Jared Grimm, Elsa Luthi, Roger Wirkala

**Mission 1**

Below are in order instructions per virtual machine on how to implement our various scenarios

**Attack**

* On 10.4.9.1
  + cd mission1, cd attack
* On 10.4.9.3
  + cd mission1, cd attack
  + python3 globalServer.py
* On 10.4.9.2
  + cd mission1
  + python3 adversary.py <desired outcome>
    - Where desired outcomes are either ‘croc’ ‘yeezy’ or ‘knot’
* On 10.4.9.1
  + python3 laceClient.py <candidate>
    - Where candidate is ether ‘croc’ or ‘yeezy’

**Defended Infrastructure, No MiTM**

* On 10.4.9.1 :
  + cd mission1, cd defense
* On 10.4.9.3
  + cd mission1, cd mitigated
  + python3 protectedServerUpdated.py
* On 10.4.9.1
  + python3 protectedLaceUpdated.py <candidate>
    - where candidate is either ‘croc’ or ‘yeezy’

**Defended Infrasture, With Attempted MiTM**

* On 10.4.9.1 :
  + cd mission1, cd mitigated
* On 10.4.9.3
  + cd mission1, cd mitigated
    - python3 protectedServerUpdated.py
* On 10.4.9.2
  + cd mission1
  + python3 adversary.py <desired outcome>
    - Where desired outcomes are either ‘croc’ ‘yeezy’ or ‘knot’
* On 10.4.9.3
  + cd mission1, cd mitigated
  + python3 protectedLaceUpdated.py <candidate>
    - where candidate is either ‘croc’ or ‘yeezy’

**File Explination**

* adversary.py: represents the adversary
* db.txt: represents the current combined votes from the laces
* globalServer.py: represents the global server that receives all of the laces votes without any decryption functionality
* laceClient.py: represents a lace without any encryption functionality
* protectedLace.py: represents a lace with encryption functionality
* protectedServer.py: represents the global server that receives all of the laces votes with decryption functionality
* main.java: creates the html for the website
* scraper.py: web scraper we scripted for the real internet