www. chameleoncloud.org

EXPERIMENTS IN THE EDGE TO CLOUD CONTINUUM

Kate Keahey

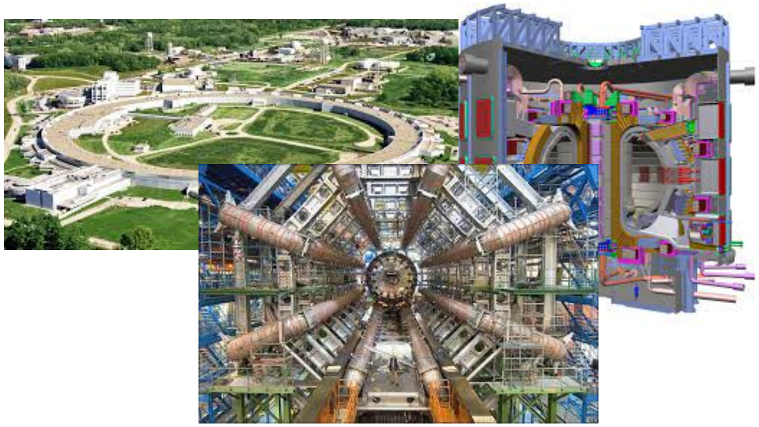
Mathematics and CS Division, Argonne National Laboratory

CASE, University of Chicago

keahey@anl.gov

September 8, 2021
IEEE Cluster

SCIENTIFIC INSTRUMENTS

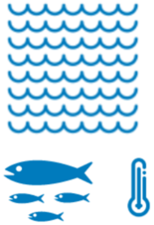


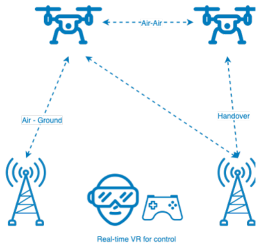
What scientific instruments do Computer Scientists need?

Innovative and diverse hardware, breadth of deployment, freedom to touch and measure every aspect of configuration and behavior.

Constantly evolving!

THE EMERGENCE OF EDGE





Challenges in connectivity, scale, security, dynamicity, resilience, data and information workflows, management – and many others!

CHAMELEON IN A NUTSHELL

- We like to change: a testbed that adapts itself to your experimental needs
  - Deep reconfigurability (bare metal) and isolation – but also a small KVM cloud
  - power on/off, reboot, custom kernel, serial console access, etc.
- Balance: large-scale versus diverse hardware
  - Large-scale: ~large homogenous partition (~15,000 cores), ~6 PB of storage distributed over 2 sites (UC, TACC) connected with 100G network
  - CHI-in-a-Box sites at Northwestern, IIT, and other places
  - Diverse: ARMs, Atoms, FPGAs, GPUs, Corsa switches, etc.
- Cloud++: CHameleon Infrastructure (CHI) via mainstream cloud tech
  - Powered by OpenStack with bare metal reconfiguration (Ironic) + "special sauce"
  - Blazar contribution recognized as official OpenStack component
- Repeatability and sharing
  - Packaging, sharing, discovering, and publishing experiments

openstack CLOUD SOFTWARE

OPEN TESTBED – BY THE NUMBERS

300+
Papers published

45
Countries

160+
Institutions

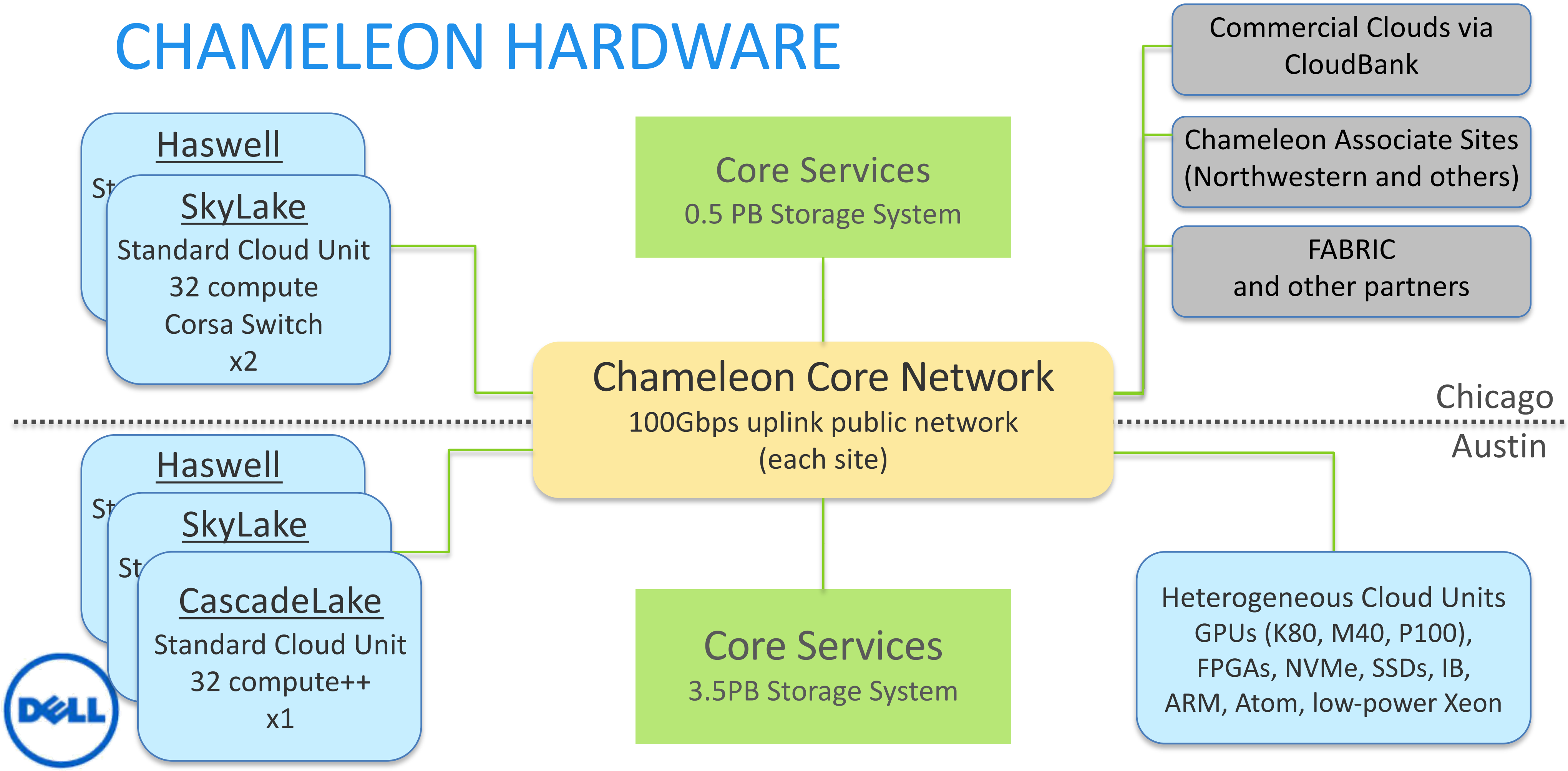
Over
5,800
Users

700+
Projects

6+
Years Old

and 3 more
years to grow!

CHAMELEON HARDWARE



Commercial Clouds via CloudBank

Chameleon Associate Sites (Northwestern and others)

FABRIC and other partners

Haswell

St

SkyLake

Standard Cloud Unit

32 compute

Corsa Switch

x2

Core Services

0.5 PB Storage System

Chameleon Core Network

100Gbps uplink public network (each site)

Chicago

Austin

Haswell

St

SkyLake

St

CascadeLake

Standard Cloud Unit

32 compute++

x1

Core Services

3.5PB Storage System

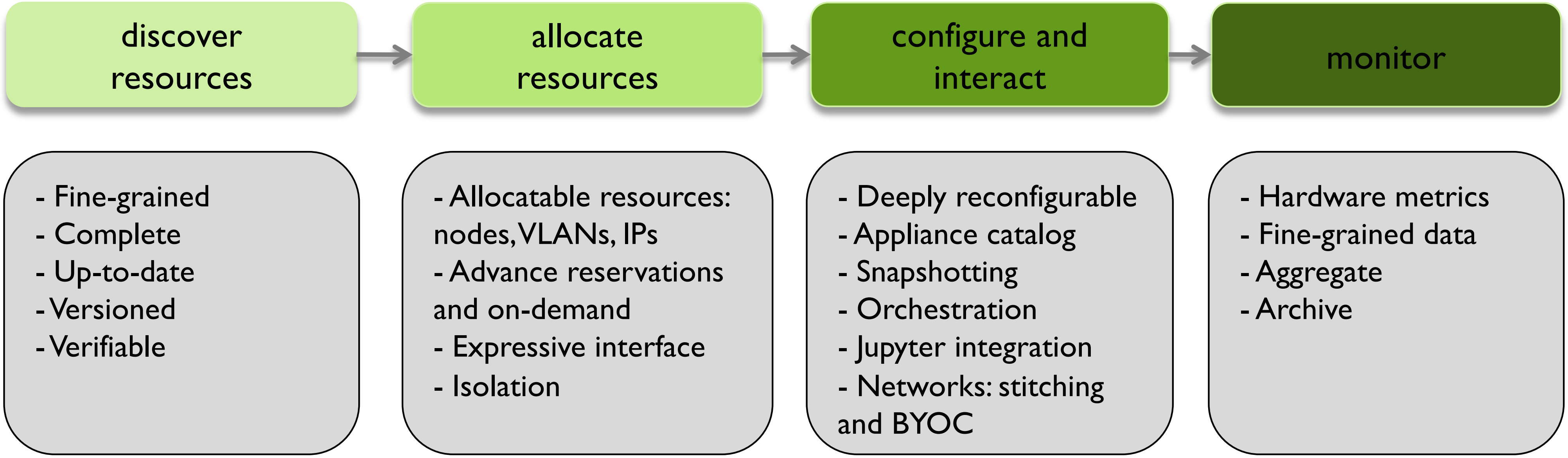
Heterogeneous Cloud Units

GPUs (K80, M40, P100), FPGAs, NVMe, SSDs, IB, ARM, Atom, low-power Xeon

CHAMELEON HARDWARE (DETAILS)

- “Start with large-scale homogenous partition”
  - 12 Haswell racks, each with 42 Dell R630 compute servers with dual-socket Intel Haswell processors (24 cores) & 128GB RAM and 4 Dell FX2 storage servers with 16 2TB drives each; Force10 s6000 OpenFlow-enabled switches 10Gb to hosts, 40Gb uplinks to Chameleon core network
  - 3 SkyLake racks (32 nodes each); Corsa (DP2400 & DP2200), 100Gb uplinks to core network
  - CascadeLake rack (32 nodes), 100Gb ulpinks to Chameleon core network
  - Allocations can be an entire rack, multiple racks, nodes within a single rack or across racks (e.g., storage servers across racks forming a Hadoop cluster)
- Shared infrastructure
  - 3.6 (TACC) + 0.5 (UC) PB global storage, 100Gb Internet connection between sites
- “Graft on heterogeneous features”
  - Infiniband with SR-IOV support, High-mem, NVMe, SSDs, P100 GPUs (total of 22 nodes), RTX GPUs (40 nodes), FPGAs (4 nodes)
  - ARM microservers (24) and Atom microservers (8), low-power Xeons (8)
- Coming in Phase 3: upgrading Haswells to CascadeLake and IceLake + AMD, new GPUs and FPGAs, more and newer IB fabric, variety of storage options for disaggregated hardware experiments, composable hardware (LiQid), networking (P4, integration with FABRIC), IoT devices -- and strategic reserve

CHI EXPERIMENTAL WORKFLOW



discover resources

- Fine-grained
- Complete
- Up-to-date
- Versioned
- Verifiable

allocate resources

- Allocatable resources: nodes, VLANs, IPs
- Advance reservations and on-demand
- Expressive interface
- Isolation

configure and interact

- Deeply reconfigurable
- Appliance catalog
- Snapshotting
- Orchestration
- Jupyter integration
- Networks: stitching and BYOC

monitor

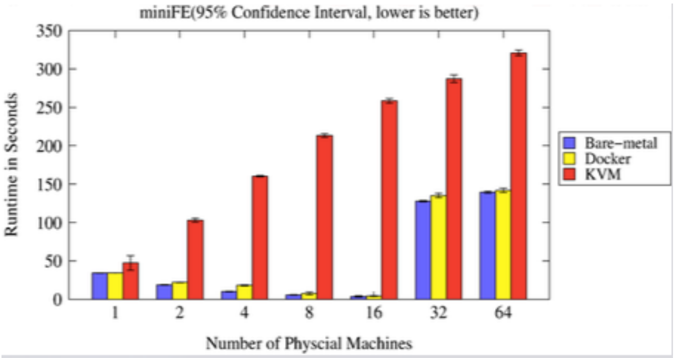
- Hardware metrics
- Fine-grained data
- Aggregate
- Archive

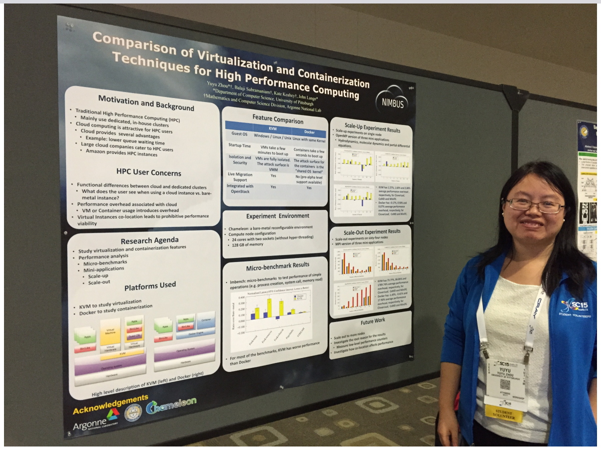
Authentication via federated identity, accessed via GUI, CLI and python/Jupyter

Paper: "Lessons Learned from the Chameleon Testbed", USENIX ATC 2020

VIRTUALIZATION OR CONTAINERIZATION?

- Yuyu Zhou, University of Pittsburgh
- Research: lightweight virtualization
- Testbed requirements:
  - Bare metal reconfiguration, isolation, and serial console access
  - The ability to “save your work”
  - Support for large scale experiments
  - Up-to-date hardware

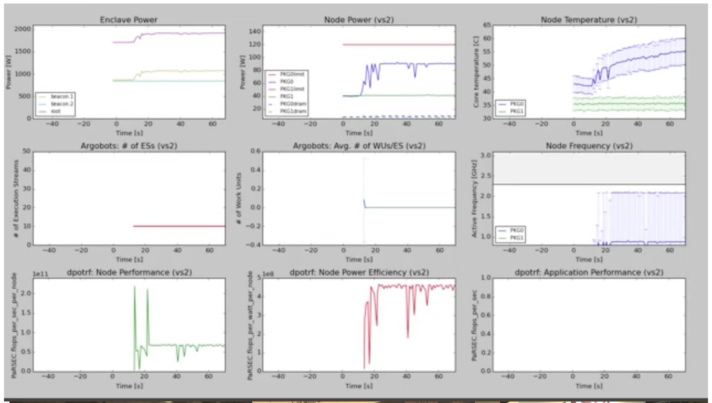


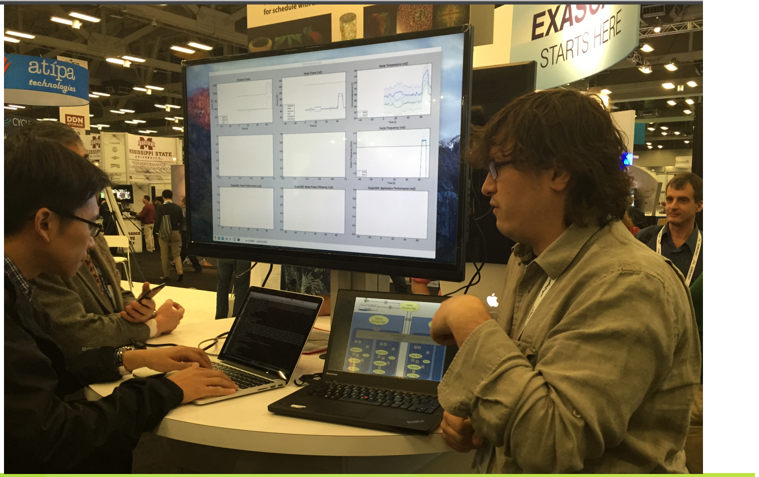

Comparison of Virtualization and Containerization Techniques for High Performance Computing
NIMBUS
Motivation and Background
HPC User Concerns
Research Agenda
Platforms Used
Feature Comparison
Experiment Environment
Micro-benchmark Results
Scale-Up Experiment Results
Scale-Out Experiment Results
Future Work
Acknowledgements
Argonne
Chameleon

SC15 Poster: “Comparison of Virtualization and Containerization Techniques for HPC”

EXASCALE OPERATING SYSTEMS

- Swann Perarnau, ANL
- Research: exascale operating systems
- Testbed requirements:
  - Bare metal reconfiguration
  - Boot from custom kernel with different kernel parameters
  - Fast reconfiguration, many different images, kernels, parameters
  - Hardware: accurate information and control over changes, performance counters, many cores
  - Access to same infrastructure for multiple collaborators





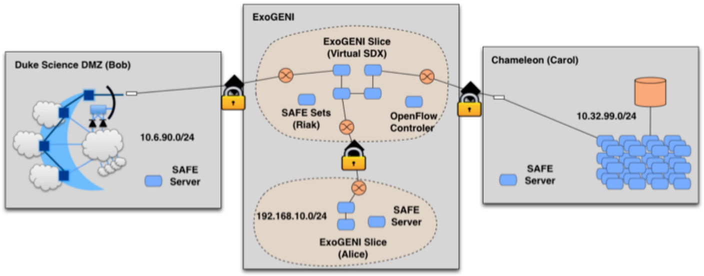
HPPAC'16 paper: "Systemwide Power Management with Argo"

CLASSIFYING CYBERSECURITY ATTACKS

- Jessie Walker & team, University of Arkansas at Pine Bluff (UAPB)
- Research: modeling and visualizing multi-stage intrusion attacks (MAS)
- Testbed requirements:
  - Easy to use OpenStack installation
  - A selection of pre-configured images
  - Access to the same infrastructure for multiple collaborators

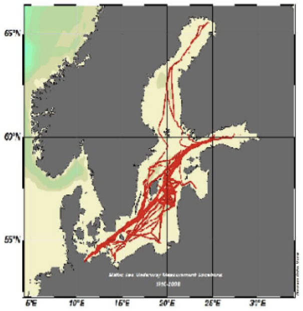
CREATING DYNAMIC SUPERFACILITIES

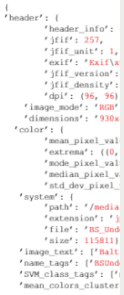
- NSF CICI SAFE, Paul Ruth, RENCI-UNC Chapel Hill
- Creating trusted facilities
  - Automating trusted facility creation
  - Virtual Software Defined Exchange (SDX)
  - Secure Authorization for Federated Environments (SAFE)
- Testbed requirements
  - Creation of dynamic VLANs and wide-area circuits
  - Support for network stitching
  - Managing complex deployments

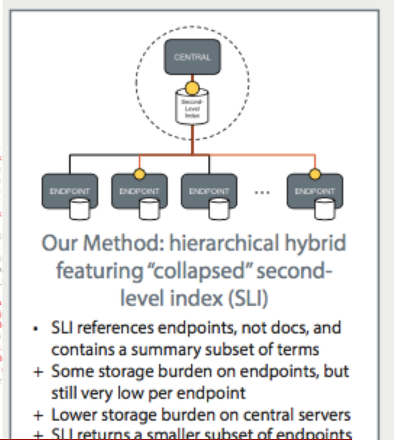


DATA SCIENCE RESEARCH

- ACM Student Research Competition semi-finalists:
  - Blue Keleher, University of Maryland
  - Emily Herron, Mercer University
- Searching and image extraction in research repositories
- Testbed requirements:
  - Access to distributed storage in various configurations
  - State of the art GPUs
  - Easy to use appliances and orchestration




```
{
'header': {
        'header_info':
        'jfif': 257,
        'jfif_unit': 1,
        'exif': 'Exif\x
        'jfif_version':
        'jfif_density':
        'dpi': (96, 96)
    'image_mode': 'RGB'
    'dimensions': '930x
  'color': {
        'mean_pixel_val
        'extrema': ((0,
        'mode_pixel_val
        'median_pixel_v
        'std_dev_pixel_
   'system': {
        'path': '/media
        'extension': 'j
        'file': 'BS_Und
        'size': 115811}
   'image_text': ['Balt
   'name_tags': ['BSUnd
   'SVM_class_tags': ['
   'mean_colors_cluster
```



Our Method: hierarchical hybrid featuring "collapsed" second-level index (SLI)

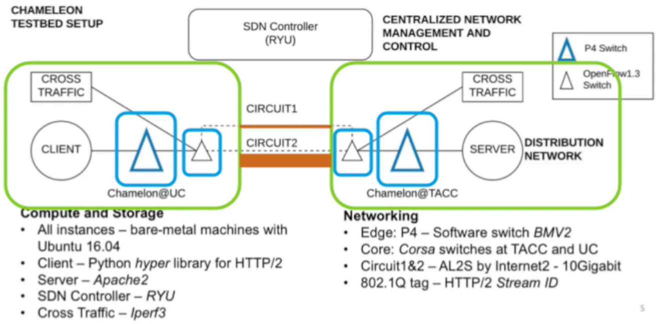
- SLI references endpoints, not docs, and contains a summary subset of terms

+ Some storage burden on endpoints, but still very low per endpoint
+ Lower storage burden on central servers
+ SLI returns a smaller subset of endpoints

Chameleon www. chameleoncloud.org

ADAPTIVE BITRATE VIDEO STREAMING

- Divyashri Bhat, UMass Amherst
- Research: application header based traffic engineering using P4
- Testbed requirements:
  - Distributed testbed facility
  - BYOC – the ability to write an SDN controller specific to the experiment
  - Multiple connections between distributed sites
- https://vimeo.com/297210055



Compute and Storage

- All instances – bare-metal machines with Ubuntu 16.04
- Client – Python *hyper* library for HTTP/2
- Server – *Apache2*
- SDN Controller – *RYU*
- Cross Traffic – *Iperf3*

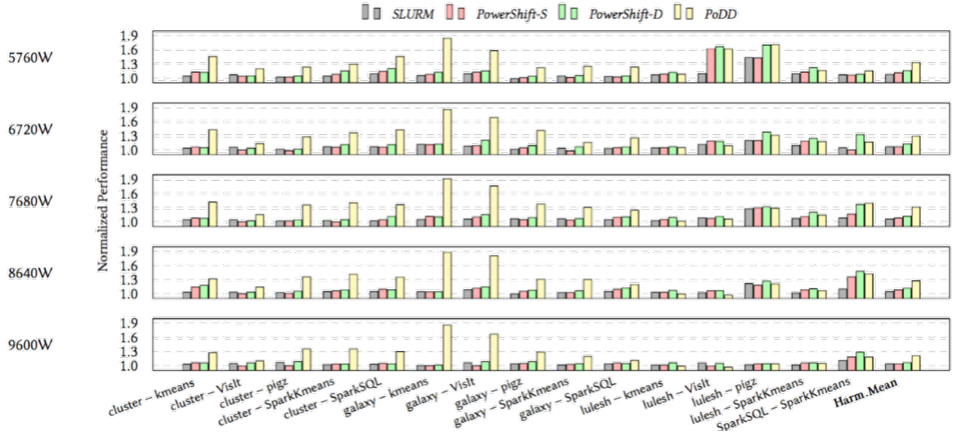
Networking

- Edge: P4 – Software switch *BMV2*
- Core: *Corsa* switches at TACC and UC
- Circuit1&2 – AL2S by Internet2 - 10Gigabit
- 802.1Q tag – HTTP/2 *Stream ID*

LCN'18: "Application-based QoS support with P4 and OpenFlow"

POWER CAPPING

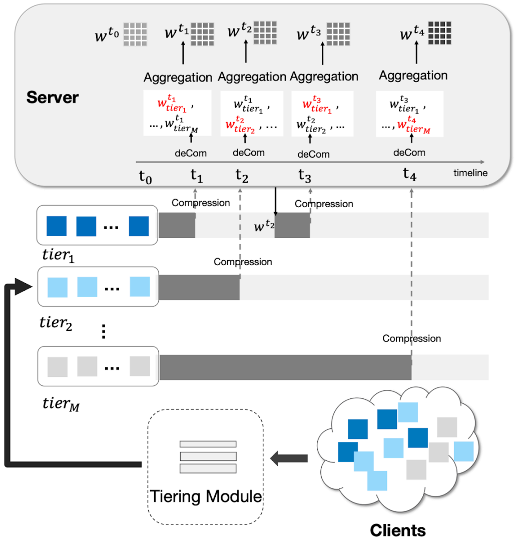
- Harper Zhang, University of Chicago
- Research: hierarchical, distributed, dynamic power management system for dependent applications
- Testbed requirements:
  - Support for large-scale experiments
  - Complex appliances and orchestration (NFS appliance)
  - RAPL/power management interface
- Finalist for SC19 Best Paper and Best Student Paper
- Talk information at bit.ly/SC19PoDD



SC'19: "PoDD: Power-Capping Dependent Distributed Applications"

FEDERATED LEARNING

- Zheng Chai and Yue Cheng, George Mason University
- Research: federated learning
- Testbed requirements:
  - Bare metal, ability to record network traffic precisely
  - Support for large-scale and diverse hardware
  - Powerful nodes with large memory



Paper: "FedAT:A Communication-Efficient Federated Learning Method with Asynchronous Tiers under Non-IID Data", October 2020

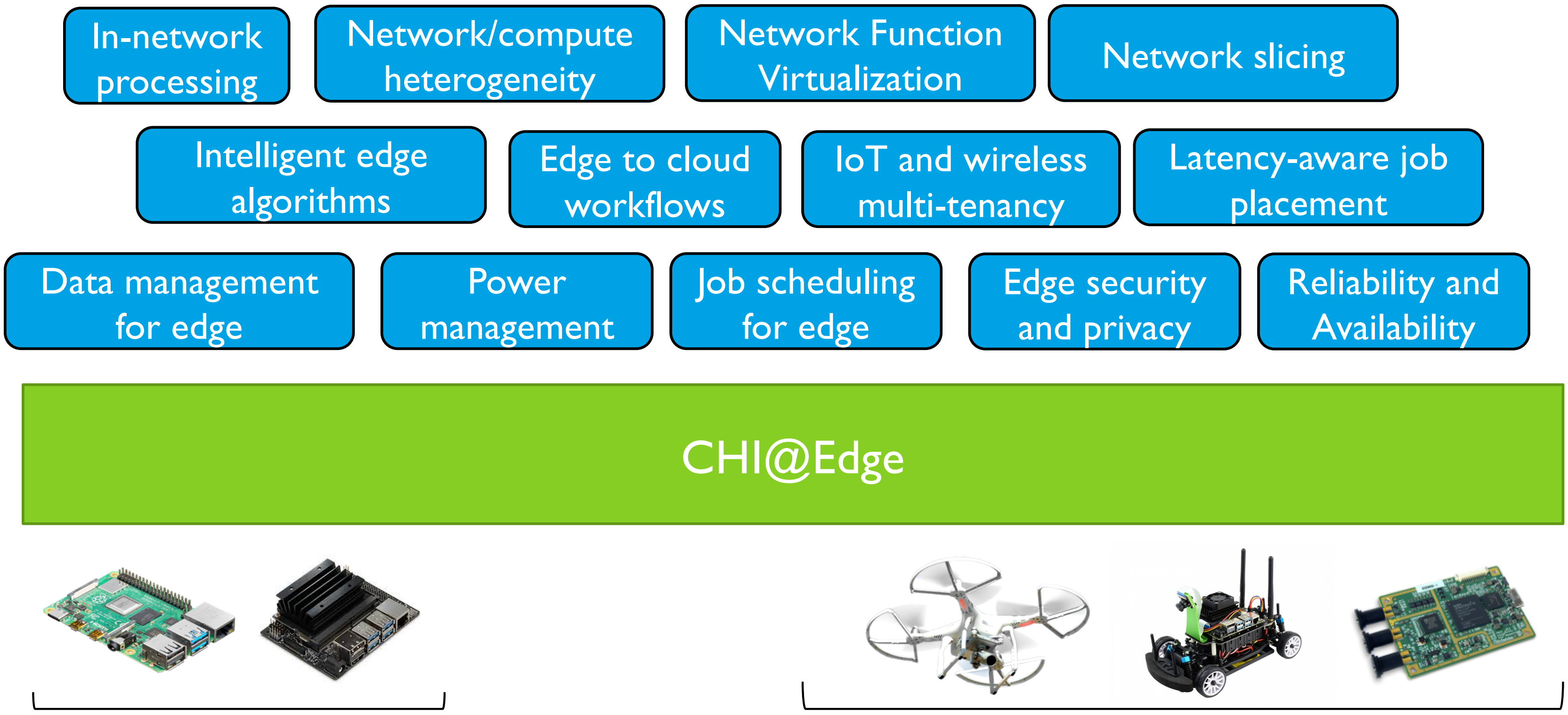
WHAT DOES AN EDGE TESTBED LOOK LIKE?

A lot like a cloud!
All the features we know
and love!

Not at all like a cloud!
Not server-class!
IoT: cameras, actuators, SDRs!
Location, location, location!
And many other challenges!

- CHI@Edge: all the features you know and love plus
  - Reconfiguration via container deployment
  - Support for peripherals based on an extensible plug-in model
  - Mixed ownership model via an SDK with devices available through virtual site(s)
  - Still working on defining the capabilities: Chameleon@Edge community workshop on 09/09, see: https://chameleoncloud.org/chiedge-community-workshop/

WHAT DOES AN EDGE TESTBED LOOK LIKE?



In-network processing

Network/compute heterogeneity

Network Function Virtualization

Network slicing

Intelligent edge algorithms

Edge to cloud workflows

IoT and wireless multi-tenancy

Latency-aware job placement

Data management for edge

Power management

Job scheduling for edge

Edge security and privacy

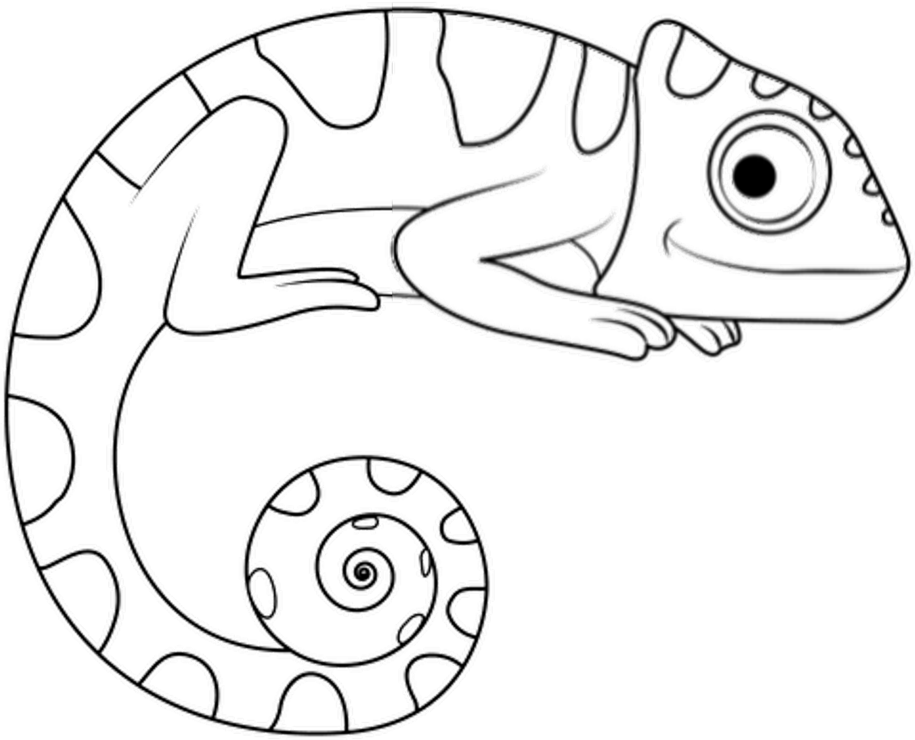
Reliability and Availability

CHI@Edge

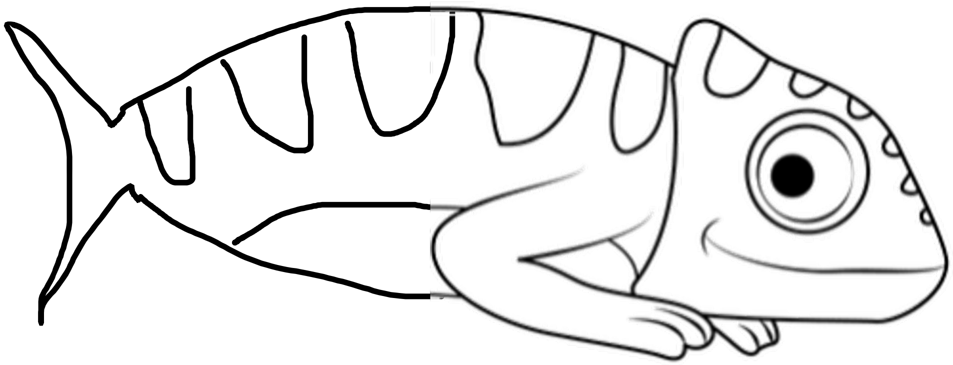
chameleon-owned devices

user-owned devices

HOW DOES IT WORK?



HOW DOES IT WORK?



OpenStack adaptation: reconfiguration via container deployment, invalidating datacenter assumptions

OpenStack interfaces: advance reservations, single-tenant isolation, isolated networking, IP assignment, snapshotting

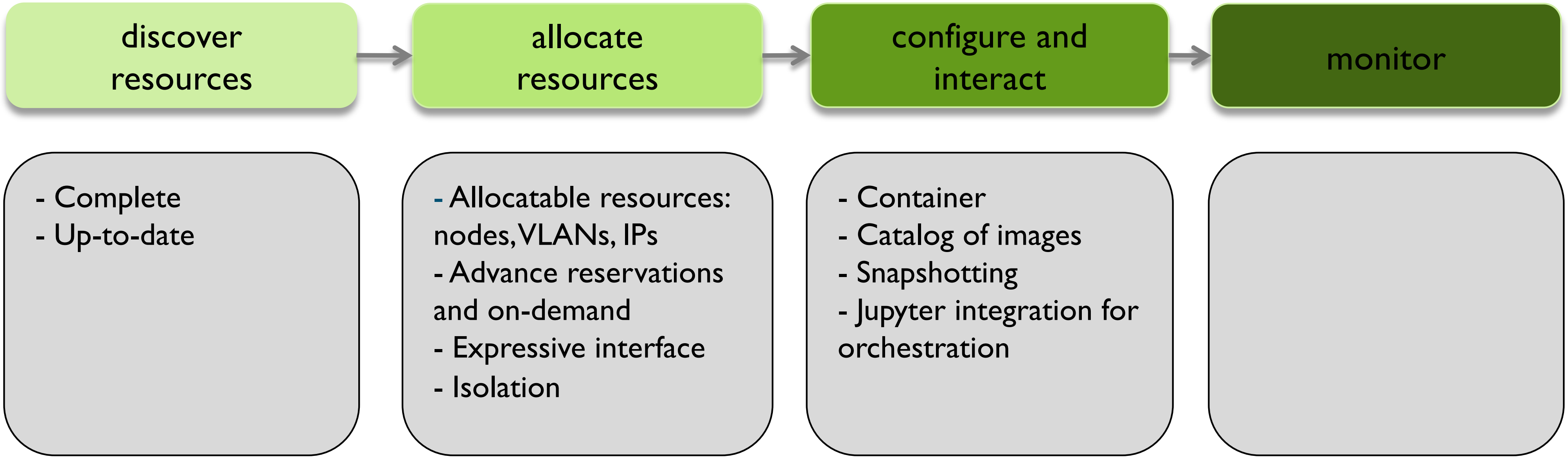
Existing user interface: identity federation, python-chi, integration with Jupyter, etc.

BUILDING CHI@EDGE

From this…

…to this!

CHI@EDGE EXPERIMENTAL WORKFLOW (PREVIEW)



discover resources

- Complete
- Up-to-date

allocate resources

- Allocatable resources: nodes, VLANs, IPs
- Advance reservations and on-demand
- Expressive interface
- Isolation

configure and interact

- Container
- Catalog of images
- Snapshotting
- Jupyter integration for orchestration

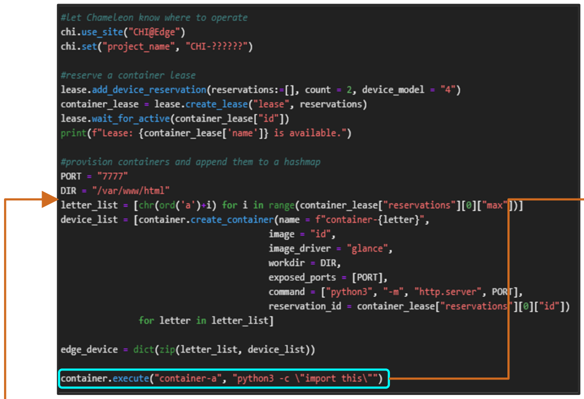
monitor

Authentication via federated identity, accessed via GUI, CLI and python/Jupyter

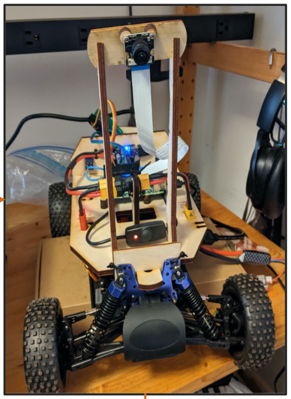
AUTONOMOUS CARS WITH CHI@EDGE

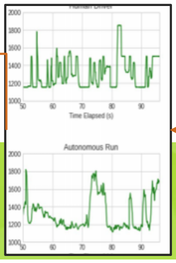
- Goal:
  - Teach machine learning and systems concepts using remote autonomous cars
- Challenges:
  - Control the cars remotely: manual workflows require lots of teacher effort
  - Iterate on code while learning and exploring
  - Collect, store, and process large datasets
- CHI@Edge:
  - Car reservations
  - Access through JupyterHub
  - Provides consistent network connection
  - Deploy code and collect results with repeatable workflows

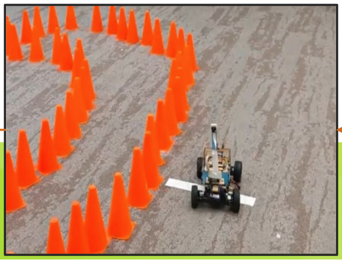
Rick Anderson
Virtual Worlds, Director
Rutgers University



```
#let Chameleon know where to operate
chi.use_site("CHI@Edge")
chi.set("project_name", "CHI-??????")

#reserve a container lease
lease.add_device_reservation(reservations:=[], count = 2, device_model = "4")
container_lease = lease.create_lease("lease", reservations)
lease.wait_for_active(container_lease["id"])
print(f"Lease: {container_lease['name']} is available.")

#provision containers and append them to a hashmap
PORT = "7777"
DIR = "/var/www/html"
letter_list = [chr(ord('a')+i) for i in range(container_lease["reservations"][0]["max"])]
device_list = [container.create_container(name = f"container-{letter}",
                                          image = "id",
                                          image_driver = "glance",
                                          workdir = DIR,
                                          exposed_ports = [PORT],
                                          command = ["python3", "-m", "http.server", PORT],
                                          reservation_id = container_lease["reservations"][0]["id"])
               for letter in letter_list]

edge_device = dict(zip(letter_list, device_list))

container.execute("container-a", "python3 -c \"import this\"")
```



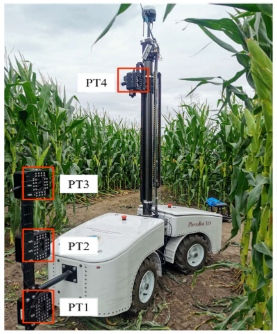




ARA: WIRELESS LIVING LAB FOR SMART & CONNECTED RURAL COMMUNITIES

- ARA objectives
  - Enable research to achieve a factor of 10+ reduction in broadband cost and make rural broadband as affordable as urban broadband!
  - Support broadband use cases for rural communities to industries
- ARA wireless living lab
  - Deploy advanced wireless platforms in Central Iowa (>600 square miles); capture systems and application and community contexts of rural broadband
  - Mainstream open-source platforms for living lab management and experimentation: OpenStack, CHI-in-a-Box & CHI@Edge, ONF (SD-RAN, SD-CORE, ONOS), srsRAN, OpenAirInterface etc
  - CHI@Edge: collaborating on spectrum reservations for management of wireless networks

Hongwei Zhang, ARA PI
Iowa State University

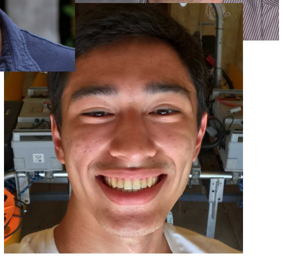


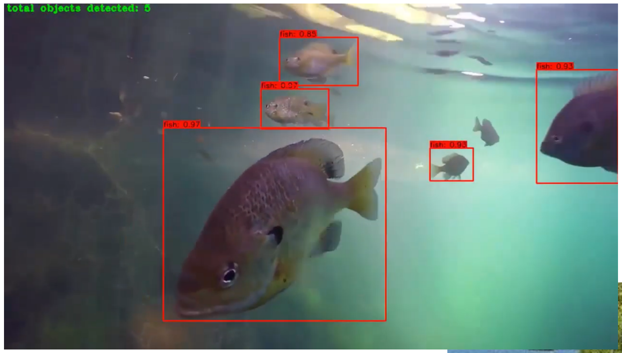
Location and Interior view of ISU Beef Nutrition Research Farm

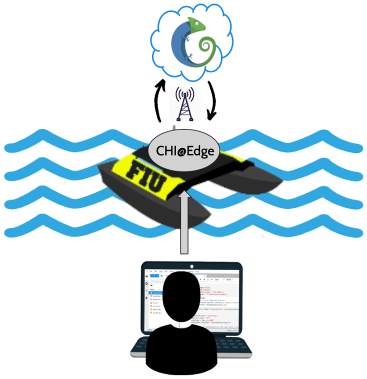
EDGE FOR MARINE BIOLOGY

- Goal: map existing fish populations and thereby understand better how pollution impacts their habitat and the general Biscayne Bay ecosystem
- Challenges: What is the best cloud/edge strategy for collecting and analyzing data from the autonomous vehicle (AV)? How does the resolution of video data and quality of network connection influence them?
- CHI@Edge: using CHI@Edge for developing edge to cloud data processing workflows via Jupyter notebooks

Kevin Boswell, Leonardo Bobadilla, and Jonathan Tsen
Florida International University

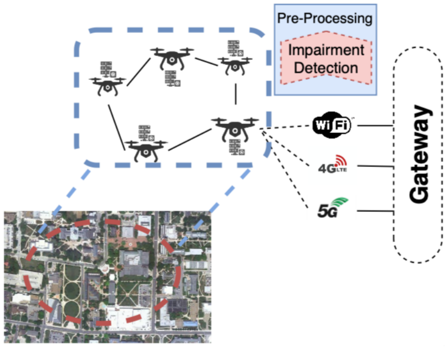




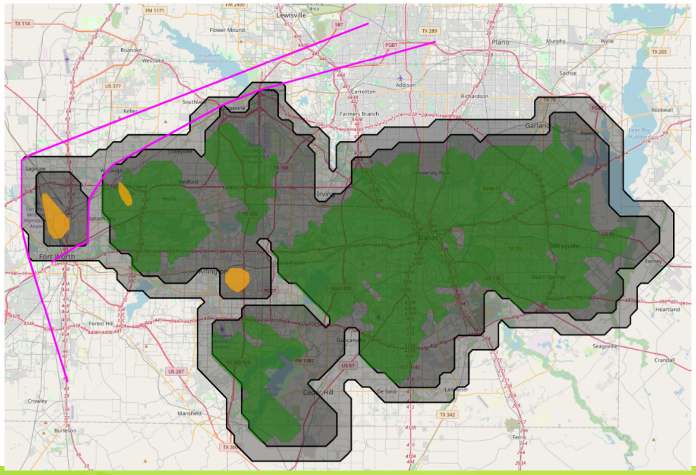


FLYNET: AN 'ON-THE-FLY' PROGRAMMABLE END-TO-END NETWORK-CENTRIC PLATFORM

- Architecture and tools that support edge computing devices in scientific workflows
- Critical for low latency and ultra-low latency applications: e.g., drone video analytics and route planning for drones
- Challenges: integration of compute and networking infrastructure, in-network processing, end-to-end monitoring, workflow management (Pegasus)
- CHI@Edge
  - Use for edge computing experiments
  - Provide experiments that can be reproduced by other researchers
  - FlyNet to provide tools to allow researcher to include CHI@Edge in their workflows



Mike Zink FlyNet PI
U of Mass, Amherst



http://www.ecs.umass.edu/mi360world/flynet/index.html

SHARING PLATFORM

- **Can experiments be as sharable as papers are today?**
- **Instruments held in common** are a reproducibility baseline
- Clouds: sharing experimental environments
  - Disk images, orchestration templates, and other artifacts
- What is missing?
  - Telling the whole story: hardware + experimental container + experiment workflow + data analysis + story – literate programming
  - The easy button: it has to be easy to package, easy to repeat, easy to find, easy to get credit for, easy to reference, etc.
  - Nits and optimizations: declarative versus imperative, transactional versus transparent

Paper: "The Silver Lining", IEEE Internet Computing 2020

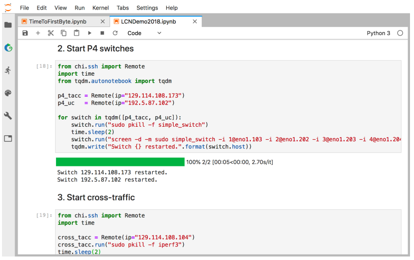
PRACTICAL REPRODUCIBILITY

- Hardware and hardware versions
  - >105 versions over 5 years
  - Expressive allocation
- Images and orchestration
  - >130,000 images, >35,000 orchestration templates and counting
- Packaging and repeating: integration with JupyterLab
- Share, find, publish and cite: Trovi and Zenodo

BENNY GOODMAN—VOL. 1 (30s)/SHEP FIELD
BENNY GOODMAN—VOL. 2/MCKINNEY'S CO
BENNY CARTER/ALVINO REY/HARLAN LEON
GLENN MILLER—VOL. 1/ WILL BRADLEY/ ORRIN
ide up

PACKAGING SHARABLE EXPERIMENTS

Literate Programming with Jupyter



2. Start P4 switches

```
[18]: from chi.ssh import Remote
      import time
      from tqdm.autonotebook import tqdm

      p4_tacc = Remote(ip="129.114.108.173")
      p4_uc   = Remote(ip="192.5.87.102")

      for switch in tqdm([p4_tacc, p4_uc]):
          switch.run("sudo pkill -f simple_switch")
          time.sleep(2)
          switch.run("screen -d -m sudo simple_switch -i 1@eno1.103 -i 2@eno1.202 -i 3@eno1.203 -i 4@eno1.204
          tqdm.write("Switch {} restarted.".format(switch.host))
```

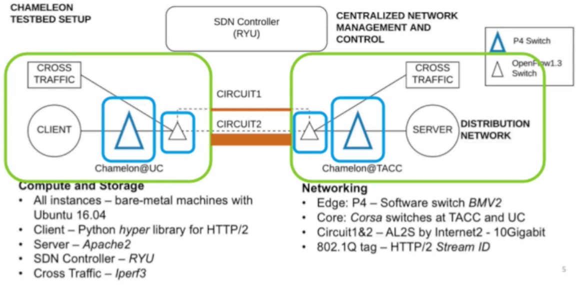
```
100% 2/2 [00:05<00:00, 2.70s/it]
Switch 129.114.108.173 restarted.
Switch 192.5.87.102 restarted.
```

3. Start cross-traffic

```
[19]: from chi.ssh import Remote
      import time

      cross_tacc = Remote(ip="129.114.108.104")
      cross_tacc.run("sudo pkill -f iperf3")
      time.sleep(2)
```

Experimental storytelling:
ideas/text, process/code, results

+



Compute and Storage

- All instances – bare-metal machines with Ubuntu 16.04
- Client – Python *hyper* library for HTTP/2
- Server – *Apache2*
- SDN Controller – *RYU*
- Cross Traffic – *Iperf3*

Networking

- Edge: P4 – Software switch *BMV2*
- Core: *Corsa* switches at TACC and UC
- Circuit1&2 – AL2S by Internet2 - 10Gigabit
- 802.1Q tag – HTTP/2 *Stream ID*

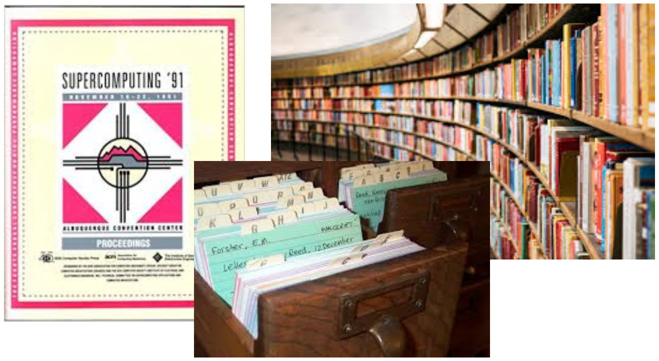
Complex Experimental containers

- Repeatability by default: Jupyter notebooks + Chameleon experimental containers
  - JupyterLab for our users: use jupyter.chameleoncloud.org with Chameleon credentials
  - Interface to the testbed in Python/bash + examples (see LCN'18: https://vimeo.com/297210055)
  - Shareable via Chameleon Trovi

Paper: "A Case for Integrating Experimental Containers with Notebooks", CloudCom 2019

PUBLISHING EXPERIMENTS

Familiar research sharing ecosystem



Digital research sharing ecosystem

Chameleon jupyterhub

?

- Digital publishing with Zenodo: make your experimental artifacts citable via Digital Object Identifiers (DOIs) – and executable via Chameleon daypass
- Integration with Zenodo
  - Export: make your research citable and discoverable
  - Import: access a wealth of digital research artifacts already published

zenodo

PARTING THOUGHTS

- Constantly in motion: scientific instruments are laying down the pavement as science walks on it
- Chameleons like to change:
  - Experimental environments that can adapt to your experiment
  - Testbed that adapts itself to your scientific needs -- from cloud to edge: CHI@Edge
- A public, sharable instrument underpins community sharing
- Sharing platform: from possible to easy – make your research, instruments, and tools shareable!

We're here to change

www.chameleoncloud.org

CHI AND CHI@EDGE SIDE BY SIDE

Chameleon for bare metal

Advanced reservations for **bare metal machines**

Bare metal reconfigurability

Single-tenant isolation

Heterogeneous collection of interesting hardware

Isolated networking, public IP capability, **OpenFlow SDN**

Composable cloud APIs (GUI, CLI, Python+Jupyter)

Owned and operated by Chameleon

Chameleon for edge

Advanced reservations for **IoT/edge devices**

Container deployment

Single-tenant isolation

Heterogeneous collection of interesting hardware **and peripherals/locations!**

Isolated networking, public IP capability

Composable cloud APIs (GUI, CLI, Python+Jupyter)

Mixed ownership model: bring your own device(s)!

JOIN US FOR THE SUMMER OF CHAMELEON!

- June 2021: CHI@Edge releases, shared hardware (nvidia nanos and raspberry pis), community webinars
- July 2021: “bring your own device” with attestations/SLAs, peripherals, support for limited sharing
- To use: https://www.chameleoncloud.org/experiment/chiedge/
- To learn: https://www.youtube.com/user/ChameleonCloud/videos
- Chameleon-edge-users mailing list: https://groups.google.com/g/chameleon-edge-users?pli=1
- Help us build a better testbed!