CS 6480: Advanced Computer Networks - Fall 2013

Lab Assignment 2:
Emulab - Cloud Computing

October 6, 2013

There are three goals associated with this assignment. First is to introduce students to the Emulab network testbed [1]. The second is to use Emulab to explore Cloud Computing through the Eucalyptus open source cloud platform [2]. The third is to use a Eucalyptus instance in the Emulab environment to replicate some results from a cloud computing research paper. Specifically, you will be required to implement a building block that could be used as part of virtual machine co-residency detection mechanism as described in [3].

Assignment Overview

The economies of scale, apparent access to unlimited resources and the management flexibility that comes with cloud computing, have made this technology a mainstay in modern data centers. With this success, however, comes concerns about new attack surfaces unique to the shared-tenant environment provided by cloud platforms.

A specific concern involves the possibility of an attacker being able to arrange for a VM under her control to be co-resident with that of a target VM [3]. Once co-resident with the target VM an attacker's VM might perform denial of service attacks, attempt to circumvent the hypervisor isolation mechanisms or try to exfiltrate information from the target VM.

For this assignment you will use the Emulab testbed infrastructure [1] combined with the Eucalyptus open source cloud environment to learn about cloud computing and to use that environment to develop code that can test for VM co-residency based on a hard-disk based covert channel approach. (The latter inspired by the cache-based co-residency test described in [3].)

The assignment will consist of the following three phases, with deliverables for each phase:

- Eucalyptus installation (phase 1): You need to install a three node instance of Eucalyptus on Emulab consisting of a cloud controller (CLC) and a storage controller (Walrus) on one physical node, a cluster controller (CC) on another node and a node controller (NC) on a third node. For your first deliverable you will need to provide evidence that you successfully installed the cloud system and understand its basic operation.

- Hard-disk based covert channel (phase 2): For this phase you need to develop the essence of a hard-disk covert channel. Your covert channel implementation does not need to be sophisticated enough to support communication. Rather, your listener (or receiver) should be sensitive enough to distinguish periods of high and low (or none) disk activity from the sender (or transmitter). You will need to provide evidence that you have implemented this functionality in your second deliverable.

- **Optional** Hard-disk based co-residency test (phase 3): The software developed in the second phase can be used as a building block for a load-based co-residency detection mechanism (along the lines of Section 8.2 of [3].) E.g., you can generate HTTP load against a Web server running on a "target" VM, and run a variant of your covert-channel listener on a VM you control to determine whether the two VMs are co-resident on the same physical machine. For this approach to work, the HTTP load will have to cause disk IO related load on the target VM, which would not be true in general for a Web server. This functionality can optionally be implemented for **extra credit**.

Assignment Details

This is a non-trivial assignment which will acquaint you with a variety of concepts, several of which might be new. As outlined above, to help you make progress in a structured manner the assignment has been divided into three parts.

- Phase 1: Eucalyptus installation:
  - Request to get access to Emulab:
    - Go to: https://www.emulab.net/
      (For some reason this website doesn't work for me with Safari...Firefox seems to work fine.)
    - Select "Request Account" on the left.
    - Select "Join an existing project"
    - Fill out the form. Do create an ssh key and upload the public part, even though it is not required, it simplifies things later on. Use "cs6480" for the project name.
  - Briefly read the Emulab tutorial:
    https://wiki.emulab.net/wiki/Tutorial
  - Create a simple Emulab experiment, e.g., two nodes connected via a LAN, and make sure you can log into the nodes and that you are familiar with the Emulab procedures. In particular be aware of the following:

    Anything created on the local file system on your node **WILL BE LOST** when your experiment is swapped out.

    If you want to store things persistently within Emulab you can use: /users/YOURUSERNAME, or /proj/cs6480. If you are using the latter, create a subdirectory with your username.
  - If you have not yet done so, read the Eucalyptus paper [2].
  - Read through the Eucalyptus installation guide:
    http://www.eucalyptus.com/docs/eucalyptus/3.3/install-guide/
    Pay particular attention to:
    - All of "Introduction to Eucalyptus".
    - All of "Planning Your Installation".
    - "Configure Bridge" under "Configuring Dependencies".
    - "Installing Eucalyptus from Release Packages" under "Installing Eucalyptus".
    - "Configure Network Modes" under "Configuring Eucalyptus". (You will be doing "Static Mode".)
    - "Starting Eucalyptus" (CLC, Walrus, CC, NC).
    - "Registering Eucalyptus" (Walrus, CC, NC).
    - "Generate Administrative Credentials" under "Configuring the Runtime Environment".
  - Read through the following sections of the administrative guide:
    http://www.eucalyptus.com/docs/eucalyptus/3.3/admin-guide/
    - "Image Overview" and "Image Tasks" from "Managing Images".
  - Read through the following sections of the "Using Instances" section of the user guide:
    http://www.eucalyptus.com/docs/eucalyptus/3.3/user-guide/

    * “Instance Concepts” from “Instance Overview”.
    * All of “Instance Tasks”.
- To get full credit for this part of the assignment you should be able to perform at least the following with your Eucalyptus installation:
    * Install Eucalyptus from release packages.
    * Configure Eucalyptus using the “Static Mode” for networking.
    * Start up and register the following Eucalyptus components: CLC, Walrus, CC and NC(s).
    * Create administrative credentials.
    * Install an example image from EuStore.
    * Create a key pair.
    * Associate your key pair with an image.
    * Start up an instance and log into it using your private key (from key pair).
- Now that you know what to do you can create an Emulab experiment to do the installation. Here is an example NS file:

```
# This is a simple ns script. Comments start with #.
set ns [new Simulator]
source tb_compat.tcl

set clc [$ns node]
set cc [$ns node]
set nc1 [$ns node]

set lan0 [$ns make-lan "$clc $cc $nc1 " 100Mb 0ms]

tb-set-hardware $clc d710
tb-set-hardware $cc d710
tb-set-hardware $nc1 d710

# Set the OS on a couple.
tb-set-node-os $clc KExplore/CENTOS63-64-BETA-kvm
tb-set-node-os $cc KExplore/CENTOS63-64-BETA-kvm
tb-set-node-os $nc1 KExplore/CENTOS63-64-BETA-kvm

set bs1 [$ns blockstore]
$bs1 set-class "local"
$bs1 set-placement "sysvol"
$bs1 set-mount-point "/var/lib/eucalyptus"
$bs1 set-node $clc

$ns rtproto Static

# Go!
$ns run
```

- Couple of notes about the NS file:
    * The CLC node should be used for both CLC and Walrus.
    * You need physical machines that are capable of virtualization. d710 and d820 nodes in Emulab are suitable for this. You indicate this to Emulab using the $tb - set - hardware$ directive.

    * The host (physical machine) OS needs to have KVM installed. Also, to make the installation process more closely resemble the default Eucalyptus process, we will be using a special Emulab image which only configures the Emulab control interface (not the experimental interfaces). You indicate this image to Emulab through the $tb-set-node-os$ directive.
    * Eucalyptus requires significant storage on the CLC/Walrus node. The NS file requests to use all available local storage on the CLC node (using the *blockstore*) directive.
- As mentioned earlier, unlike with regular Emulab images, the experimental interfaces on the $CENTOS63-64-BETA-kvm$ image will not be configured. A specific interface (out of a relatively large set) will be assigned to your experiment and the Emulab network will be configured to create the network you requested. You will need to configure the experimental interfaces on your nodes as follows:
    * To find out the interface and parameters assigned to your node do the following:

      ```
      cat /var/emulab/boot/tmcc/ifconfig
      ```

      This file will contain a single line like this:

      ```
      INTERFACE IFACETYPE=bce INET=10.1.1.3 MASK=255.255.255.0
      MAC=0024e87a436a SPEED=100Mbps DUPLEX=full IFACE= RTABID=0 LAN=lan0
      ```

      Take the MAC address shown and find the physical interface associated with this. E.g.:

      ```
      /usr/local/etc/emulab/findif 0024e87a436a
      ```

      The output is the physical interface you are looking for. E.g., "em4".
    * You will use this information to follow a variant of the Eucalyptus "Configure Bridges" instructions, under "Configuring Dependencies" in the Installation guide. Specifically, **for each NC node**:
      From the above you have the info needed to create an ifcfg-* file in /etc/sysconfig/network-scripts. (The Eucalyptus default instructions say to modify the existing script, but there may not be one in our case since Emulab doesn't use these). So create, e.g. "ifcfg-em4" (assuming that the above indicated you have interface *em4*) with contents:

      ```
      DEVICE="em4"
      ONBOOT="yes"
      BRIDGE="br0"
      ```

      Then create the ifcfg-br0 like they say for the "static IP" case, and using the IP address and netmask from the tmcc/ifconfig output, e.g.:

      ```
      DEVICE=br0
      TYPE=Bridge
      BOOTPROTO=static
      IPADDR=10.1.1.3
      NETMASK=255.255.255.0
      ONBOOT=yes
      ```

      Run "service network restart".
    * For non-NC nodes you do not need to create a bridge. You would still need to configure the experimental interface though. E.g., assuming you got the above information for a non-NC node you would create /etc/sysconfig/network-scripts/ifcfg-em4 with:

      ```
      DEVICE=em4
      BOOTPROTO=static
      IPADDR=10.1.1.3
      NETMASK=255.255.255.0
      ONBOOT=yes
      ```

      Run "service network restart".

    * Verify network connectivity between all nodes using ping.

  - Note that to make your NC work with KVM you will need to uncomment the following lines in the eucalyptus.conf file:

    ```
    MAX_MEM="0"
    MAX_CORES="0"
    NC_WORK_SIZE=50000
    NC_CACHE_SIZE=50000
    CONCURRENT_DISK_OPS=4
    DISABLE_KEY_INJECTION="0"
    ```

  - You might find it useful to verify that VMs are running on the NCs by using libvirt virsh commands: http://libvirt.org/sources/virshcmdref/html-single/

    Note that you should not use these commands to instantiate your VM instances, that will be done by Eucalyptus. However, using virsh might be useful for debugging purposes.

  - Note that Emulab automatically swaps out an experiment that has been inactive for more than two hours. A good strategy might therefor be to script your installation procedure as you refine it. That way when you swap your experiment in you can simply run the scripts to take you to a known starting point in the installation process. **If you do follow this approach be sure to keep your scripts on a persistent file system. (E.g., /users/YOURUSERNAME.) Not on the Emulab node local file system which will be re-imaged on swap in.**

  - **Evaluation:** You will have to demonstrate your working Eucalyptus instance to the instructor or the TA. You will also need to document evidence of your successful installation and show that you understand the functioning of your setup in your report for this phase.

    You will need to demonstrate:

    * The health of your cloud. (E.g., using the $euca - describe - services$ command.)
    * That you have successfully installed an image. (E.g., using the $euca - describe - images$ command.)
    * That you have successfully run an instance. (E.g., by using the $euca - describe - instances$ command and showing that you can log into the instance.)

    Your report should briefly describe what you demonstrated and how it fits together. I.e., show that you understand what you did and demonstrated.

    Note that due to the fact that Emulab automatically swaps out experiments after 2 hours of inactivity you will need to plan your demonstration and coordinate with the instructor or TA to show your setup. We should have remote access to your experiment, but will need advance notice to be available.

- Phase 2: Hard-disk based covert channel:

  - In the prescribed Eucalyptus configuration, VMs on the same physical machine will share the local storage system on that machine, thus forming a shared resource that could be exploited to create a basic covert channel. Section 6 of [3] describes at a high level how such a disk-based covert channel can be implemented.

  - Your goal will be to implement a basic version of such a covert channel and provide evidence of its functionality. Evidence could take the form of your transmitter "sending" a continuous series of 0s and 1s, and showing a time series of disk read times in the receiver which shows corresponding shorter and longer read times.

  - **Evaluation:** You will have to demonstrate your working system to the instructor or the TA. You will also need to document details of your approach and evidence that it worked in your report for this section.

- **Optional** Phase 3: Hard-disk based co-residency test:
  - The goal of this part of the assignment will be to to use the listener you have developed in the second phase to attempt a version of load-based co-residency detection as described in Section 8.2 of [3]. Specifically, in stead of directly generating load by a sender in the "target" VM, you will generate load in the target VM by performing HTTP requests from an external system (e.g., another node or VM) under your control and attempt to detect this from your covert channel "listener". As with the previous phase, evidence could take the form of your external HTTP requester continuously generating a series of high and low (or no) load requests and showing a time series of disk read times in the receiver which shows corresponding longer and shorter read times.
  - **Evaluation:** You will have to demonstrate your working system to the instructor or the TA. You will also need to document details of your approach and evidence that it worked in your report for this section.

Grading and evaluation

What to hand in

Report You will need to submit, via Canvas, a report for each phase of the assignment. As described above, these reports should detail what you have done and provide evidence that you executed the relevant phase of the assignment successfully. Since the first phase, Eucalyptus installation, essentially involve following a (somewhat involved) recipe, your report should specifically provide evidence that you understand what you have done and how the Eucalyptus cloud functions. For the second and third phases, you will need to provide details about your approach, how you implemented it and, as said before, evidence that it functioned correctly.

One way to think of these reports is to compare it with the papers we have been reading: When you read a paper you typically do not have access to any artifact that the authors have generated to verify their claims. Rather, based on their description and the evidence (results) they provide, as a reader you decide whether you think the work was credible or not. In the same way in your reports you need to convince your reader (i.e., your instructor) that you have performed a credible execution of the assignment.

Format for phase 1 report You should describe your Eucalyptus setup and how that realizes the things you were required to demonstrate as part of the evaluation. This could include outputs from relevant commands used to show correct functionality.

Format for phase 2 (and 3) report If you do the optional phase 3 part of the assignment that should be included in a single report together with that of phase 2.

Your assignment report should be formatted as a "short paper" with citations. One to two pages should be adequate. Your report should include the following sections (you may use the latex template used for LA 1):

- Introduction: Where you will motivate what problem you addressed and why that is important.
- Design: Here you would briefly describe details of your approach *at the architectural level.*

  This would be a good section in which to provide a simple figure to explain what you have done.
- Implementation: Here you would briefly describe the actual implementation.
- Evaluation: Here you would provide results of your evaluation to show that your system worked.

- Discussion: Here you would discuss the pros and cons of your approach. E.g., you might want to point out limitations of your implementation. This would also be a good place to suggest "future work". I.e., how some of these limitations might be addressed etc.

Grading

Criteria	Points
Eucalyptus installation: Demo of working instance	40
Eucalyptus installation: Report	20
Disk-based covert channel: Working code	25
Disk-based covert channel: Report	15
(Extra credit) Co-residency detection: Demo	(10)
(Extra credit) Co-residency detection: Report	(10)
Total	100
With extra credit	(120)

References

[1] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 255–270, Dec. 2002. [Online]. Available: http://doi.acm.org/10.1145/844128.844152

[2] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The eucalyptus open-source cloud-computing system," in *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, ser. CCGRID '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 124–131. [Online]. Available: http://dx.doi.org/10.1109/CCGRID.2009.93

[3] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds," in *Proceedings of the 16th ACM conference on Computer and communications security*, ser. CCS '09. New York, NY, USA: ACM, 2009, pp. 199–212. [Online]. Available: http://doi.acm.org/10.1145/1653662.1653687