# Lab Four: Release Engineering and Web Site Updates

## Introduction/ Use Case:

In our last lab we deployed a Load balancing architecture with three web servers and one load balancer. This was our first real operational environment and a major step forward for us. In this lab, we are going to see what it means to use this architecture as a real operational environment. To do this we are going to create the Ansible playbooks/roles that can be used to update our Explore California web site with new content. The problem is that we want to do this while it is still in production. Rather than just push new files out to each of our web servers at the same time we are going to do what is called a blue/green deployment. In a blue/green deployment what you do is illustrated below:



Starting with the web architecture you had in Lab 3 and a pool of available servers at your load balancer as shown.



Add a new set of servers configured exactly as the ones you had below, except with the new content that you want to serve. Notice, we have not changed the pool of servers being used by the load balancer. Until we change this the new servers are not being used for production and we can mess around with them to get them ready.



Above, we have replaced the file that contains the pool of available servers at the load balancer and thereby transferred the incoming load from the old servers to the new servers. The old servers can now be deleted. I would like you to go through this process four times. To come up with four versions of Explore California all you have to do is change the index.html to include version 1, version 2, version 3, and version 4. You should be able to cycle through this. Doing all of this with RLES might be a little cumbersome, but use your imagination and you can figure out a way to make it work.

## Lab Outcomes:

Operations Team Outcome One:

Develop a Visio drawing to include the new process of release engineering. In your diagram show how a new version of your application moves into and replaces an old version. Be sure to include in your drawings such things as naming schemes and version numbers, example IP addresses, example operating systems, network connections, etc.

### Deliverables:

Visio drawings showing the migration of new servers into production and old servers out of production.

## Operations Team Outcome Two:

### Release Engineering Process

Develop the Ansible/Bash scripts that can be used to reliably deploy new versions of the Explore California web site to multiple web servers behind a single load balancer. Be sure to start from the scripts that you used to create your fleet of monolithic web servers. Your goal in this is to create a production ready process that satisfies the security requirements of your Security Team. Remember: what you are building is a system that can be reused with little modification for ongoing operations and deployments of new versions of your application. Because of the peculiarities of working with RLES this is not going to be perfect, but do the best that you can.

The process you develop might be something like the following:

1. start the new servers,
2. copy the new files from a location where your developers could post them onto the servers,
3. making any other configuration changes that are required
4. pull the old servers out of the monitoring system so you don’t get any false positives when you take them down
5. add the new servers to the monitoring system
6. remove the old servers from the load balancer list of available servers
7. add the new servers to the list of available servers
8. delete or reimage the old servers so they could be put back into production when needed

### Deliverables:

A process is developed that allows you to copy new versions of Explore California from a set location to new servers and then migrate those servers into a production role and also migrate the old servers out of production. This can be done by modifying the pool of servers available to receive incoming connections at your load balancers. The goal is to do this with as little disruption to your ability to continue to receive incoming connections as possible.

## Security Team Outcome One:

This is a pretty dangerous thing that I am tasking the Ops team with developing. Probably the only more dangerous thing would be to not have a process like this in place and just randomly update our application without having thought through how to do this. So, your task as a sec team is to review the process that the Ops team has developed and make recommendations to make the process more “secure”. For this exercise, “secure” can mean things like more reliable (an unreliable/difficult process could dangerously impact the availability of your site). The ultimate process would be so simple and repeatable that it would involve only a single click and automatically update the site.

For this lab your job is to review a process and test it to make sure that it functions as it should.

## Security Team Outcome Two:

After you have done this your second task is to write a script (this could be an Ansible based script, a Bash script, or any other programming language) to test to see if the process that your Operations Team creates lives up to the security policies you have developed. Remember: Availability is a security requirement. Be sure to include this test of availability in the security testing you develop.

## Results:

Be sure to include the following documents in your Lab Four submission:

1. Project Plan with Visio drawing.
2. Operations Report of Functionality.
3. Security Report of Functionality.
4. Group Reflection Report