Project 3

ECE544 Communication Networks II

Francesco Bronzino

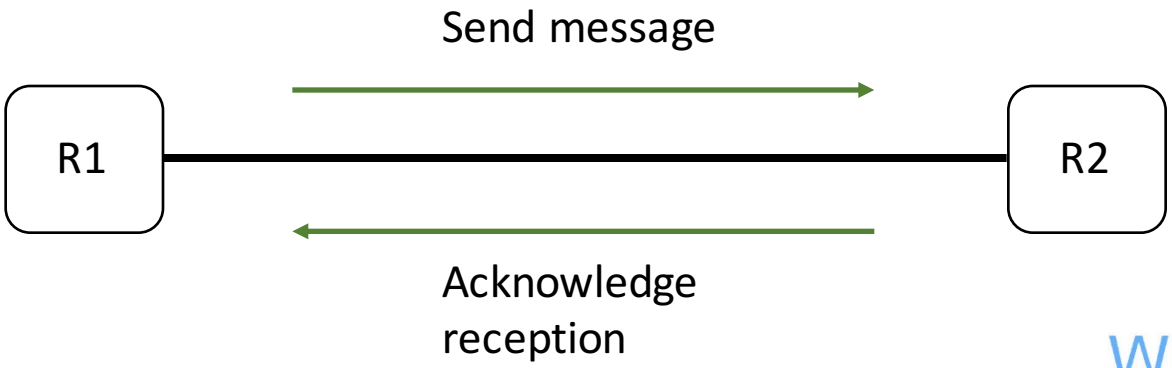
Includes teaching material from Bart Braem and Michael Voorhaen

Project Goals

- Write clients using Click
- Implement basic routing concepts

Exercise 1 - Intro

- **DISCLAIMER:** Click is not meant to be used to write clients, but to simplify your task we will use its framework to implement basic network clients
- Consider the scenario where two clients are directly connected
- Implement a basic client that ***reliably*** sends a message to another client

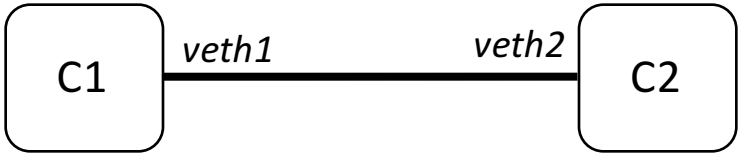


Exercise 1

- Define a custom packet
  - **HINT**: How many header fields? (Type? (data or ack), Destination address?, Source Address?, Sequence number?, etc..)
- Write an element that generates a data message and confirms whether the message was correctly delivered
  - **HINT**: Should it retransmit the message if the ack is not received?
- Write an element that when receives a data message transmits an acknowledgment

Exercise 1

- Generate a small network of two clients
  - $ sudo createNet2



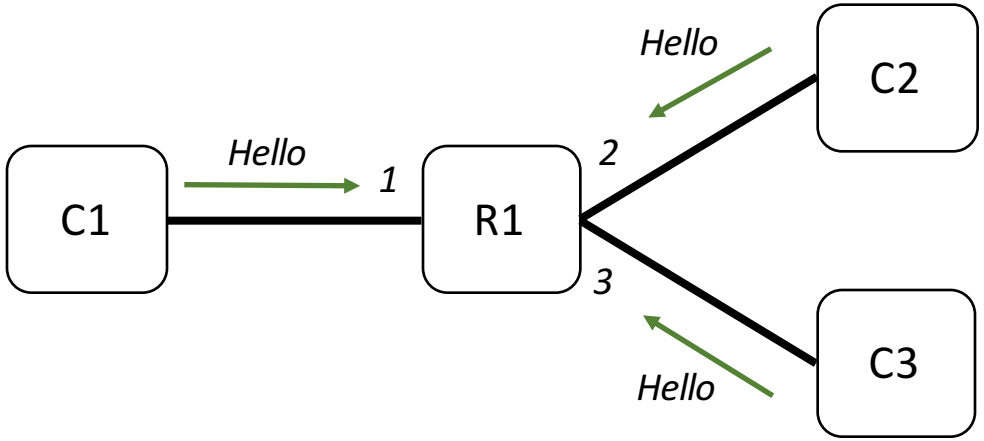
- C1 transmits the data packet and prints when acknowledgment from C2 is received
- **HINTS:**
  - If a source address is used, should C2 consider its content?

Exercise 2 - Intro

- Consider a scenario where more clients are connected to a router.
- The clients periodically (every 2 seconds) transmit *hello* messages to announce their presence and their address to the router
- The router learns which port clients are attached to
- When a data message is received by the router, it decides where to forward it based on the destination address.

Exercise 2 - Intro

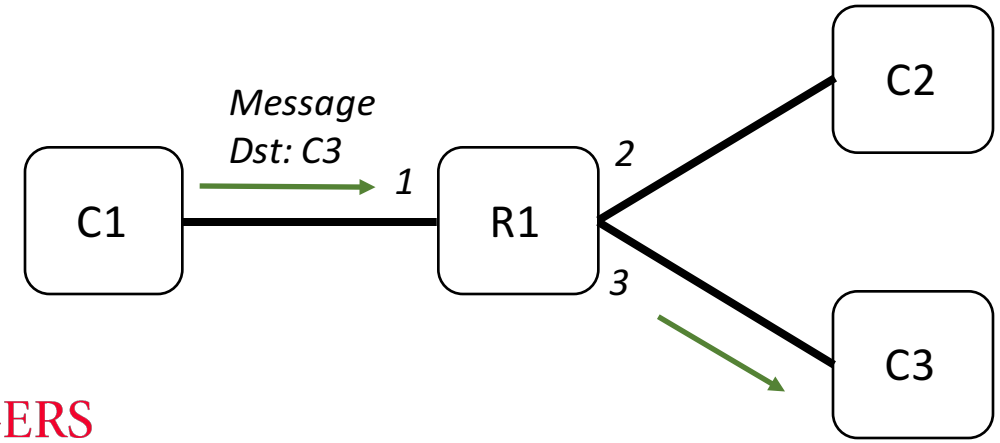
- Discovery phase



Forwarding Table

Dest	Next Hop
C1	1
C2	2
C3	3

- Forwarding phase



Forwarding Table

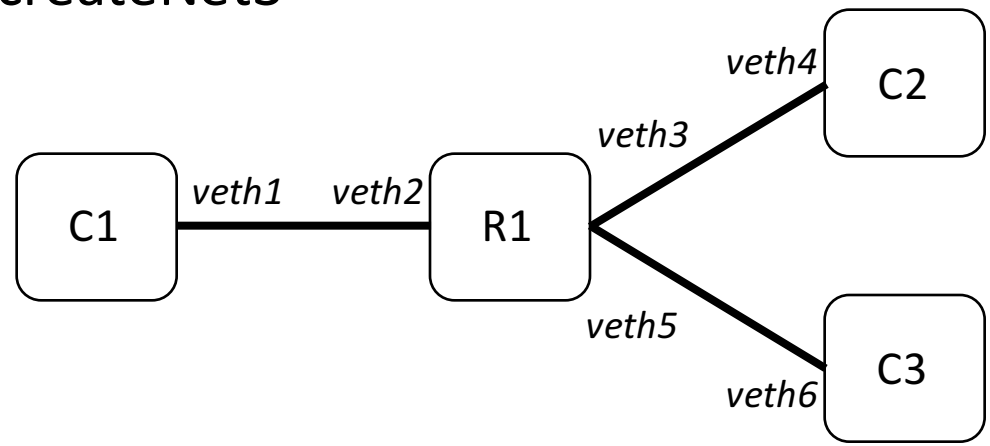
Dest	Next Hop
C1	1
C2	2
C3	3

Exercise 2

- Client:
  - Define two custom packets, one for the *hello* and one for data/ack
    - **HINT**: Can you re-use the same packet from ex 1?
  - Write an element that periodically generates *hello* messages announcing the client address
    - **HINT**: Period 2 seconds
- Router:
  - Write an element that classifies incoming packets based on type (data/ack or hello)
  - Write an element that learns from *hello* packets and stores to which port clients are attached to and forwards data/ack packets to the proper port
    - See hints for more details

Exercise 2

- Generate a small network of one router and three clients
  - $ sudo createNet3



- Initial discovery phase
- C1 transmits the data packet to a given destination and prints when the acknowledgment is received

Exercise 2 – Ports Based Switching

- Basic solution:
  - You learnt from previous classes that an element can have a flexible number of input and output ports
    - *const char *port_count() const { return "1-/1-"; }*
  - You can learn from which port a packet is incoming based on a function parameter:
    - *void push(**int port**, Packet *p)*
  - Match input and output ports of your element to the right RouterPort elements used
- Advanced solution:
  - Use annotations (Paint element) and switch based on the set annotation (PaintSwitch element)

Exercise 3

- Same as exercise 2 but now use LossyRouterPort with a loss probability of 0.2
  - elements/lossyrouterport.click
  - Same as RouterPort plus two parameters:
    - LOSS: expressed as probability P (e.g. 0.2)
    - DELAY: expressed in seconds
- **HINT:** If everything in exercise 1 and 2 has been correctly implemented, exercise 3 does not require any additional work except for changing RouterPort with LossyRouterPort in the configuration files

General Info

- Due: April 1st
- Submission instructions:
  - Submit a single archive (zip or tar.gz) to bronzino@winlab.rutgers.edu with subject "ECE544 Project 3"
  - Include in the archive 3 folders named "exercise1", "exercise2", "exercise3". They should contain only files that you implemented (i.e. click configuration files, new elements, etc.).
  - **Do not** include the whole click resources or binary files!