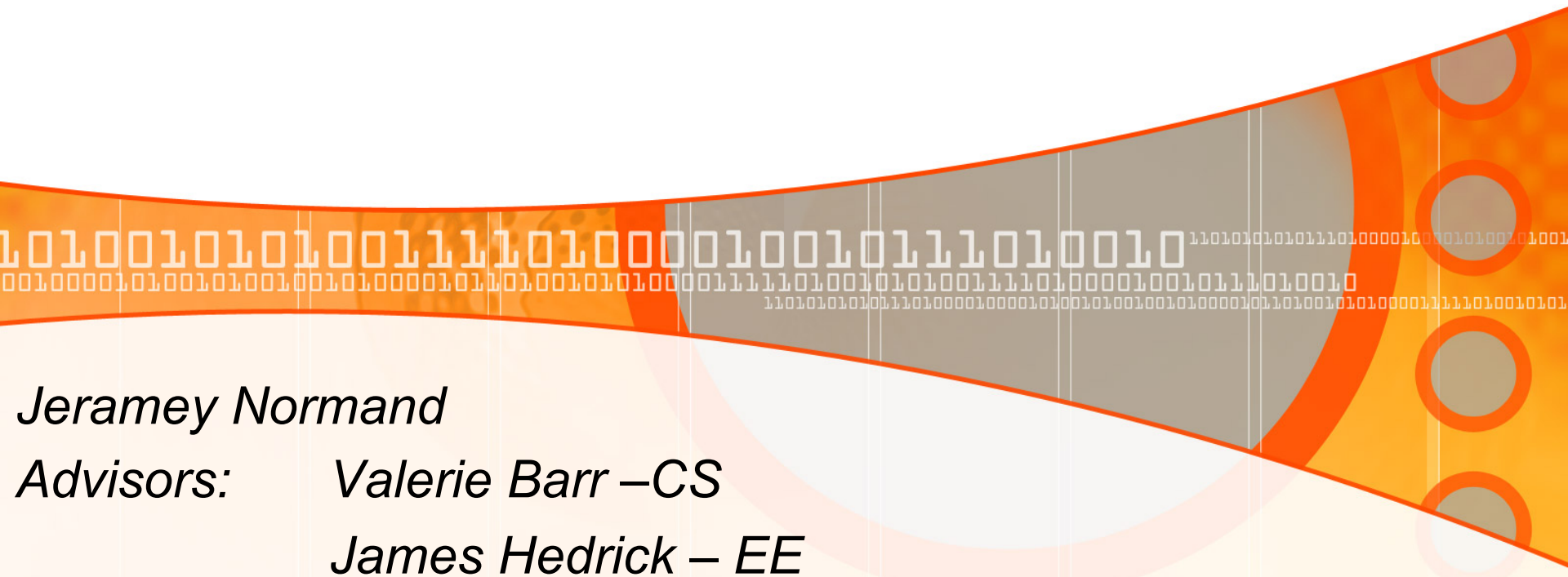
Intrusion Detection Systems: Investigation of Evaluation Corpora



Jeramey Normand

Advisors: Valerie Barr –CS

James Hedrick – EE

Big Picture

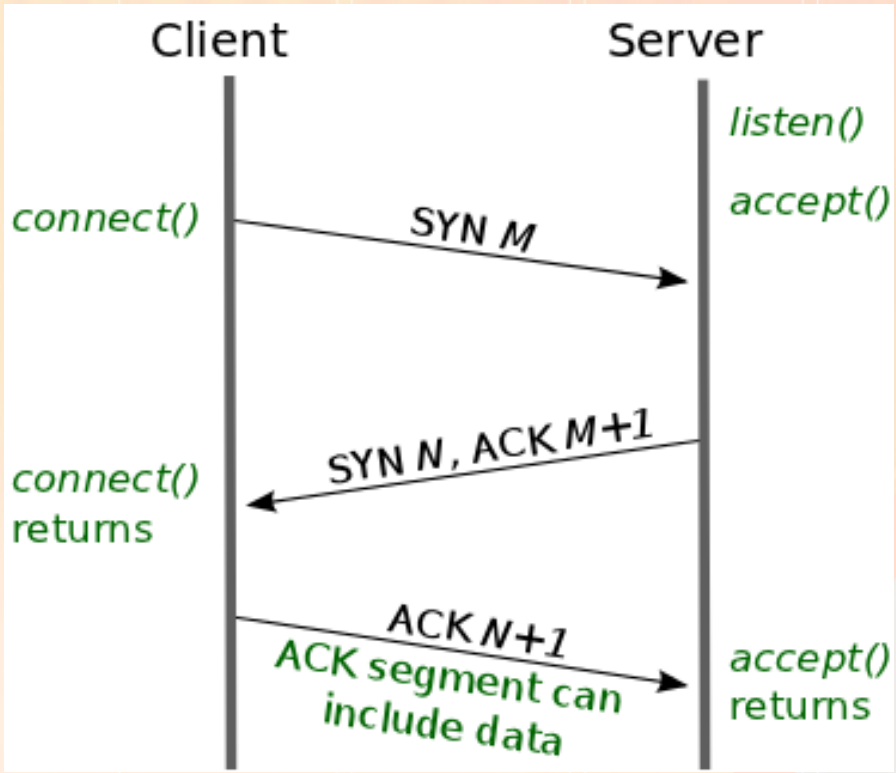
- No one is 100% secure
- Types of intrusions
  - DoS
  - U2R
  - R2L
  - Port Scan/Sniff
- QoS is a driving force in most all industries
  - Rule of five 9's
  - →Detection is paramount

My Interest

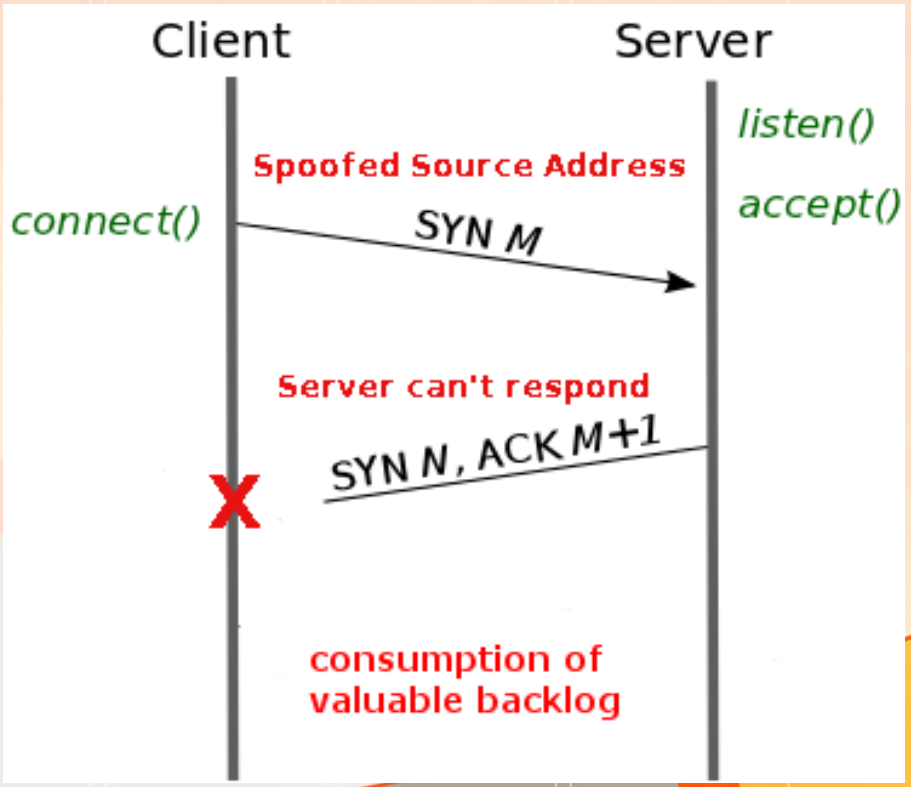
- Focus: Denial of Service (DoS) attack
- Why?
- What is it?

Normal vs DoS Connection

Normal TCP Connection Establishment



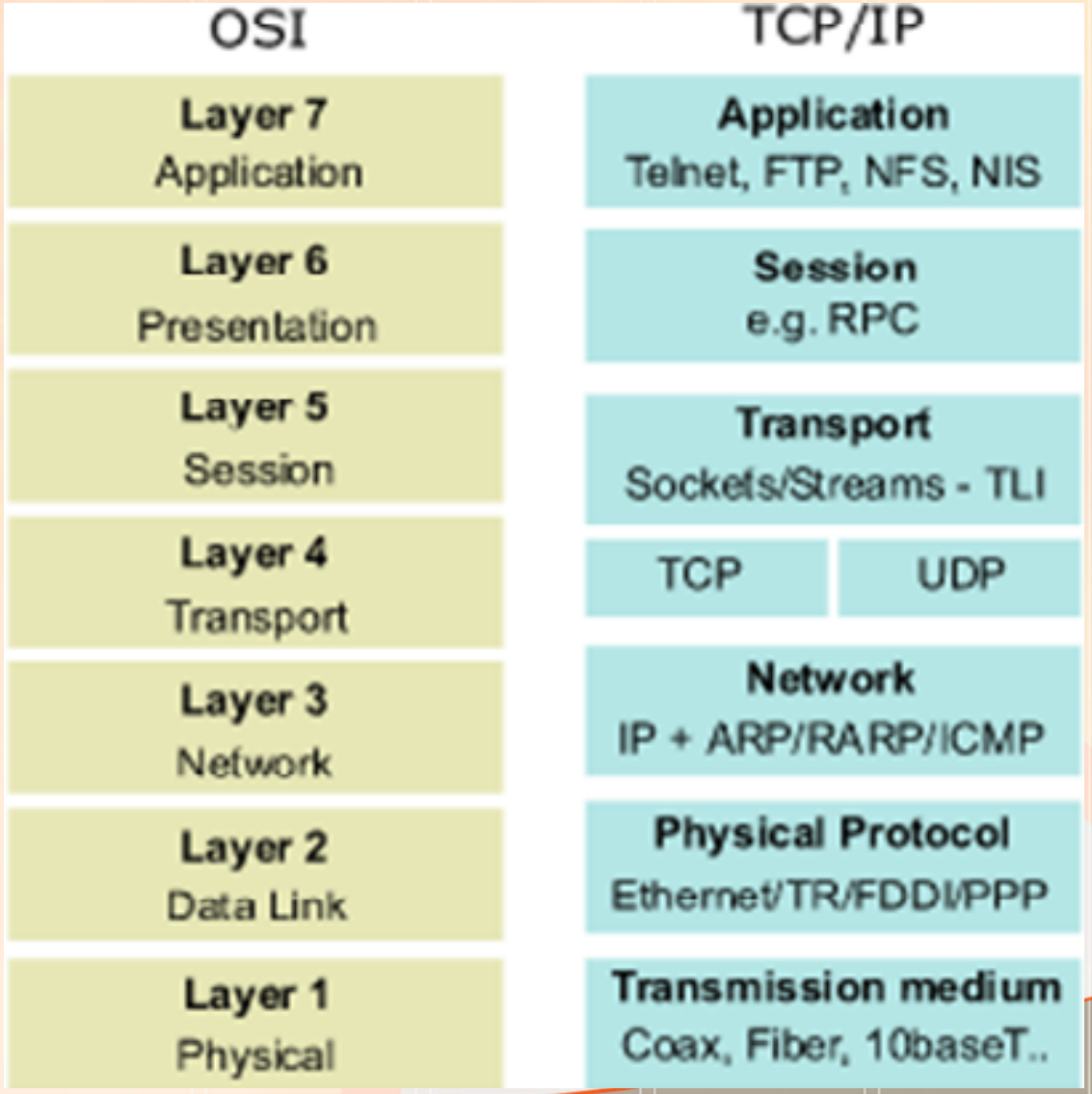
Neptune DoS Attack



DoS Complexity

- TCP/IP protocol suite
- Protocol + protocol field settings = many variations of DoS
- Primitive attacks: lower 4 layers
- Sophisticated attacks: 7th layer

OSI Model and TCP/IP Stack



DoS Complexity

- TCP/IP protocol suite
- Protocol + protocol field settings = many variations of DoS
- Primitive attacks: lower 4 layers
- Sophisticated attacks: 7th layer

Evaluating the Detection

- Penetration testing
  - Need test cases
    - Specific to type of attack
    - Attack signature
  - Ideally common across industry
  - DARPA

Data Sets

- DARPA 98, 99, 2000….. Why no more?
- Few updated/available
- Could be valuable asset to cyber security and network technology development
- Must be complete and exhaustive

Initial Strategy

- Show that I can detect attacks represented by DARPA data set
- Show that there exist DoS attacks not present in DARPA data set
- Create suitable signatures for the new attacks

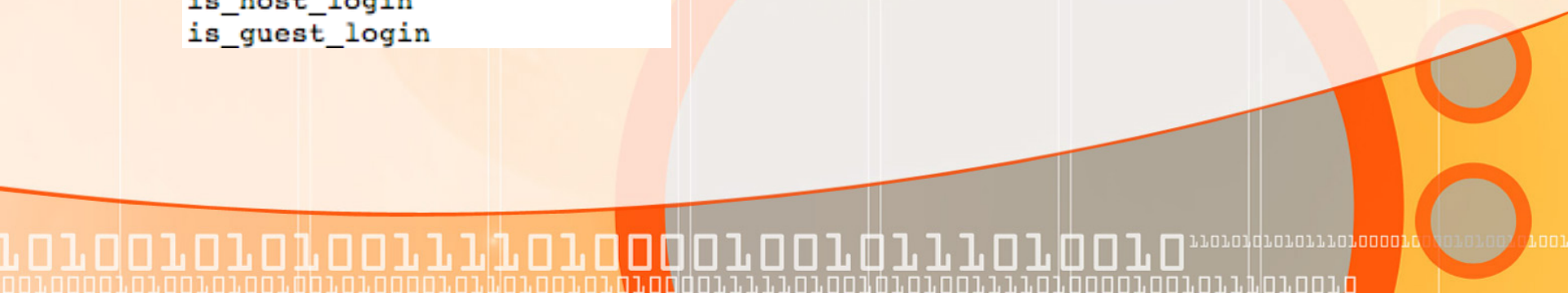
Reality

- DARPA data set not tailored to this kind of approach
  - Not ready to be sent into the test bed network
  - More beneficial to machine learning
  - Majority of time spent learning how to inject the attack into the network

What the data looks like

```
duration
protocol_type
service
flag
src_bytes
dst_bytes
land
wrong_fragment
urgent
hot
num_failed_logins
logged_in
num_compromised
root_shell
su_attempted
num_root
num_file_creations
num_shells
num_access_files
num_outbound_cmds
is_host_login
is_guest_login
```

```
srv_count
serror_rate
srv_serror_rate
rerror_rate
srv_rerror_rate
same_srv_rate
diff_srv_rate
srv_diff_host_rate
dst_host_count
dst_host_srv_count
dst_host_same_srv_rate
dst_host_diff_srv_rate
dst_host_same_src_port_rate
dst_host_srv_diff_host_rate
dst_host_serror_rate
dst_host_srv_serror_rate
dst_host_rerror_rate
dst_host_srv_rerror_rate
```



Original Approach

- Experiment test bed
- Show that test bed can detect the known attacks
- Develop signatures for the attacks not known
- Show that test bed can detect the new signature(s)

What I was Hoping to do

- Take the traffic log from data set and translate into an attack
- And have the IDS detect the attack

Or

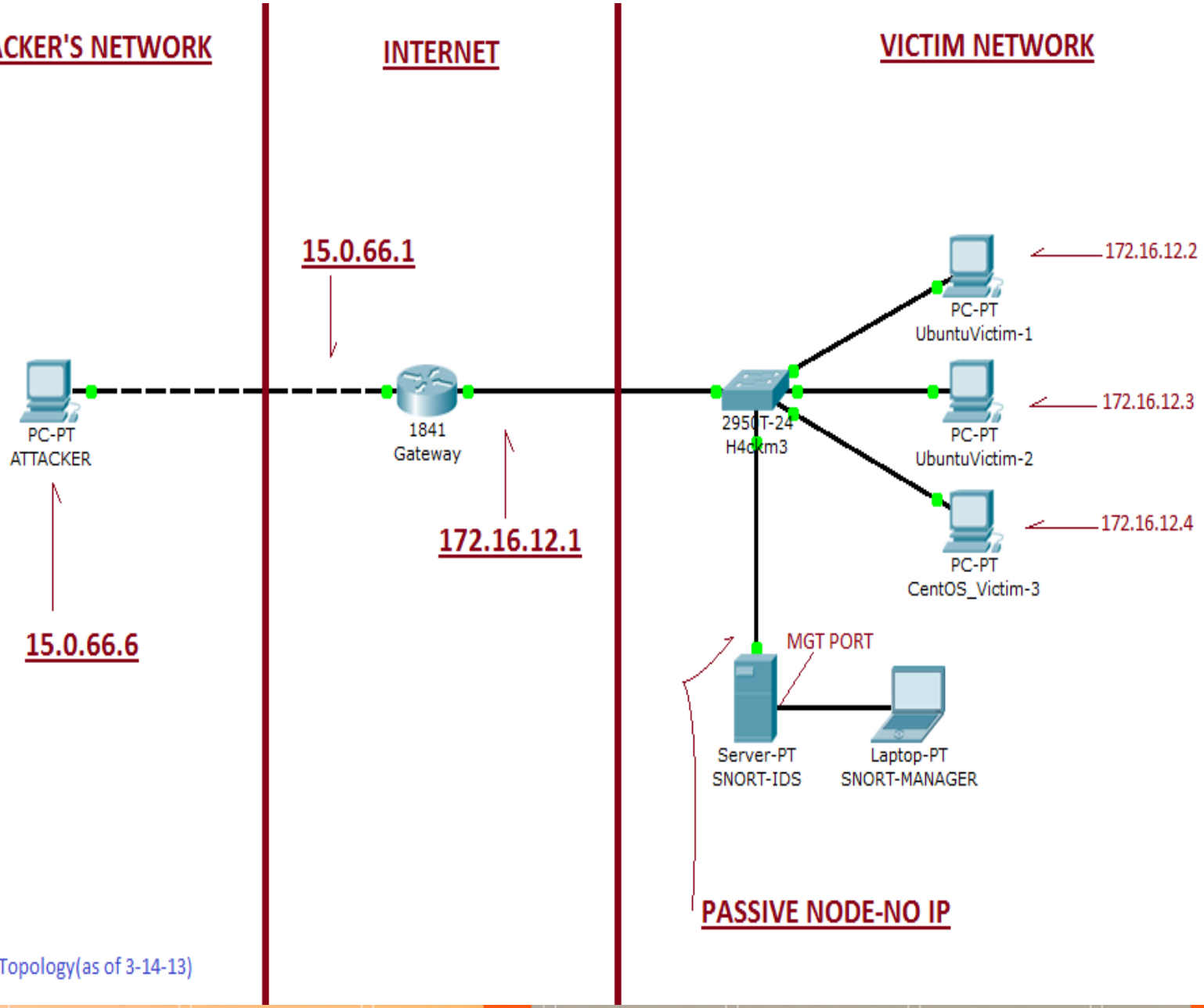
- Use IDS rules to translate into an attack

Difficulties

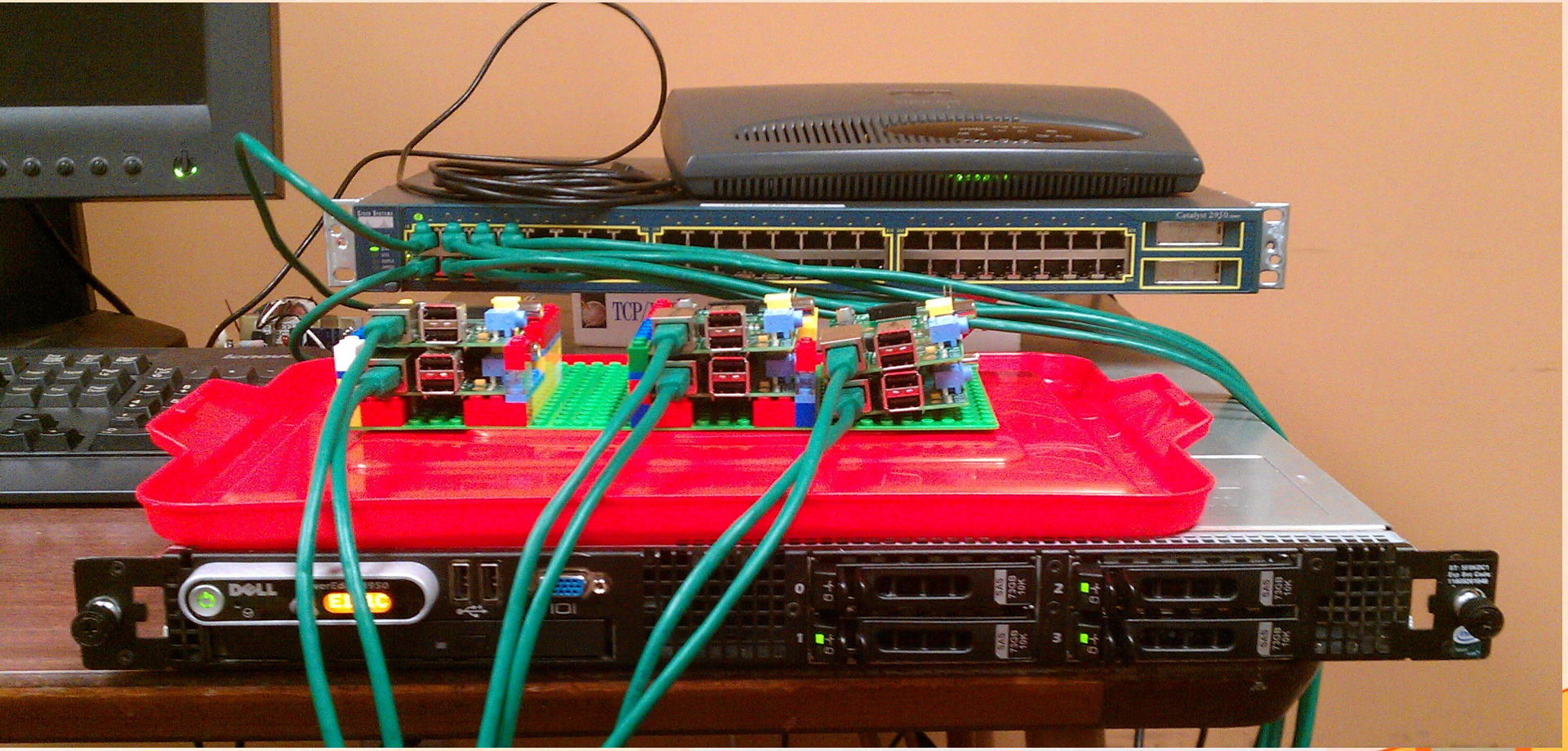
- The data sets and attacks are not easily translated.
- Snort IDS rules not easily translated into attacks

Lessons Learned

- This is not easily done
  - For data sets
  - From rules



Test Bed Topology(as of 3-14-13)


DELL
EMC
SAS
73GB
10K

Reality Part 2

- Using python CLI and Scapy.py
- Creating specific IDS rules for each attack that I'm testing

Test Bed

- Attacker
- Cisco 1600 router
- Cisco 2950 switch
- SNORT IDS
- Victim Network

End Result

- Test bed fully functional
- Constructed 3 distinct attack signature/ payloads
- Future work
  - Constructing attacks not in the data set.

Special Thank You

- Professor Barr
- Professor Hedrick
- Dan Mahar and Chris Cooke – ITS
- Union College Undergraduate Research Program

Questions?