Man-in-the-Middle (MITM) Attack

# Objective

You will use Ettercap to perform man-in-the-middle (MITM) attacks between the HMI and PLC for your traffic light simulation. This exercise has 3 parts.

1. Write an Ettercap filter to change actuator states.
2. Write an Ettercap filter to perform DoS attacks.
3. Write an Ettercap filter to inject information to the PLC.

**NOTE:** You will need to look up function codes here: <https://www.simplymodbus.ca/FC01.htm>

You may need to upgrade your Ettercap version to get it to work properly. Try:

1. sudo apt-get update
2. sudo apt-get install ettercap

You can refer to some Ettercap examples here:  
<https://github.com/Ettercap/ettercap/blob/master/share/etter.filter.examples>

You must compile the filters before using them:

etterfilter filename.ecf -o filename.ef

Load the filter into Ettercap GUI with Filters → Load a filter or on command line -F [filter.ef]

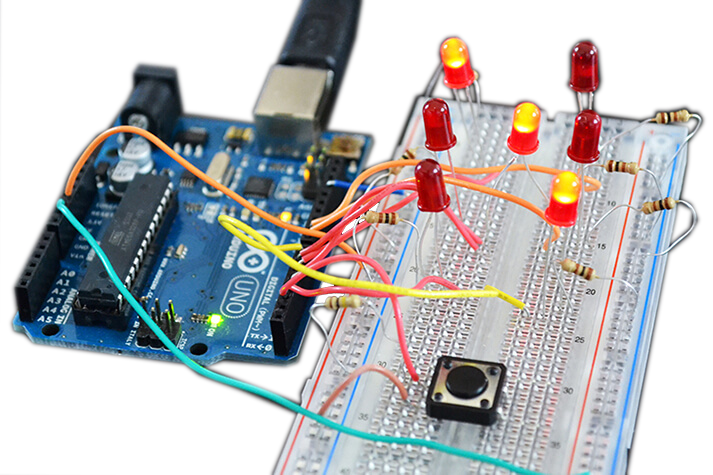
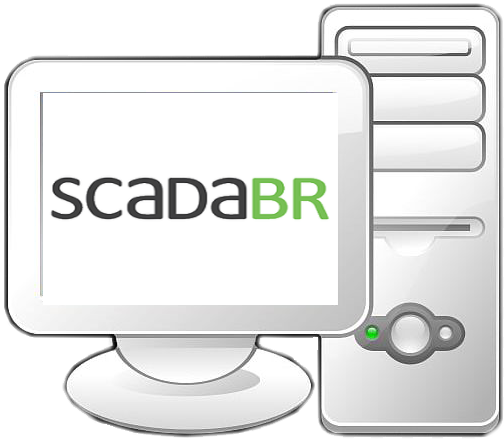
**Obs: Some versions of Ettercap require the filter name to be appended with :1 for the filter to be active. If you're facing problems with your filter, just rename it so that it has :1 at the end, for example: filter:1.ef**

# Before Starting

It is essential that you use the system you have created from the previous assignments.

## Network Architecture Requirement

A man in the middle is so named because the attacker is placed in the middle of the typical connection between two parties. The network architecture thus has an attacker positioned between the PLC and the HMI. The attacker can use Ettercap to disrupt network communication. So, check to make sure you have the network architecture below before beginning this assignment.



Traffic Lights

**Attacker** - [IP ADDRESS #3]

**PLC** - [IP ADDRESS #1]

**HMI** - [IP ADDRESS #2]

# Part 1: MITM – Perform a DoS attack

DoS attacks prevent normal operations in a system and can render the system unavailable. You can implement a DoS attack using Ettercap by specifying certain behavior for the traffic to follow, such as dropping all the packets or fencing a connection reset.

## DoS: Dropping Packets

1. Write an Ettercap filter to drop all the packets sent from the PLC to the HMI and from the HMI to the PLC, creating a DoS attack.

# Part 2: MITM – Change actuator states

## MITM: Invert System Behavior

## For these exercises it is assumed that the LEDs on your breadboard represent the North-South light stack. If in your system the LEDs are representing East-West, please run the attacks against East-West instead.

## Which bit holds the value of the red-light in the North South light stack?

## Write an Ettercap filter that inverts the coil value when in transit from the PLC to the HMI. HINT: It is always useful to capture network traffic on wireshark before the attack. This will give you a better view of the network data and which bytes represent what on the communication protocol.

## MITM: Set the System State to an Impossible State

## Using the information gathered in 4.1, write a filter that manipulates all traffic lights on the North South stack so that they all seem to be on at the same time on the HMI.

# Post Exercise Report

## (30 points) Submit a video showing the effects of each one of your attacks (3.1, 4.1, and 4.2) on the HMI and the PLC (breadboard).

## (30 points) Submit all your Ettercap filters.

## (20 points) In *Exercise 6: DoS*, the HMI was able to respond when performing SYN flood; however, the HMI does not respond when conducting a MITM DoS on 3.1. Explain why this is the case. Compare and contrast these attacks.

## (20 points) What are some countermeasures to prevent MITM attacks? Explain.