A Cloud Based Entitlement Granting Engine

David Krovich
Lane Department of CSEE
West Virginia University

About Me

- Research Associate Faculty in the CSEE Department at West Virginia University
- Research areas are cybersecurity and open source software
- Late bloomer, got my MSSE in 2016
- Spent previous life as a unix/linux sysadmin for roughly 20 years
- Grew up in central Pennsylvania
- Lived most of my adult life in Morgantown, WV
- 1 year stint in Raleigh/Durham during the dot.com boom
- 1 year stint in San Francisco Bay area while I was getting my MSSE

Introduction

- The need for cybersecurity professionals is well documented
- Training cybersecurity professionals brings with it many challenges.
- Challenge Based Learning (CBL) is one method for teaching cybersecurity that shows promise
- Cybersecurity CBL methods require a lot of infrastructure to give hands on experience
- Configuration and maintenance of cybersecurity training environments presents a barrier to effectively teaching cybersecurity

Motivations

- Collegiate Cyber Defense Competition (CCDC)
- Needed a way to cheaply give students exposure to real world training simulations to teach cybersecurity concepts
- Needed a way to quickly setup and teardown training simulations

Approaches to training

- Have a lab full of computers with the software and environments needed.
- Use virtual machines and distribute images via OVAs
- Manually configure instances in AWS using the web interface
- Setup you own private cloud on premises and use that for training.

Downsides

- Administrative access in shared environments can be difficult
- Transferring Virtual Machine images can take significant amount of times
- Manually configuring in AWS is time consuming and error prone
- Private clouds cost money and need people to maintain them.

Solution: Use the cloud and APIs

- Allows for an on demand pay as you go model
- Has many different preloaded operating systems and pre configured systems and environments
- Has tools to build complex network topology environments. Can replicate on premise setups in the cloud
- Comes with programming APIs to access cloud resources from most languages such as java, python, and ruby

Cyber Security Simulation Portal (CSSP)

- Started as a group project for a graduate MSSE course taken in the Summer of 2016
- Provides a web portal for controlling and delegating access to cloud resources to gmail email addresses
- Ruby on Rails application
- Uses ruby to interact with Amazon Web Services API
- Uses OAuth2 with google as the backend for authentication
- Renamed to Cyber Software Sandbox Portal in early 2019.

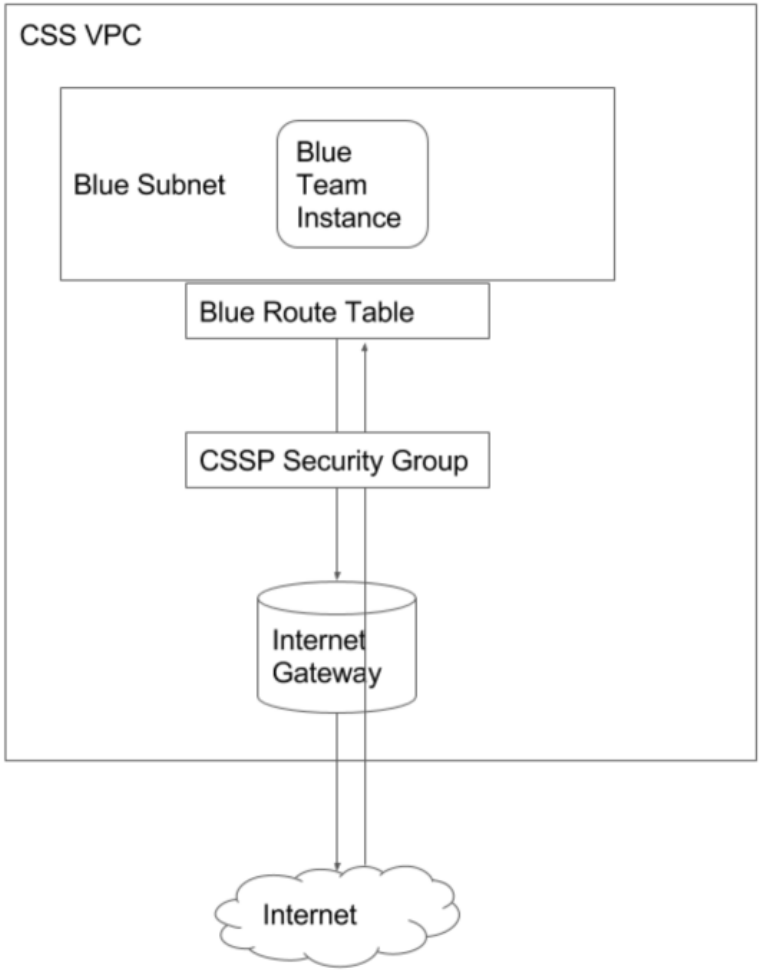
Cyber Sandbox Software Portal

- Recasting of “Cyber Security Simulation Portal”
- Goal is now to deliver cloud based sandbox environments
- Can be used outside of cybersecurity.

CSSP Functionality

- Welcome Controller - Handles main landing page for the application
- Omniauth_callbacks Controller - Interfaces with google OAuth2 for authentication
- Blue Team Controller - User level role. Provides connection information about instances that have been assigned to a user.
- AMI Controller - Handles interfacing with AMIs managed by the CSSP application
- User2_Instance_Maps Controller - Manages the mapping of instances to users
- Instances Controller - Used to create, terminate, start, and stop instances

CSSP VPC


CSS VPC
Blue Subnet
Blue
Team
Instance
Blue Route Table
CSSP Security Group
Internet
Gateway
Internet

CSSP Identity and Authentication Management

- Uses devise, which is a ruby gem used for identity and authentication
- Uses devise-google-oauth2 plugin to interface with OAuth2 and Google
- Allows blue role users to login with their gmail credentials
- Allows admin role users to associate instances to gmail email address

Under the hood of OAuth2

- Identity Provider (IDP)
- Relaying Party (RP)
- Typically, RP registers a developer level account with the IDP
- Developer registers application with IDP. Typically need to provide a name, a application URL, and a callback URL
- IDP grants RP secret keys
- RP integrates secret keys into application. This then uses the Client Credentials Grant Flow of OAuth2 and implements the IDP into the application
- When a user logs in, they are sent to the IDP. If they authenticate the IDP sends back OAuth2 keys and the application will load identify information for the user

CSSP Access Control

- The CSSP application itself is opened to IP ranges as needed
- Instances created by CSSP that live within the simulation are all assigned the default CSSP security group
- When an instance is associated to the user, an additional security group is created with the name of the user
- When a user logs into CSSP, CSSP logs the IP where that user is connecting from and then populates the user security group with that IP.

Simulation Costs

- 0.012/hr for linux instances
- 0.065/hr for windows server instances
- Scenario with 10 linux servers and 10 windows servers for 2 hours would cost roughly (0.012 * 10 * 2) + (0.065 * 10 * 2) = $1.54

Acceptable Use Policy of Cloud Resources

- AWS Acceptable Use Policy: https://aws.amazon.com/aup/
- Some cybersecurity training activities come in direct conflict with the acceptable use policy
- AWS Customers can request authorization for cybersecurity type testing
- https://aws.amazon.com/security/penetration-testing/

Live Demo (launch ncs19 host)

- Launch EC2 instance in Virginia from ncs19 prebuilt AMI
- Make sure security group allows me to connect via ssh
- Assign IP via Route 53 (for later) (needed for OAuth2)

Live Demo (Configure environment variables)

```
# source this file
export AWS_ACCESS_KEY_ID=
export AWS_SECRET_ACCESS_KEY=
export GOOGLE_CLIENT_ID=
export GOOGLE_SECRET=
export CSSP_ADMIN_EMAIL=
export REGION=
```

Live Demo (Configure environment variables)

- scp file containing environment
- Source environment
- Start CSSP container

Live Demo (Configure VPC)

- Delete old VPC from us-west-1
- ruby initialSetup.rb

Live Demo (Configure CSSP Application)

- # setup-cssp
- # start-puma

Live Demo

- Connect to app
- http://ncs19.lcseecloud.net:3000
- Show login on Chrome Browser in incognito mode

Live Demo

- Admin user logs in, launches default AWS ubuntu 18.04 image from AMI and injects a public key
- Admin user associates instance to their email address
- Switch to blue team controls and get connection information
- Connect to instance
- Modify instance to allow password logins
- Create new AMI from modified instance.
- Delete running instance
- Launch new instance from AMI that was just created
- Associate new instance to email address of admin user
- Switch to blue team controls and get connection information and connect

Future work for CSSP

- Manipulating security group information from CSSP
- Automatic shutdown of instances after a specified time period
- Use Cloud Formation or something similar to deploy sets of machines
- Simplify deployment
- Release as an open source software project

Questions?

dmkrovich@mail.wvu.edu