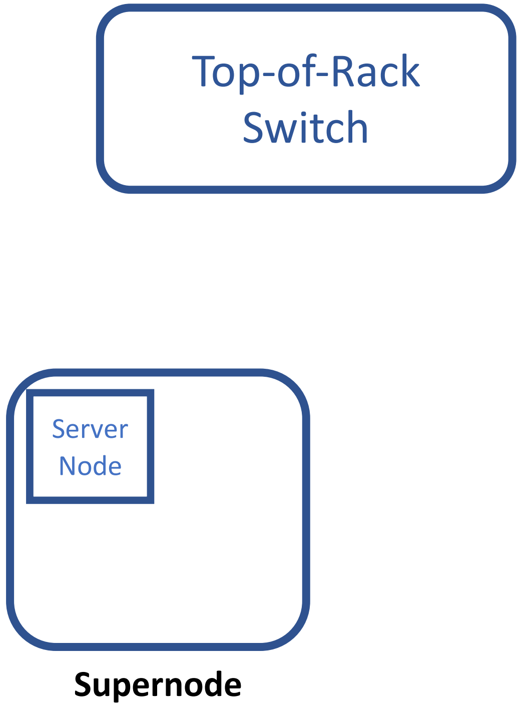
Supernode Example (Using single f1.16xlarge)

```
def supernode_example_32config(self):
    self.roots = [FireSimSwitchNode()]
```

Top-of-Rack Switch

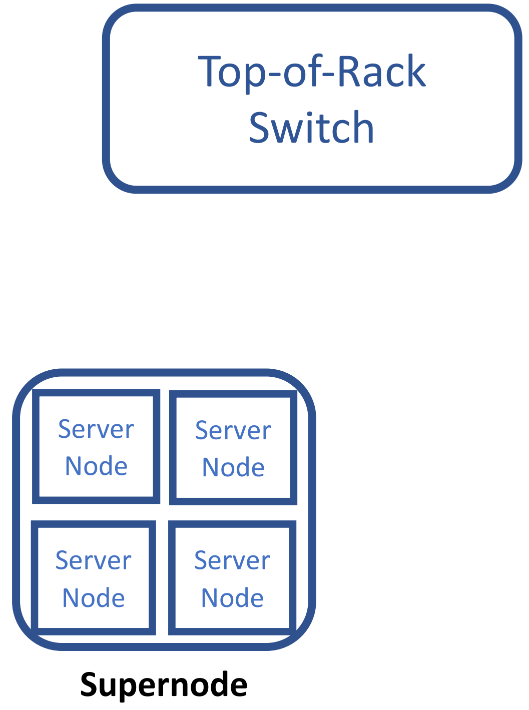
Supernode Example (Using single f1.16xlarge)

```
def supernode_example_32config(self):
    self.roots = [FireSimSwitchNode()]
    servers = UserTopologies.supernode_flatten([[FireSimSuperNodeServerNode(),
```



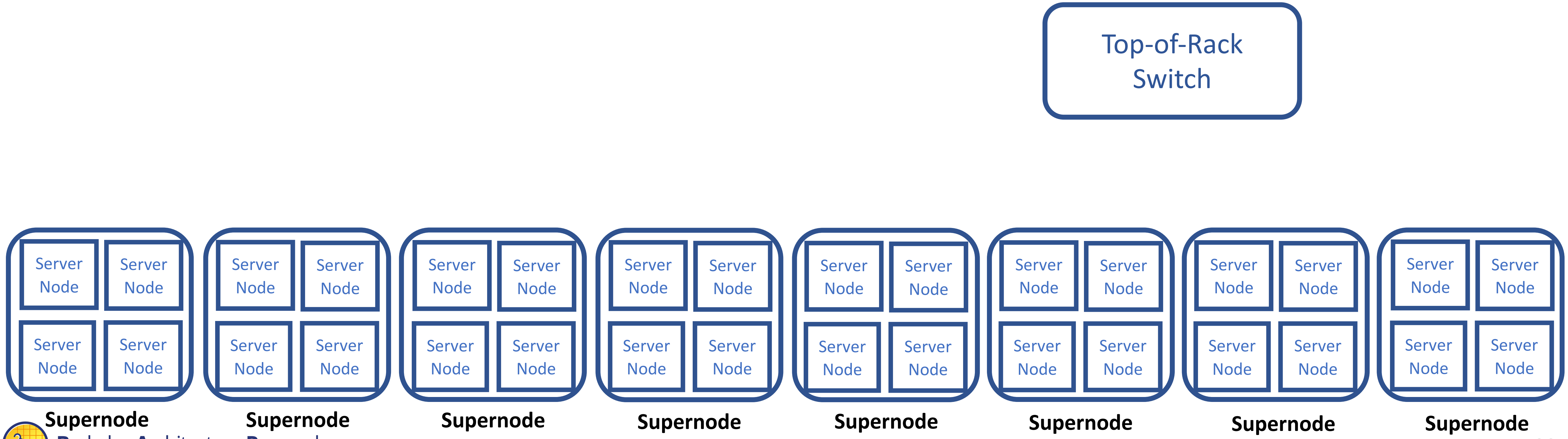
Supernode Example (Using single f1.16xlarge)

```
def supernode_example_32config(self):
    self.roots = [FireSimSwitchNode()]
    servers = UserTopologies.supernode_flatten([[FireSimSuperNodeServerNode(),
                                                 FireSimDummyServerNode(),
                                                 FireSimDummyServerNode(),
                                                 FireSimDummyServerNode()]])
```



Supernode Example (Using single f1.16xlarge)

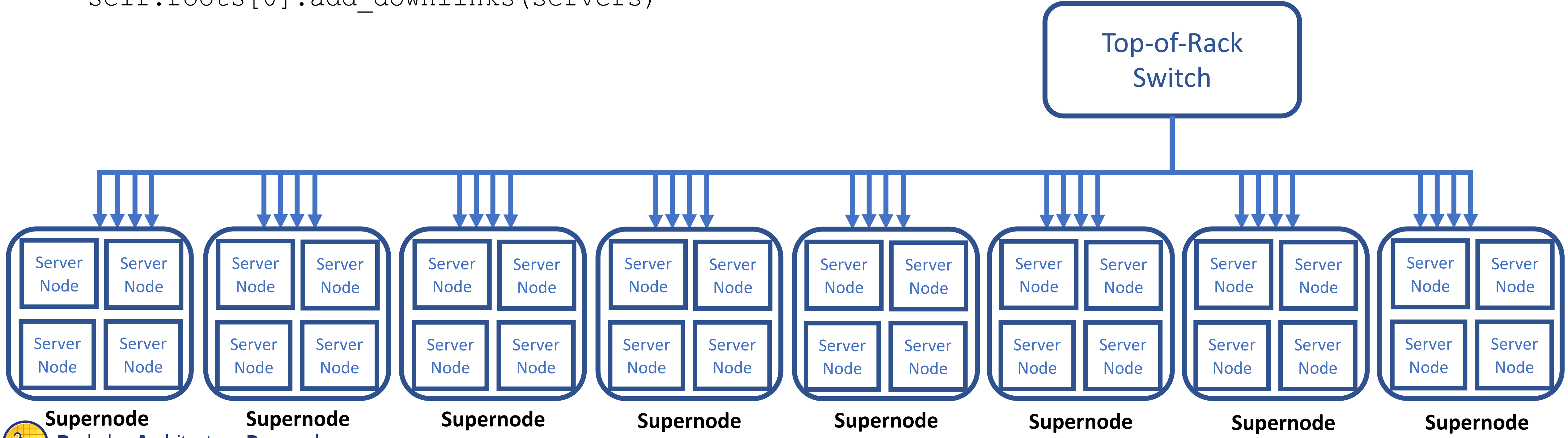
```
def supernode_example_32config(self):
    self.roots = [FireSimSwitchNode()]
    servers = UserTopologies.supernode_flatten([[FireSimSuperNodeServerNode(),
                                                 FireSimDummyServerNode(),
                                                 FireSimDummyServerNode(),
                                                 FireSimDummyServerNode()] for y in range(8)])
```



Berkeley Architecture Research

Supernode Example (Using single f1.16xlarge)

```
def supernode_example_32config(self):
    self.roots = [FireSimSwitchNode()]
    servers = UserTopologies.supernode_flatten([[FireSimSuperNodeServerNode(),
                                                 FireSimDummyServerNode(),
                                                 FireSimDummyServerNode(),
                                                 FireSimDummyServerNode()] for y in range(8)])
    self.roots[0].add_downlinks(servers)
```



Supernode Example (Using single f1.16xlarge)

- Update `config_runtime.ini` with the appropriate resources and topology
  - One `f1.16xlarge` instance is sufficient for a 32-node supernode simulation since it includes 8 FPGAs
  - Supernode currently has a restricted set of target design, and is therefore considered an advanced-user feature

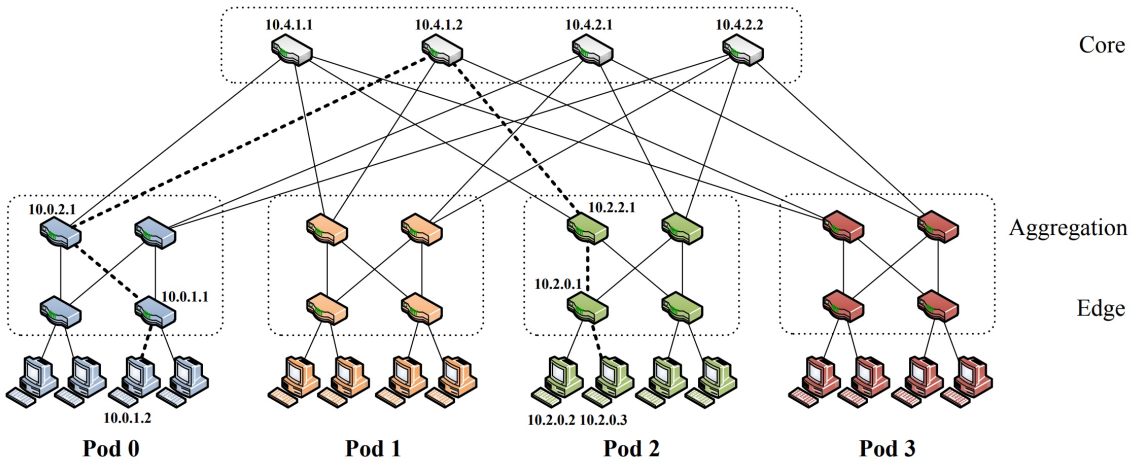
```
[runfarm]
runfarmtag=mainrunfarm

f1_16xlarges=1
m4_16xlarges=0
f1_4xlarges=0
f1_2xlarges=0

runinstancemarket=ondemand
spotinterruptionbehavior=terminate
spotmaxprice=ondemand

[targetconfig]
topology=supernode_example_32config
no_net_num_nodes=2
linklatency=6405
switchinglatency=10
netbandwidth=200
profileinterval=-1
```

Complex Topology Example

- The basic network topology primitives should allow any graph-based topology
- The `$FDIR/deploy/runtools/user_topology.py` file include multiple example of complex topologies such as fat-tree, clos, and nodes with multiple links.

Fat Tree Example

```
def fat_tree_4ary(self):
    coreswitches = [FireSimSwitchNode() for x in range(4)]
    self.roots = coreswitches
    aggrswitches = [FireSimSwitchNode() for x in range(8)]
    edgeswitches = [FireSimSwitchNode() for x in range(8)]
    servers = [FireSimServerNode() for x in range(16)]
    for switchno in range(len(coreswitches)):
       core = coreswitches[switchno]
       base = 0 if switchno < 2 else 1
       dls = range(base, 8, 2)
       dls = map(lambda x: aggrswitches[x], dls)
       core.add_downlinks(dls)
    for switchbaseno in range(0, len(aggrswitches), 2):
       switchno = switchbaseno + 0
       aggr = aggrswitches[switchno]
       aggr.add_downlinks([edgeswitches[switchno], edgeswitches[switchno+1]])
       switchno = switchbaseno + 1
       aggr = aggrswitches[switchno]
       aggr.add_downlinks([edgeswitches[switchno-1], edgeswitches[switchno]])
   for edgeno in range(len(edgeswitches)):
       edgeswitches[edgeno].add_downlinks([servers[edgeno*2], servers[edgeno*2+1]])
```



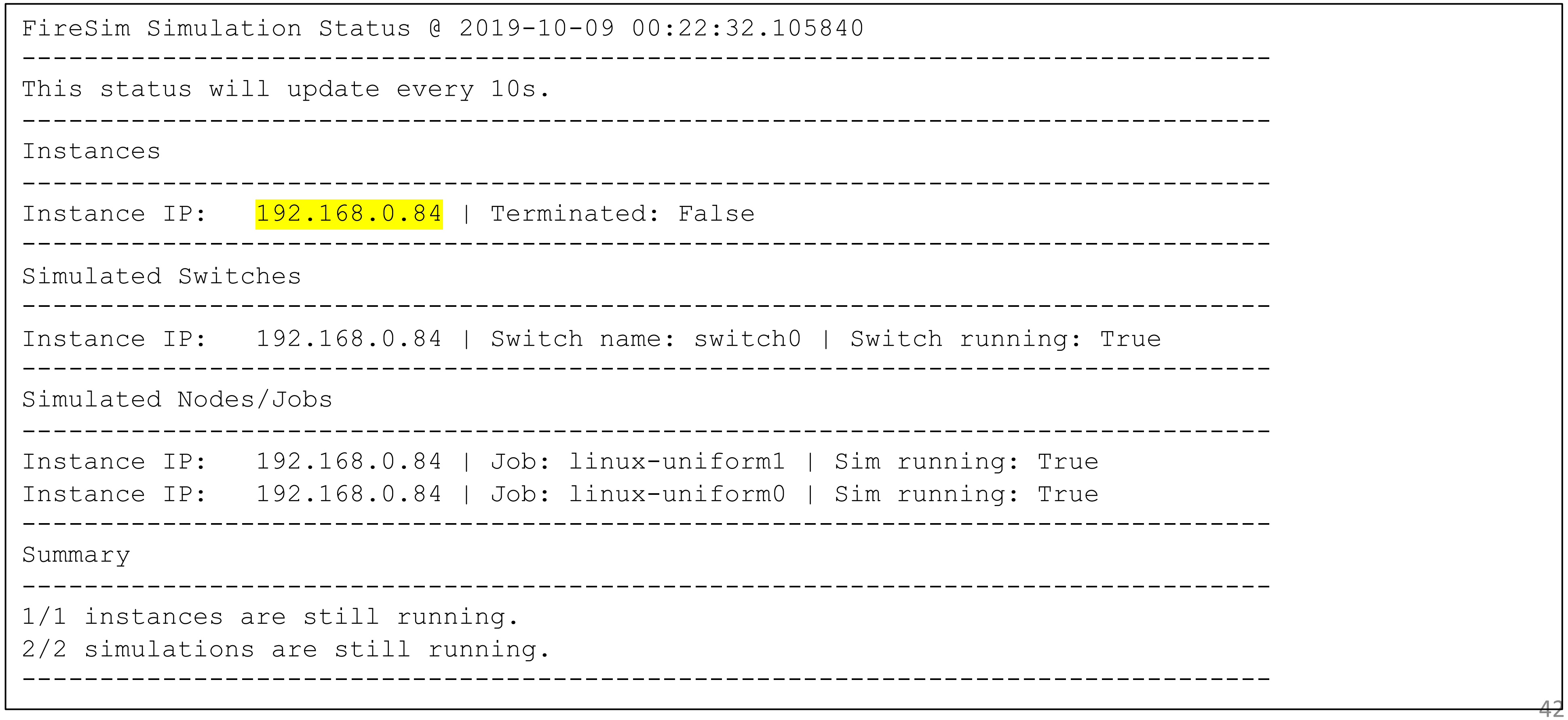
From: A Scalable, Commodity Data Center Network Architecture, Al-Fares et al. SIGCOMM 2008

Back to our hand-on experiment

Heterogenous Topology Example: Hands-on

- Find the IP address of your runfarm in the manager monitor

You will have a different IP address here



```
FireSim Simulation Status @ 2019-10-09 00:22:32.105840
--------------------------------------------------------------------------------
This status will update every 10s.
--------------------------------------------------------------------------------
Instances
--------------------------------------------------------------------------------
Instance IP:   192.168.0.84 | Terminated: False
--------------------------------------------------------------------------------
Simulated Switches
--------------------------------------------------------------------------------
Instance IP:   192.168.0.84 | Switch name: switch0 | Switch running: True
--------------------------------------------------------------------------------
Simulated Nodes/Jobs
--------------------------------------------------------------------------------
Instance IP:   192.168.0.84 | Job: linux-uniform1 | Sim running: True
Instance IP:   192.168.0.84 | Job: linux-uniform0 | Sim running: True
--------------------------------------------------------------------------------
Summary
--------------------------------------------------------------------------------
1/1 instances are still running.
2/2 simulations are still running.
--------------------------------------------------------------------------------
```

Heterogenous Topology Example: Hands-on

- On the *manager* instance, ssh into your runfarm instance (you will have a different IP here)

```
$ ssh 192.168.0.84
```

Heterogenous Topology Example: Hands-on

- Attach to the console of the first simulated node using

```
$ screen -r fsim0
```

- Log in as “root” with password “firesim” (password does not echo)

```
Starting dropbear sshd: OK
launching firesim workload run/command
firesim workload run/command done

Welcome to Buildroot
buildroot login: root
Password:
#
```

Heterogenous Topology Example: Hands-on

- Within the first simulated node, run `cat /proc/cpuinfo` to check which processor we have on this node

```
# cat /proc/cpuinfo
processor       : 0
hart            : 0
isa             : rv64imafdc
mmu             : sv39
uarch           : ucb-bar,boom0
```

- Within the first simulated node, create a text file with a message (you can write any message you want):

```
# echo "Having fun at the firesim-chipyard tutorial" > message0.txt
```

Heterogenous Topology Example: Hands-on

- Send a message from the first simulated node to the second node using `scp` to IP 172.16.0.3 (reminder, password is firesim) :

```
# scp message0.txt root@172.16.0.3:/root/

Host '172.16.0.3' is not in the trusted hosts file.
(ecdsa-sha2-nistp256 fingerprint sha1!!
37:19:89:0c:9a:04:08:22:46:2e:f3:99:99:04:cb:09:04:a0:cd:55)
Do you want to continue connecting? (y/n) yes
root@172.16.0.3's password:
message0.txt                                  100%   44     0.0KB/s   00:00
#
```

- Detach from the console of first simulated node (`CRTL+a d`)

Berkeley Architecture Research

Heterogenous Topology Example: Hands-on

- Attach to the console of the second simulated node using

```
$ screen -r fsim1
```

- Log in as “root” with password “firesim” (password does not echo)

```
Starting dropbear sshd: OK
launching firesim workload run/command
firesim workload run/command done

Welcome to Buildroot
buildroot login: root
Password:
#
```

Berkeley Architecture Research

Heterogenous Topology Example: Hands-on

- Within the second simulated node, run `cat /proc/cpuinfo` to see that this is indeed a heterogenous network configuration

```
# cat /proc/cpuinfo
processor             : 0
hart                  : 0
isa                   : rv64imafdc
mmu                   : sv39
uarch                 : sifive,rocket0
```

Open the message that was sent by the first simulated node using `cat`:

```
# cat message0.txt
Having fun at the firesim-chipyard tutorial
```

Heterogenous Topology Example: Hands-on

- Power off the interactive simulated node (this takes 1 minute)

```
# poweroff -f
```

```
Stopping dropbear sshd: OK
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using
127.0.1.1. Set the 'ServerName' directive globally to suppress this message
Stopping network: OK
Saving random seed... done.
Stopping mdev... stopped process in pidfile '/var/run/mdev.pid' (pid 103)
OK
Stopping klogd: OK
Stopping syslogd: OK
umount: can't remount /dev/iceblk read-only
umount: none busy - remounted read-only
The system is going down NOW!
Sent SIGTERM to all processes
logout
```

Heterogenous Topology Example: Hands-on

- Back in the manager (after the simulated node powered-off)

```
Teardown required, manually tearing down...
[192.168.0.84] Executing task 'kill_switch_wrapper'
[192.168.0.84] Killing switch simulation for switchslot: 0.
[192.168.0.84] Executing task 'kill_simulation_wrapper'
[192.168.0.84] Killing FPGA simulation for slot: 0.
[192.168.0.84] Killing FPGA simulation for slot: 1.
[192.168.0.84] Executing task 'screens'
Confirming exit...
[192.168.0.84] Executing task 'monitor_jobs_wrapper'
[192.168.0.84] Slot 0 completed! copying results.
[192.168.0.84] Slot 1 completed! copying results.
[192.168.0.84] Killing switch simulation for switchslot: 0.
FireSim Simulation Exited Successfully. See results in:
/home/centos/chipyard-tutorial/sims/firesim/deploy/results-workload/2019-10-09--00-22-20-linux-
uniform/
The full log of this run is:
/home/centos/chipyard-tutorial/sims/firesim/deploy/logs/2019-10-09--00-22-20-runworkload-
QATGI5DOAIQBTAEY.log
```

Heterogenous Topology Example: Hands-on

Back in your manager instance, don't forget to terminate your runfarm (otherwise, we are going to pay for a lot of FPGA time)

```
$ firesim terminaterunfarm
```

Type yes at the prompt to confirm

Summary

- Writing network topologies
  - Basic network topologies
  - Heterogenous network topologies
  - Supernode network topologies
  - Custom network topologies
- Choosing network parameters
- Run-farm configuration for scale-out simulations
- Running a simulation
  - Hands-on experience – it's easy!

Check out https://docs.fires.im/ for more usage details