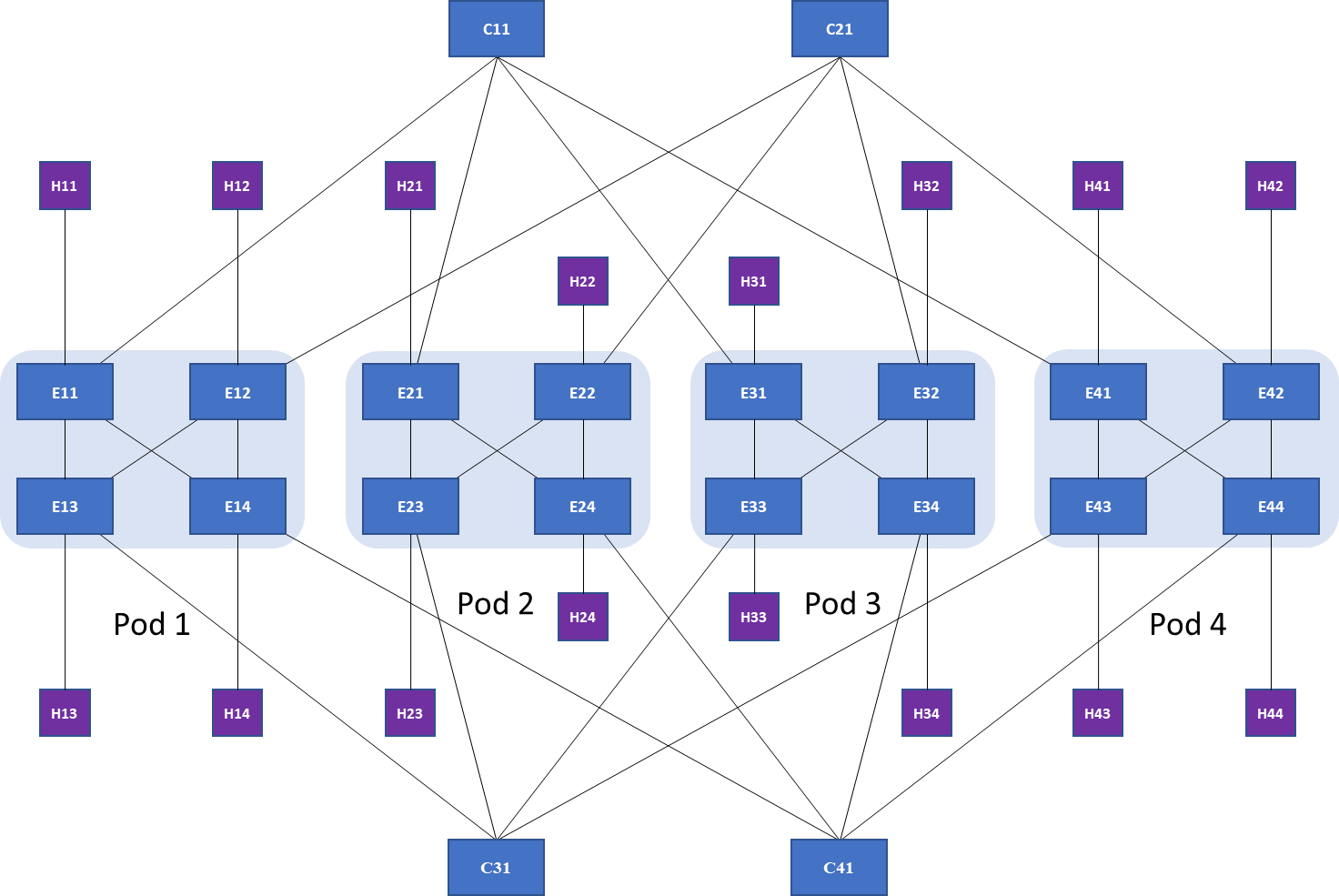
**Software-Defined Networking Assignment 2 2019-20:**

**“Design, create and debug an SDN network consisting of multiple network elements (SDN switches and controller).”**

## Description

Write a Mininet script in Python to create a small network using the Diamond experimental Data Centre Network topology. This topology is built using switches with ‘k’ ports (e.g. 4-port switches, as shown in the example topology below), where ‘k’ is an even number. Note that the value ‘k’ must be passed to your Mininet scripts as a parameter. There are ‘k’ pods, each containing ‘k’ edge switches which are divided into an upper row and a lower row. An edge switch is connected to one host, all switches in the opposite row, and a core switch. There are ‘k’ core switches, divided into upper and lower rows. Each upper row core switch is connected to ‘k’ upper row edge switches, and each lower row core switch is connected to ‘k’ lower row edge switches. The switches should be Open vSwitch instances, and the remote controller can be Pox (or other, by agreement in advance with your lecturer).

Core switches should be named ‘C*x*1’, where x is a number from 1 to ‘k’. Edge switches should be named ‘E*xy*’, where *x* is the pod number, and *y* is the number of the switch within the pod. Hosts should be named ‘H*ab*’, where ab is the same as *xy* in the name of the switch to which the host is connected. A host’s IP address should be ’10.*a*.*b*.10’.

Write an application for the controller to proactively add rules that don’t time out for the following functions:

* H11 and H12 should be able to exchange any kind of traffic with each other (reachability functionality).
* H13 and H14 should never be able to communicate with each other (traffic isolation, similar to VLAN functionality).
* H11 should be able to telnet and SSH to H44, but no other traffic should be allowed between them (stateless firewall functionality).

Rules should be reactively configured to achieve the following functions, and the rules should have an idle timeout of 30 seconds:

* H21 should be able to telnet and SSH to H34, and H34 should be able to send any kind of traffic to H21 (stateful firewall functionality).
* For HTTP traffic going from H21 to H31/H32, every second flow should go to H31, and every other flow to H32 (load balancer functionality).

You will need to think about ARP. You can set static ARP entries on all hosts, or you can handle ARP requests in your own code, or use an existing controller application to deal with ARP.

You must submit a .zip file (named <student-number>-<firstname.surname>-assign2.zip) including only the following files:

* sdntopo2.py – a script to create the Mininet topology.
* policy.py – a Pox script to implement the policies listed above. Your code must be meaningfully commented.
* output.txt – something that shows that the functions are achieved by your code (e.g. the commands to run your code, and output of testing with hping3 and tcpdump).

## Other Information

* Submission is through Canvas (<https://cit.instructure.com>) only.
* Assignment value: 35%
* Submission date: 12 Dec 2019

## Marking

* Correctly following submission instructions –> 2 \* 3 = 6%
  + Correct naming of files,
  + Correct file formats – py, txt & zip
* Mininet script (sdntopo2.py) – (20%)
  + Scripting style (indentation, loops, parameters, clean code) –> 4 \* 2 = 8%
  + Correctly build topology (switches/hosts/links/controller present, named correctly –> 4 \* 3 = 12%
* Pox script (policy.py) – (61%)
  + Scripting style (indentation, clean code, meaningful comments) – 3 \* 3 = 9%
  + Proactive policy (in ConnectionUp: new rule, match criteria, action + timeout, sent to switches) –> 4 \* 3 \* 3 = 36%
    - Reachability rule
    - Traffic isolation rule
    - Stateless firewall rule
  + Reactive policy (in PacketIn: new rule, match criteria, action + timeout, sent to switch) -> 4 \* 2 \* 2 = 16%
    - Stateless firewall rule
    - Load balancer rule
* ARP consideration (static in Mininet script, dynamic in Pox script, existing Pox application) – 4%
* Evidence of successful operation of scripts (output.txt – topology created; proactive / reactive behaviour, Pox output) –> 3 \* 3 = 9 %

[Changes for next time: