# CSCI 400 Security Lab
# Topic: Intrusion Detection Systems

Prof. Dietrich

October 4, 2017

Group size: 4 — Setup needed: Windows/MacOS X, Unix hosts, DETER.

## 1 General description

In this lab we are getting familiar within intrusion detection systems (IDSes), both at the host and network levels. You are also setting up an environment to be used for future labs.

## 2 File with samples and results

The files are in lab-ids.zip. The files contain sample configurations (basic) and examples for setting up simple IDSes. In some cases, you still have to develop some scripts, write your own code to improve the results, or develop signatures (direct or behavioral).

## 3 Setting up an IDS

In this part of the lab, we will be setting up signature for hosts and networks.

### 3.1 Requirement

- You use the instructions provided with this lab. You should use DETER, as we will continue using this setup beyond this lab.
- Your output should demonstrate that you have done the experiments, such as screenshots, Unix typescripts, or logs.

## 4 Tasks

You are on the security team at FrobozzCo. Recent requests from management after the restructuring fiasco aim at upgrading the security of hosts and networks at FrobozzCo to deflect any attempts by outsiders, including those from the Zembor Corporation. You are to upgrade the security stance and explore host and network intrusion detection systems, using rule-based, signature, and behavioral approaches. First you must do this in a test environment before deploying it in a production setting.

## 4.1 Snort

1. In a FreeBSD or Ubuntu environment, install snort in a small network. Populate the network with at least a web server, an IMAP mail server, an IRC chat server, and an NFS mount point.
2. Trigger these two rules in the Snort IDS using packet generation of your own (without attack tools), e.g. using netcat.
    - `log tcp any :1024 -> 192.168.1.0/24 500: log tcp traffic from privileged ports less than or equal to 1024 going to ports greater than or equal to 500`
    - `alert tcp any any -> 192.168.1.0/24 143 (content: "|90C8 C0FF FFFF|/bin/sh"; msg: "IMAP buffer overflow!";)`
3. Develop a rule to detect traffic to the IRC server and the NFS server, specifically the traffic to the ports for each service.

## 4.2 Bro

1. In a FreeBSD or Ubuntu environment, install the Bro IDS.
2. Trigger three rules in the Bro IDS using packet generation of your own (without attack tools), e.g. using netcat. (suggestions: connflood, targeted-scan, arp, anon-http-user-agent)

## 4.3 OSSEC HIDS

1. In a FreeBSD, Ubuntu, or OpenBSD environment, install the OSSEC Host-based IDS.
2. Trigger three rules in the OSSEC IDS.
3. Bonus: How would you circumvent the triggering of the alerts by the IDS?

## 4.4 Honeyd

1. In a FreeBSD orUbuntu environment, install honeyd.
2. Place honeyd strategically in the network you created above, faking a small set of nodes, e.g. at most 5 with a variety of operating systems (Windows, MacOS, Linux) and respective versions. Remember that your DETER setup limits the total number of real nodes per group.

## 4.5 Nmap and Nessus

1. In a FreeBSD or Ubuntu environment, install nmap and nessus (or openvas).
2. Scan, with the goal of obtaining operating system type, open ports, and vulnerabilities, the network you created from at least two hosts (including the one running OSSEC HIDS) using nmap and nessus (or openvas), observing the reaction from the various IDSes.

### 4.6 Testing and validation

1. If possible, scan the network with a penetration testing team within your group, e.g. using nmap and nessus (or openvas). You may use the Kali Linux OS in DETER as a scanning tool as well. Comment on your findings from the reconnaissance and detection perspectives. Did you detect the penetration testing groups attempts? Did you have to write your own scripts/signatures?

2. Can you trigger this alert: `alert tcp any any -> any any (minfrag: 256; msg: "Tiny fragments detected, possible hostile activity";)`

3. Using simple Unix operating system tools (e.g. netcat) or by writing your own code, create a packet stream that is composed of 3 UDP packets, with 0xF00F at offset 100 in the first packet, 0xCAFE at offset 200 in the second packet, and 0xBEEF at offset 300 in the third packet. Write a Snort rule for this packet stream and detect it.

4. Using simple Unix operating system tools (e.g. netcat) or by writing your own code, trigger this alert: `alert tcp any any -> any 23 ( msg:"MALWARE-BACKDOOR w00w00 attempt"; flow:to_server,established; content:"w00w00"; classtype:attempted-admin; )`

5. Write a Bro policy to detect the viewing/web browsing of the CUNY John Jay front page by content only (not by URL).

## 5 Word Problems

1. Do the systems mentioned in section 4 work together? If so, explain how. If not, why not?

## 6 Deliverables

1. A report describing all your findings above.

2. A zip file containing:
   - Any signatures or scripts developed.
   - Answers to all word problems in Part 5.

3. Submit by the lecture on October 11, 2017.

## 7 Grading

Points will be subtracted if any of the pieces of the deliverables are missing or incomplete.