A Hands-on Workshop on P4 Programmable Switches

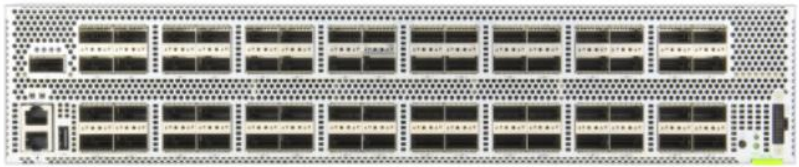
Jorge Crichigno, Elie Kfoury
University of South Carolina
http://ce.sc.edu/cyberinfra
jcrichigno@cec.sc.edu, ekfoury@email.sc.edu

February 16th, 23rd, 2022

Hands on Session 1: Intro to P4 and BMv2

Examples of P4 Programmable Switches

- Behavioral Model Version 2 (BMv2)
  - Open source
  - Software switch used for teaching, researching ideas
  - Good to validate ideas
- Commercial physical devices
  - E.g., Edgecore Wedge 100BF-65X (based on Intel's Tofino chip)
  - 65x100G switch ports
  - Used in production networks and research
  - Software license and confidentiality agreement (SLACA) with Intel



Introduction to P4 - BMv2 Lab Series

Lab experiments

Lab 1: Introduction to Mininet
Lab 2: Introduction to P4 and BMv2
Lab 3: P4 Program Building Blocks
Lab 4: Parser Implementation
Lab 5: Introduction to Match-action Tables (Part 1)
Lab 6: Introduction to Match-action Tables (Part 2)
Lab 7: Populating and Managing Match-action Tables
Lab 8: Checksum Recalculation and Packet Deparsing

Exercises

Exercise 1: Building a Basic Topology
Exercise 2: Compiling and Testing a P4 Program
Exercise 3: Parsing UDP and RTP
Exercise 4: Building a Simplified NAT
Exercise 5: Configuring Tables at Runtime
Exercise 6: Building a Packet Reflector

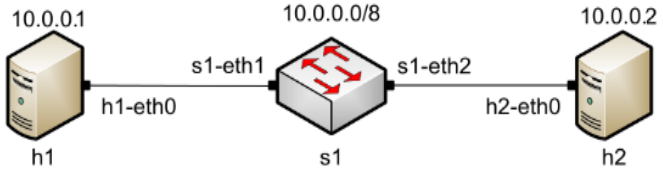
Environment: Mininet

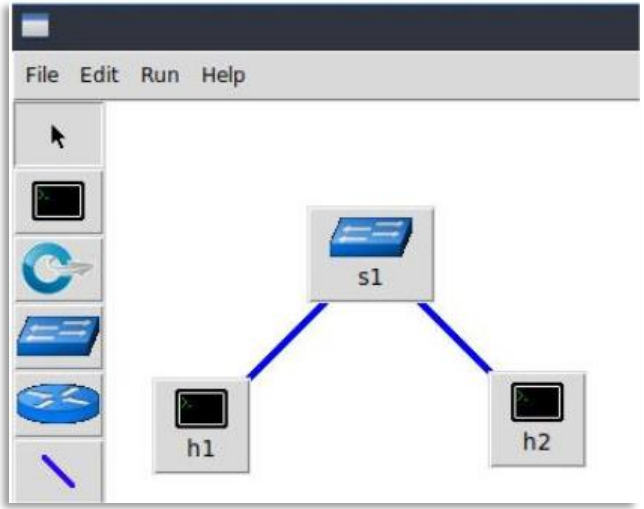
Mininet

- Mininet is a virtual testbed
- Nodes are containers, or more accurately, *network namespaces*
- Features
  - Fast prototyping for new protocols
  - Simplified testing for complex topologies
  - Realistic emulation, real code
  - Open source
  - Complex networks can be created (100s or 1,000s of nodes)

MiniEdit

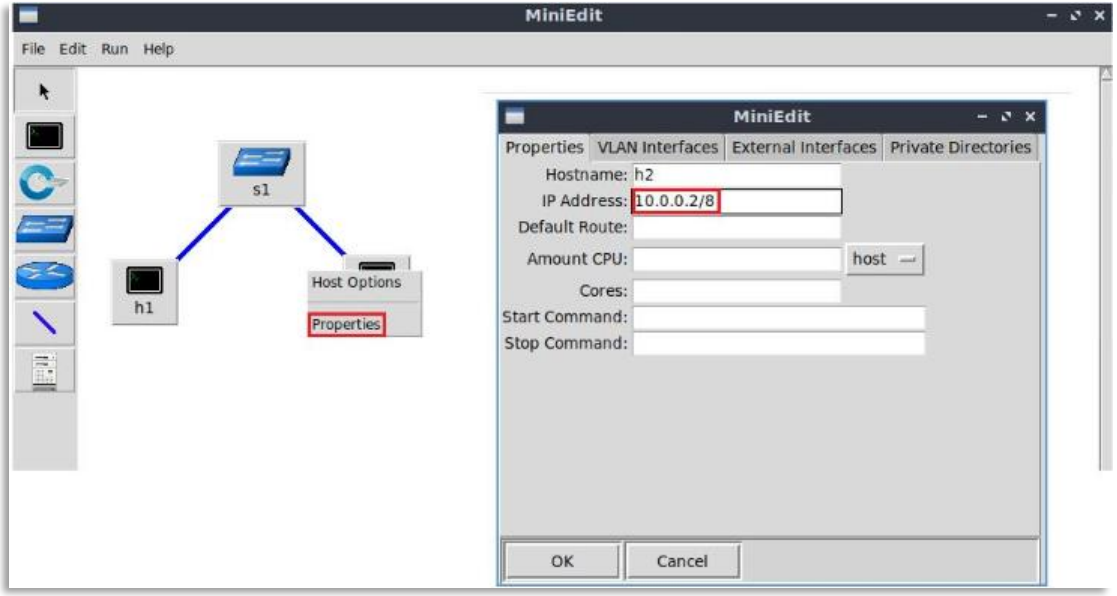
- To build a topology, we use MiniEdit
- MiniEdit is a simple GUI editor for Mininet
- Example:





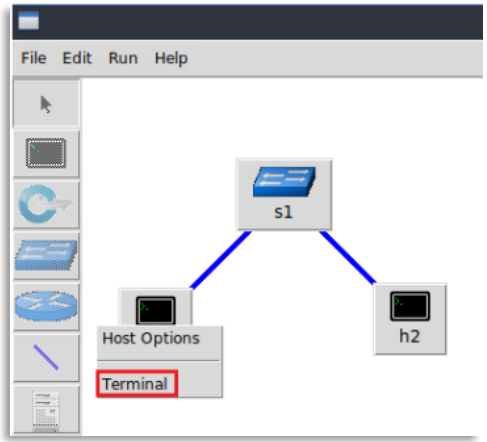
Host Configuration

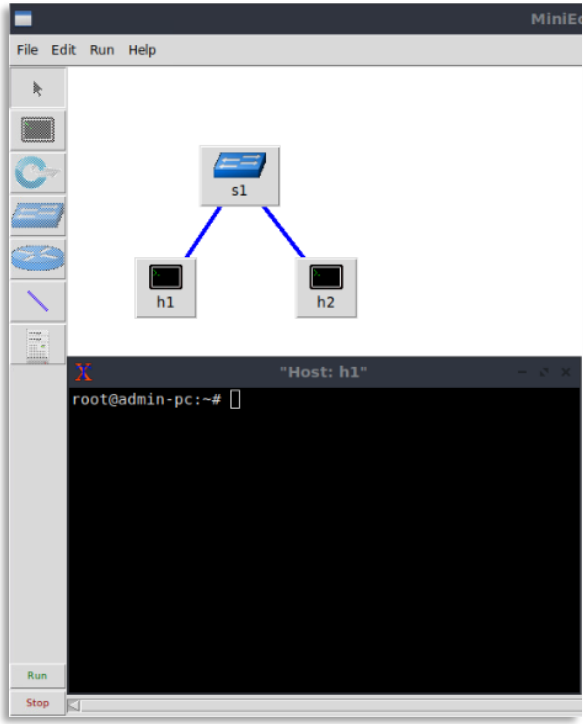
- A host can be configured by holding the right click and selecting properties on the device

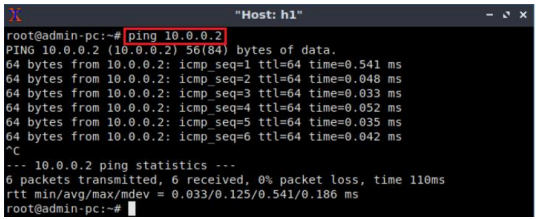


Executing Commands on Hosts

- Open a terminal on host by holding the right click and selecting *Terminal*

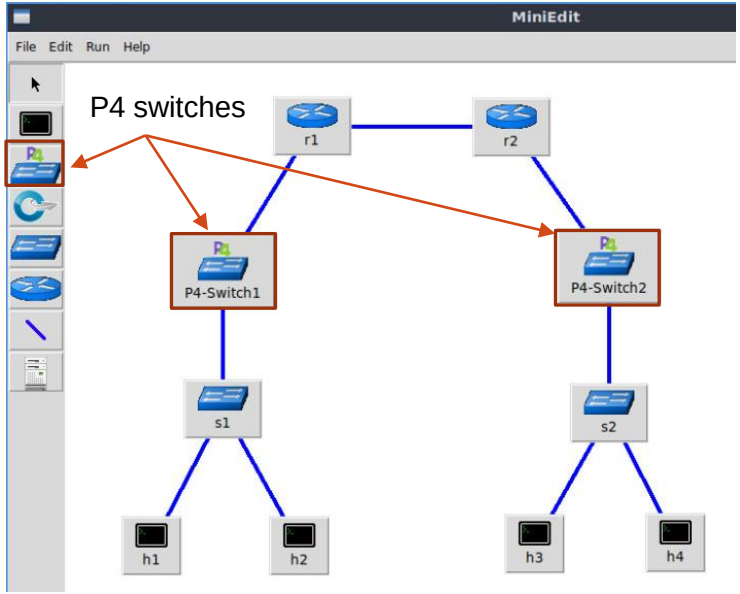


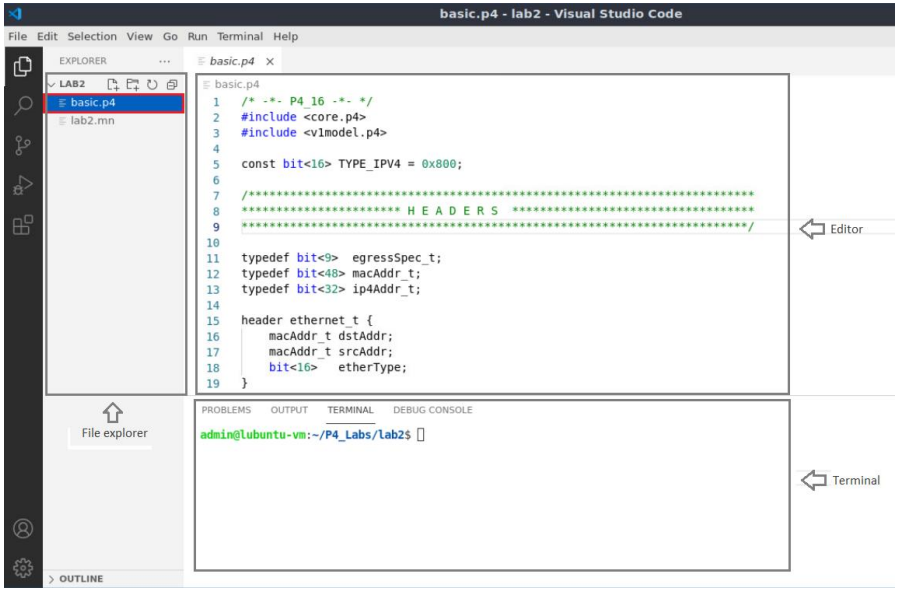




Development Environment

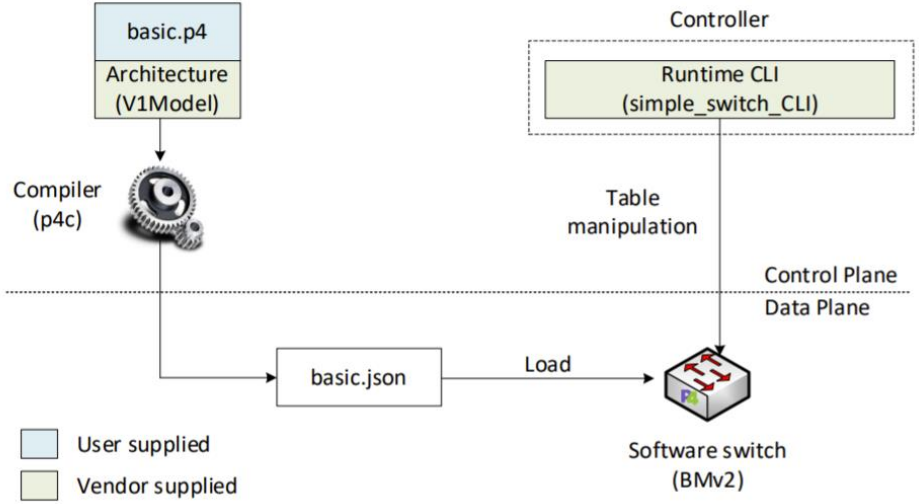
- BMv2 switches running inside Docker containers
- Code written in Visual Studio Code with a built-in terminal
- Other devices available: FRR routers, OvS switches, Linux hosts





Workflow of a P4 Program

- Workflow used to program the BMv2 switch

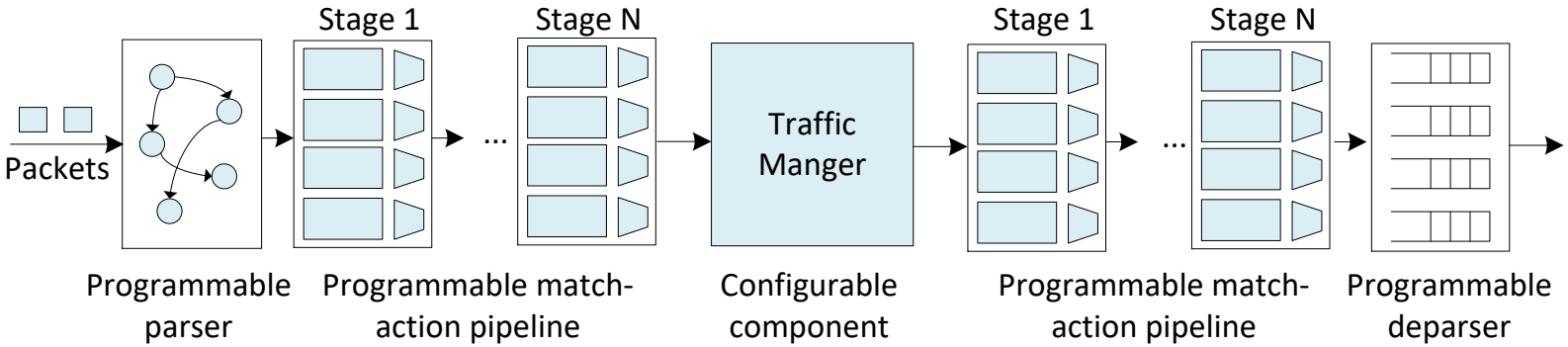


Workflow used in the lab series

Lab 3: P4 Program Building Blocks

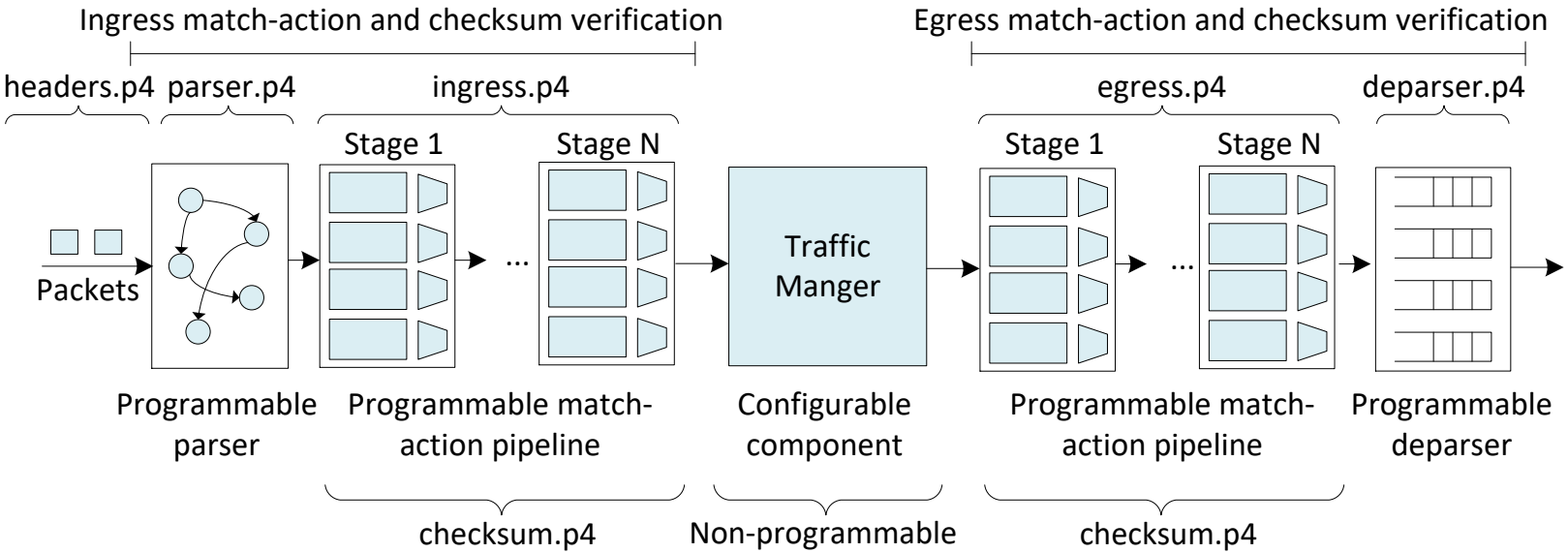
V1Model

- Common $P4_{16}$ architecture used with BMv2
- Implemented on top of BMv2's *simple_switch* target
- It consists of a programmable parser, an ingress match action pipeline, a traffic manager, an egress match-action pipeline, and a deparser



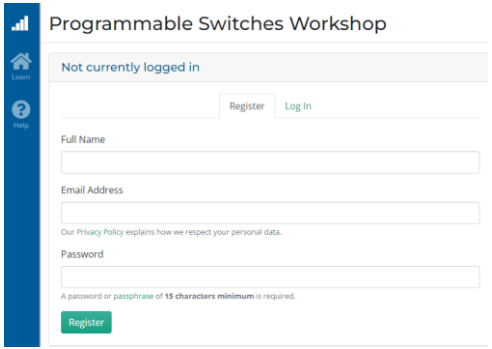
V1Model

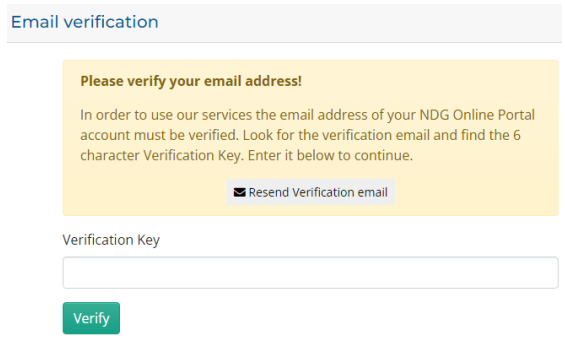
- Common $P4_{16}$ architecture used with BMv2
- Implemented on top of BMv2's *simple_switch* target



Registering to the Netlab Portal

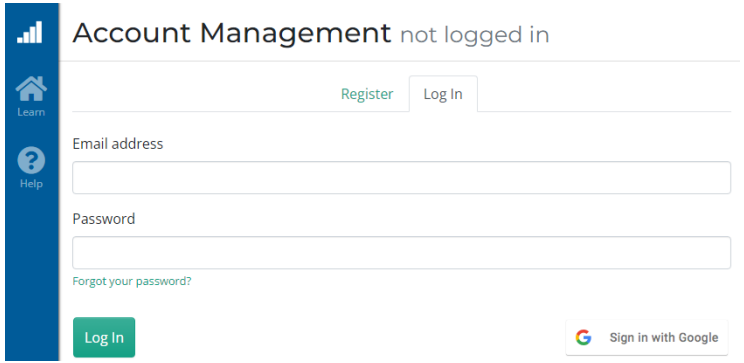
- If you DID NOT register in the Netlab portal, please use the following link:
- https://portal.netdevgroup.com/learn/fmgqx8/enroll/
- Fill out the form with your full name, email address, and password.
- Check your email inbox for the verification key.
- Complete your enrollment by accepting the terms and conditions and claiming your free access
- Finalize the registration by claiming your free access

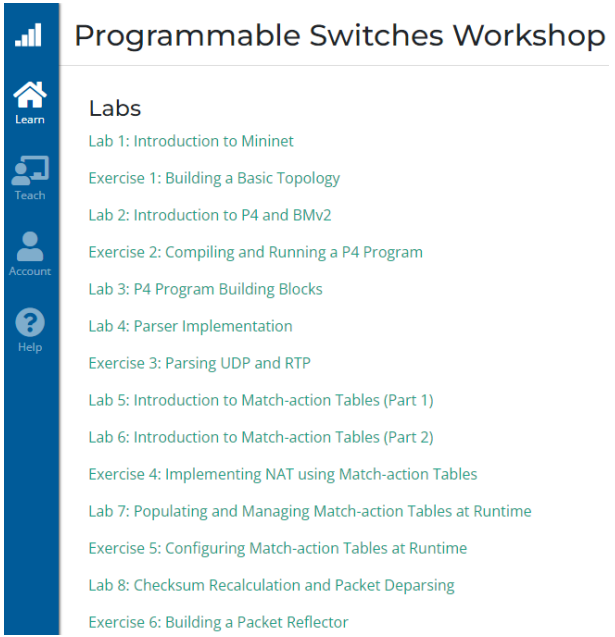




Accessing the P4 labs

- If you already registered, login to the Netlab portal using the following link:
- https://portal.netdevgroup.com/account/login
- Click on the "*Programable Switches Workshop*" course
- Select the lab you want to run (e.g., Lab 3)





Lab Topology and Objectives

- The topology consists of two hosts: h1 and h2; one P4 switch: s1
- Compiling a P4 program and pushing the output to the data plane
- Starting the switch daemon and allocating interfaces
- Testing and verifying the P4 program