信息安全技术实验报告

姓名：梁莉莉

学号：2017011581

专业：软件工程

日期:2019/6/15

目录：

Lab3：基于 Vulhub 开源靶场实验（web 渗透测试实验）

一、实验目的（简要写，不必全部照抄实验要求文档中的内容）

通过 Vulhub 提供的环境，复现 Web 网站常见漏洞，实验者进行系列攻击实验，获取 Shell。

二、实验环境（如实给出实验平台）

1. VirtualBox 6.0.8
2. Ubuntu 18.04 系统
3. Docker 18.09.6

三、实验内容（主要是实验结果，不必全部照抄实验要求文档中的内容。实验结果截图需按照顺序标号并给出标题，且给出简单文字说明。）

1.实验环境搭建

1）VirtualBox 与 Ubuntu 的搭建过程及环境搭建过程中笔者所遇到的问题都已详细记录在个人博客中：
https://blog.csdn.net/liangllhahaha/article/details/91480071

2）安装及启动 Docker

a）安装 Curl：sudo apt install curl

```
liangll@liangll-VirtualBox:~$ sudo apt install curl
[sudo] liangll 的密码:
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
将会同时安装下列软件:
  libcurl4
下列【新】软件包将被安装:
  curl libcurl4
升级了 0 个软件包，新安装了 2 个软件包，要卸载 0 个软件包，有 337 个软件包未被
升级。
需要下载 373 kB 的归档。
解压缩后会消耗 1,036 kB 的额外空间。
您希望继续执行吗？ [Y/n] Y
获取:1 http://cn.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcurl4 a
md64 7.58.0-2ubuntu3.7 [214 kB]
获取:2 http://cn.archive.ubuntu.com/ubuntu bionic-updates/main amd64 curl amd64
 7.58.0-2ubuntu3.7 [159 kB]
```

b) 安装最新版 docker：curl -s https://get.docker.com/ | sh

```
liangll@liangll-VirtualBox:~$ curl -s https://get.docker.com/ | sh
```

c) 检查 Docker 版本:docker -v

```
liangll@liangll-VirtualBox:~$ docker -v
Docker version 18.09.6, build 481bc77
```

d) 启动 Docker 服务：service docker start

```
liangll@liangll-VirtualBox:~$ service docker start
liangll@liangll-VirtualBox:~$
```

e) 安装 pip：sudo apt install python-pip

```
liangll@liangll-VirtualBox:~$ sudo apt install python-pip
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
将会同时安装下列软件:
  build-essential cpp cpp-7 dpkg-dev fakeroot g++ g++-7 gcc gcc-7 gcc-7-base
  gcc-8-base libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libasan4 libatomic1 libc-dev-bin libc6-dev libcc1-0
  libcilkrts5 libexpat1-dev libfakeroot libgcc-7-dev libgcc1 libgomp1 libitm1
  liblsan0 libmpx2 libpython-all-dev libpython-dev libpython-stdlib
  libpython2.7 libpython2.7-dev libpython2.7-minimal libpython2.7-stdlib
  libquadmath0 libssl1.1 libstdc++-7-dev libstdc++6 libtsan0 libubsan0
  linux-libc-dev make manpages-dev python python-all python-all-dev
  python-asn1crypto python-cffi-backend python-crypto python-cryptography
  python-dbus python-dev python-enum34 python-gi python-idna python-ipaddress
  python-keyring python-keyrings.alt python-minimal python-pip-whl
  python-pkg-resources python-secretstorage python-setuptools python-six
  python-wheel python-xdg python2.7 python2.7-dev python2.7-minimal
建议安装:
  cpp-doc gcc-7-locales debian-keyring g++-multilib g++-7-multilib gcc-7-doc
  libstdc++6-7-dbg gcc-multilib autoconf automake libtool flex bison gcc-doc
  gcc-7-multilib libgcc1-dbg libgomp1-dbg libitm1-dbg libatomic1-dbg
  libasan4-dbg liblsan0-dbg libtsan0-dbg libubsan0-dbg libcilkrts5-dbg
  libmpx2-dbg libquadmath0-dbg glibc-doc libstdc++-7-doc make-doc python-doc
  python-tk python-crypto-doc python-cryptography-doc
  python-cryptography-vectors python-dbus-dbg python-dbus-doc
```

f) 安装 compose：pip install docker-compose

```
liangll@liangll-VirtualBox:~$ pip install docker-compose
Collecting docker-compose
  Downloading https://files.pythonhosted.org/packages/51/56/5745e66b33846e92a88
14466c163f165a26fadad8b33afe381e8b6c3f652/docker_compose-1.24.0-py2.py3-none-an
y.whl (134kB)
    100% |                                | 143kB 124kB/s
Collecting six<2,>=1.3.0 (from docker-compose)
  Downloading https://files.pythonhosted.org/packages/73/fb/00a976f728d0d1fecfe
898238ce23f502a721c0ac0ecfedb80e0d88c64e9/six-1.12.0-py2.py3-none-any.whl
Collecting backports.ssl-match-hostname>=3.5; python_version < "3.5" (from dock
er-compose)
  Downloading https://files.pythonhosted.org/packages/ff/2b/8265224812912bc5b7a
607c44bf7b027554e1b9775e9ee0de8032e3de4b2/backports.ssl_match_hostname-3.7.0.1.
tar.gz
Collecting docopt<0.7,>=0.6.1 (from docker-compose)
  Downloading https://files.pythonhosted.org/packages/a2/55/8f8cab2afd404cf5781
36ef2cc5dfb50baa1761b68c9da1fb1e4eed343c9/docopt-0.6.2.tar.gz
Collecting PyYAML<4.3,>=3.10 (from docker-compose)
  Downloading https://files.pythonhosted.org/packages/9e/a3/1d13970c3f36777c583
f136c136f804d70f500168edc1edea6daa7200769/PyYAML-3.13.tar.gz (270kB)
    100% |                                | 276kB 91kB/s
```

3) Vulhub 环境准备

a) 克隆 github 仓库：

git clone https://github.com/vulhub/vulhub.git

```
liangll@liangll-VirtualBox:~$ git clone https://github.com/vulhub/vulhub.git
正克隆到 'vulhub'...
remote: Enumerating objects: 111, done.
remote: Counting objects: 100% (111/111), done.
remote: Compressing objects: 100% (74/74), done.
remote: Total 7657 (delta 40), reused 93 (delta 32), pack-reused 7546
接收对象中: 100% (7657/7657), 84.96 MiB | 29.00 KiB/s, 完成.
处理 delta 中: 100% (2990/2990), 完成.
```

b) 进入 Weblogic 环境：cd vulhub/weblogic/weak_password/

```
liangll@liangll-VirtualBox:~$ cd vulhub/weblogic/weak_password/
```

c) 启动漏洞环境：docker-compose up -d

```
liangll@liangll-VirtualBox:~/vulhub/weblogic/weak_password$ docker-compose up -
d
Creating network "weak_password_default" with the default driver
Pulling weblogic (vulhub/weblogic:)...
latest: Pulling from vulhub/weblogic
6599cadaf950: Pull complete
23eda618d451: Pull complete
f0be3084efe9: Pull complete
52de432f084b: Pull complete
a3ed95caeb02: Pull complete
a2318f26c625: Pull complete
1aa642dd8cc1: Pull complete
b307208f8bf5: Pull complete
1dfbbdcc497d: Pull complete
a53e674a7606: Pull complete
5f06bb51fa3c: Pull complete
ff0ff72567f2: Pull complete
684862046025: Pull complete
abbf8d475455: Pull complete
848eb11ef744: Pull complete
2f3438f2b83b: Pull complete
8e5871e15571: Pull complete
Digest: sha256:275ec19477cfda389dc1c42158033e7e8c650dd4cba9f090ca0ba673902b73c9
Status: Downloaded newer image for vulhub/weblogic:latest
Creating weak_password_weblogic_1 ... done
```

2.WebLogic 弱口令登录

1） 查看本机 ip 地址

a） 安装 net-tools：sudo apt install net-tools

```
liangll@liangll-VirtualBox:~/vulhub/weblogic/weak_password$ sudo apt install ne
t-tools
[sudo] liangll 的密码：
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列【新】软件包将被安装：
  net-tools
升级了 0 个软件包，新安装了 1 个软件包，要卸载 0 个软件包，有 334 个软件包未被
升级。
需要下载 194 kB 的归档。
解压缩后会消耗 803 kB 的额外空间。
获取:1 http://cn.archive.ubuntu.com/ubuntu bionic/main amd64 net-tools amd64 1.
60+git20161116.90da8a0-1ubuntu1 [194 kB]
已下载 194 kB，耗时 3秒 (70.5 kB/s)
正在选中未选择的软件包 net-tools。
(正在读取数据库 ... 系统当前共安装有 120954 个文件和目录。)
正准备解包 .../net-tools_1.60+git20161116.90da8a0-1ubuntu1_amd64.deb  ...
正在解包 net-tools (1.60+git20161116.90da8a0-1ubuntu1) ...
正在处理用于 man-db (2.8.3-2ubuntu0.1) 的触发器 ...
正在设置 net-tools (1.60+git20161116.90da8a0-1ubuntu1) ...
```

b） 查看 ip 地址:ifconfig -a

```
liangll@liangll-VirtualBox:~/vulhub/weblogic/weak_password$ ifconfig -a
br-3b2efd8413a8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.18.0.1  netmask 255.255.0.0  broadcast 172.18.255.255
        inet6 fe80::42:82ff:fead:55e7  prefixlen 64  scopeid 0x20<link>
        ether 02:42:82:ad:55:e7  txqueuelen 0  (以太网)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 40  bytes 5047 (5.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:82:c7:25:d1  txqueuelen 0  (以太网)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::3a83:a6b5:375f:fba3  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:51:29:c7  txqueuelen 1000  (以太网)
        RX packets 2306904  bytes 2447948352 (2.4 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 665625  bytes 40557198 (40.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

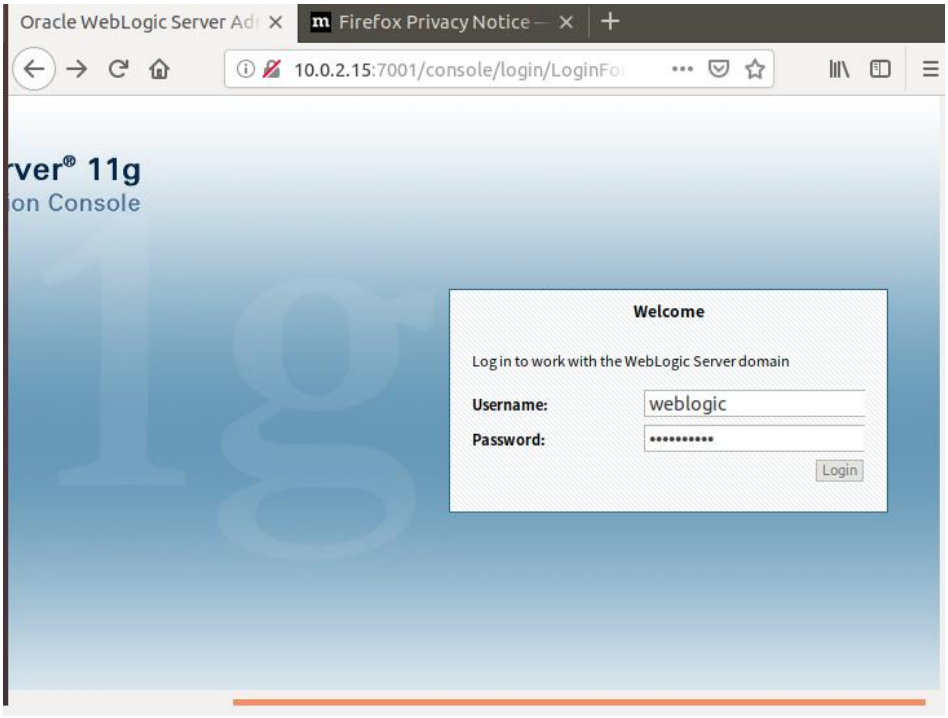
2） 访问 Weblogic 后台：

a） 网址：http://10.0.2.15:7001/console

b) 本环境存在弱口令：

Username: weblogic

Password: Oracle@123



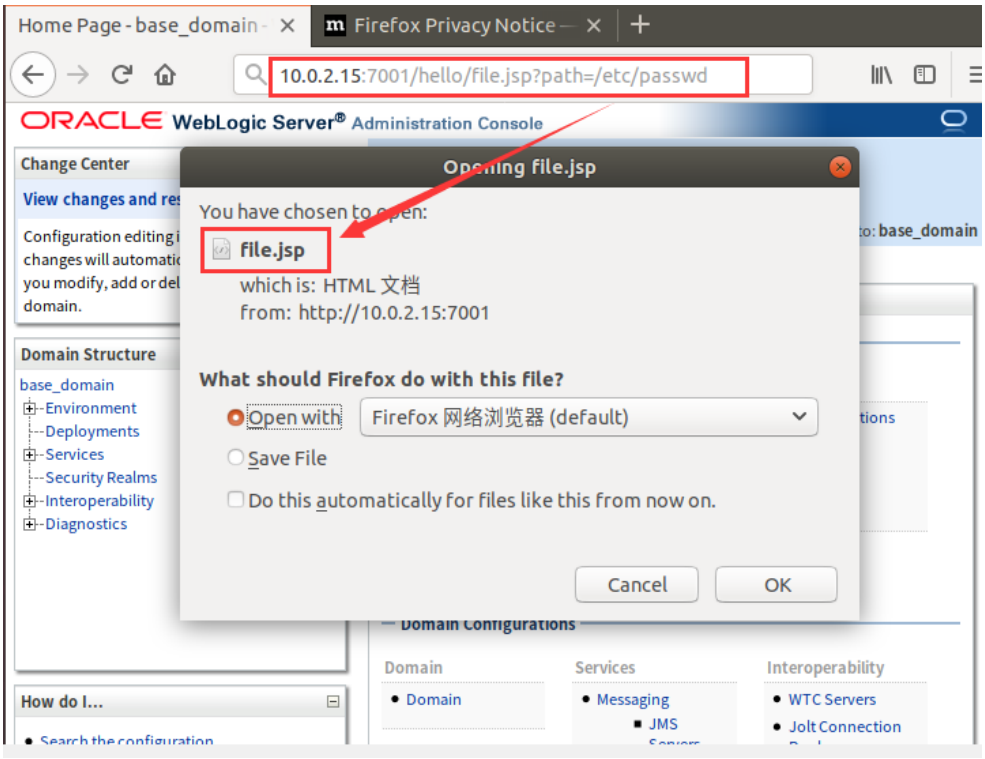
c) Weblogic 常用弱口令：

http://cirt.net/passwords?criteria=weblogic

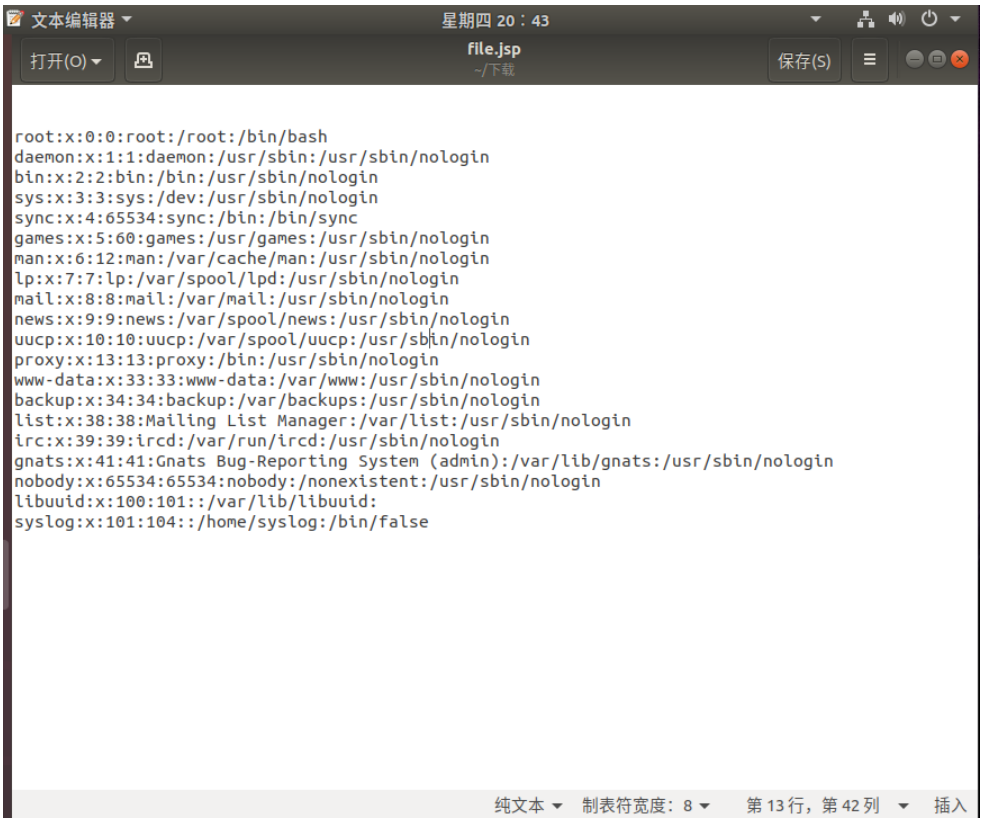
3.Weblogic 任意文件下载漏洞

1) 访问 http://your-ip:7001/hello/file.jsp?path=/etc/passwd

2) 获取并下载文件：



3) 成功读取 passwd 文件：



4.Weblogic 读取后台用户密文与密钥文件

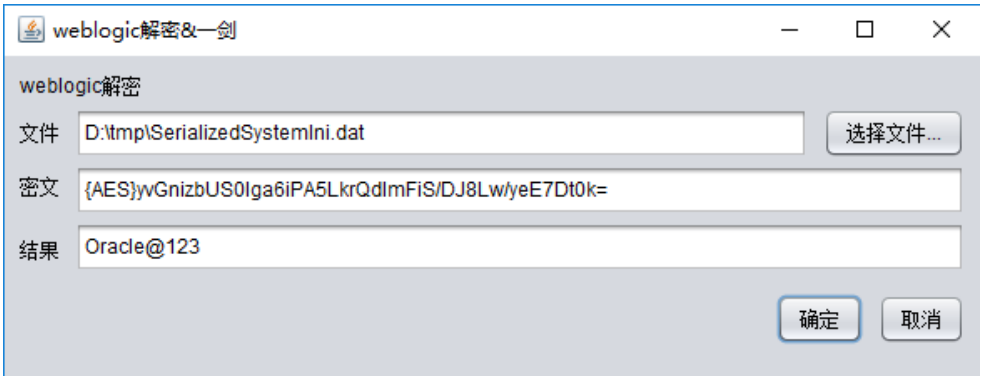
1) 读取文件

a) 基于当前目录：

/root/Oracle/Middleware/user_projects/domains/base_domain 读取名为 SerializedSystemIni.dat 和 config.xml 的文件

b) 解密密文：

使用 Vulhub 环境中的解密工具 weblogic_decrypt.jar 解密密文



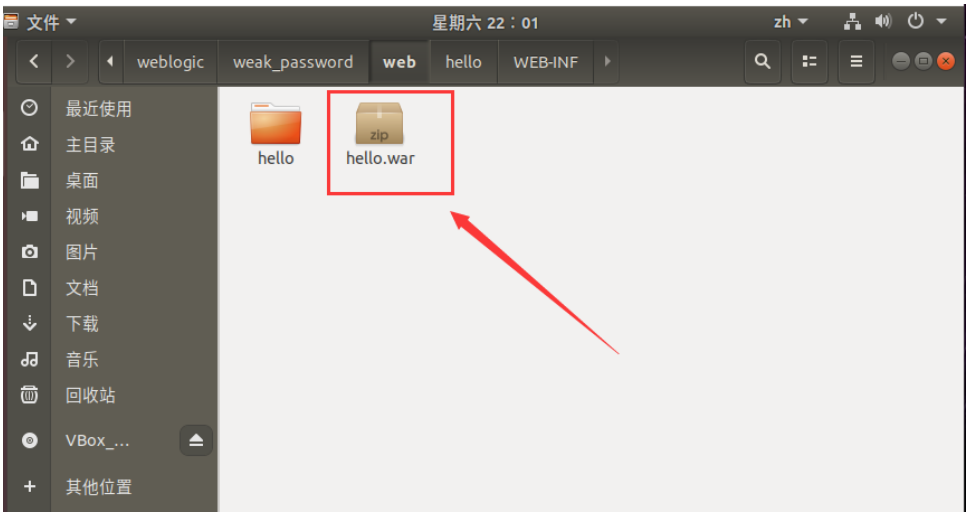
发现解密后的结果与预设登陆密码一致，说明成功。

5.Weblogic 后台部署 war 包获取 Webshell

1) 打包 war 包：

a) 使用 github 上 vulhub 项目里

weblogic/weak_password/web 目录里的 hello.war 包



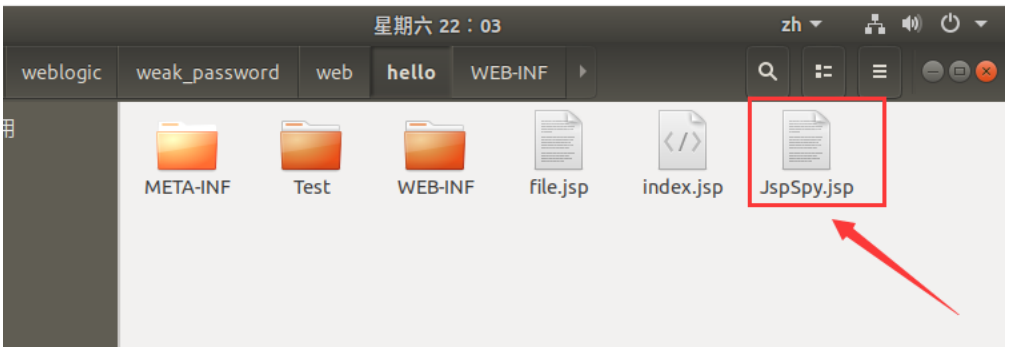
b）获取 Jspspy 大马：

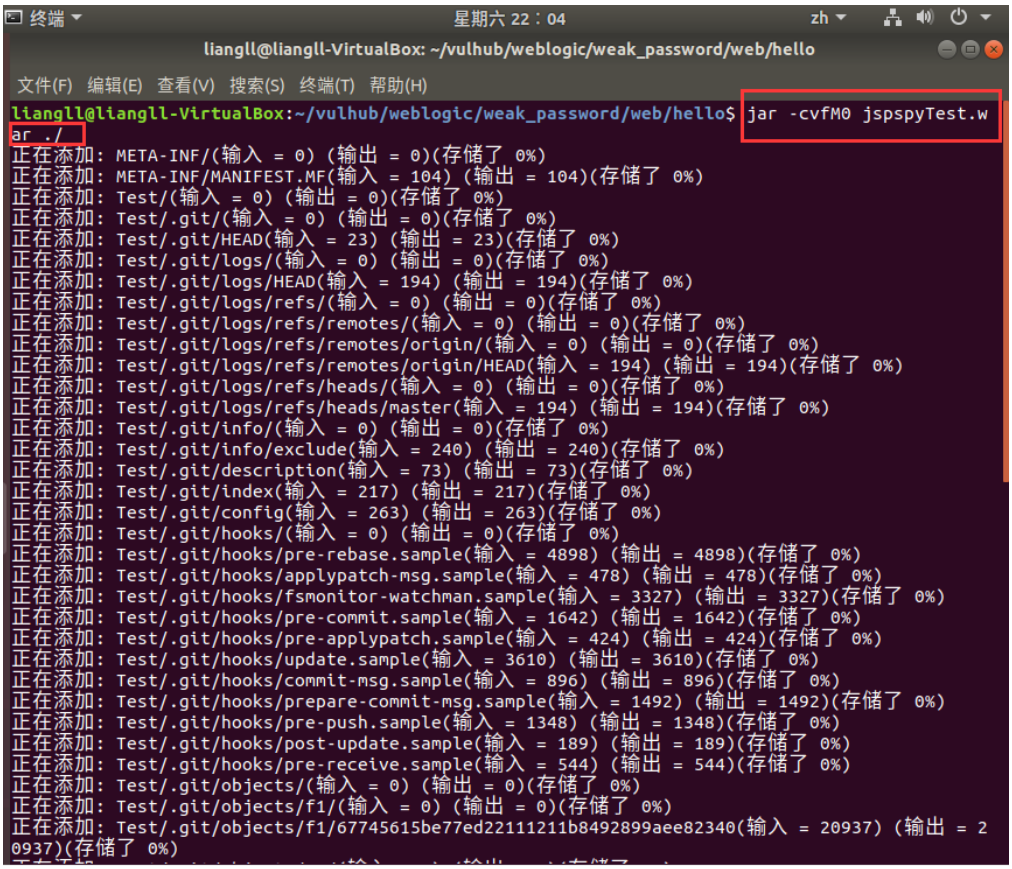
https://www.webshell.cc/4422.html

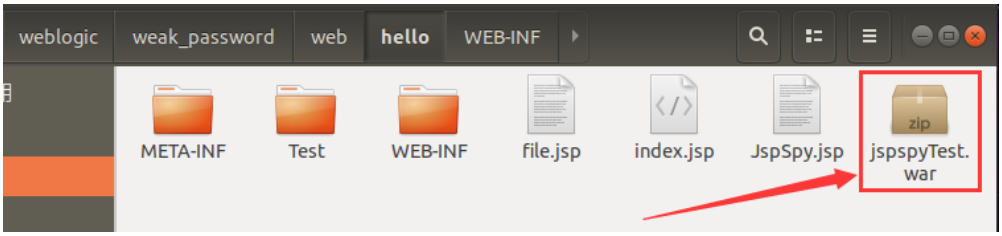
c）修改 war 包中 WEB-INF/weblogic.xml：

d) 将 jspspy.jsp 大马放到 hello.war 包中：

打包：jar -cvfM0 jspspyTest.war ./



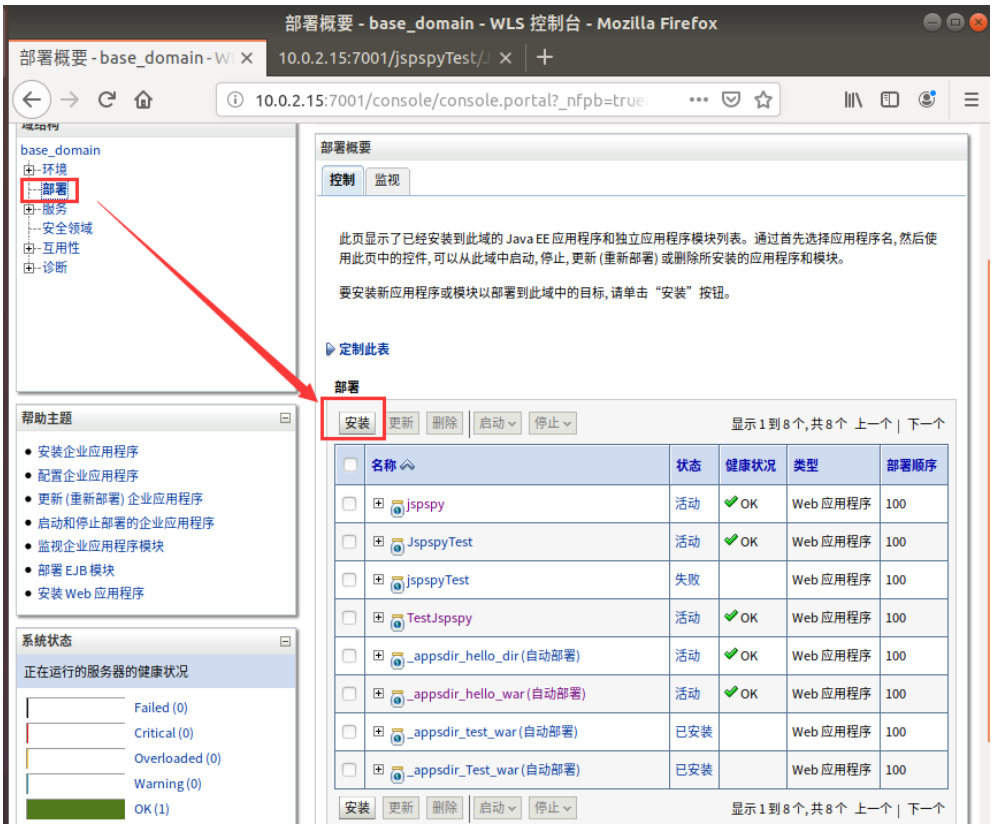




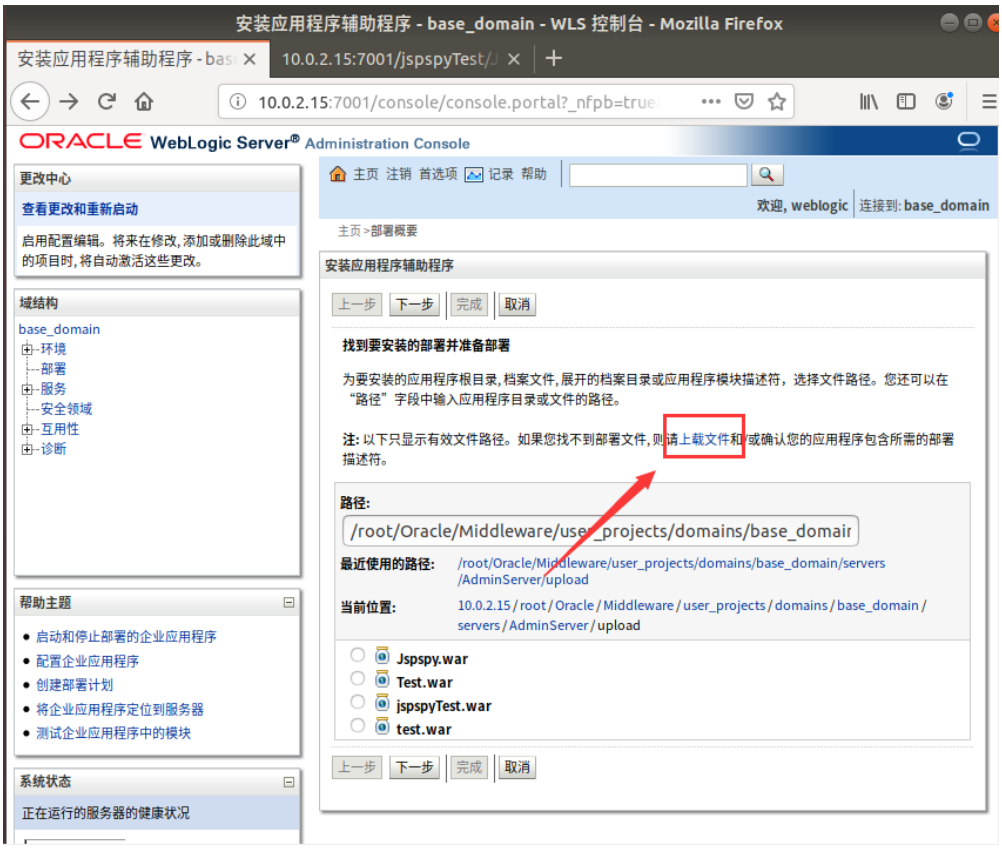
2） 开始部署

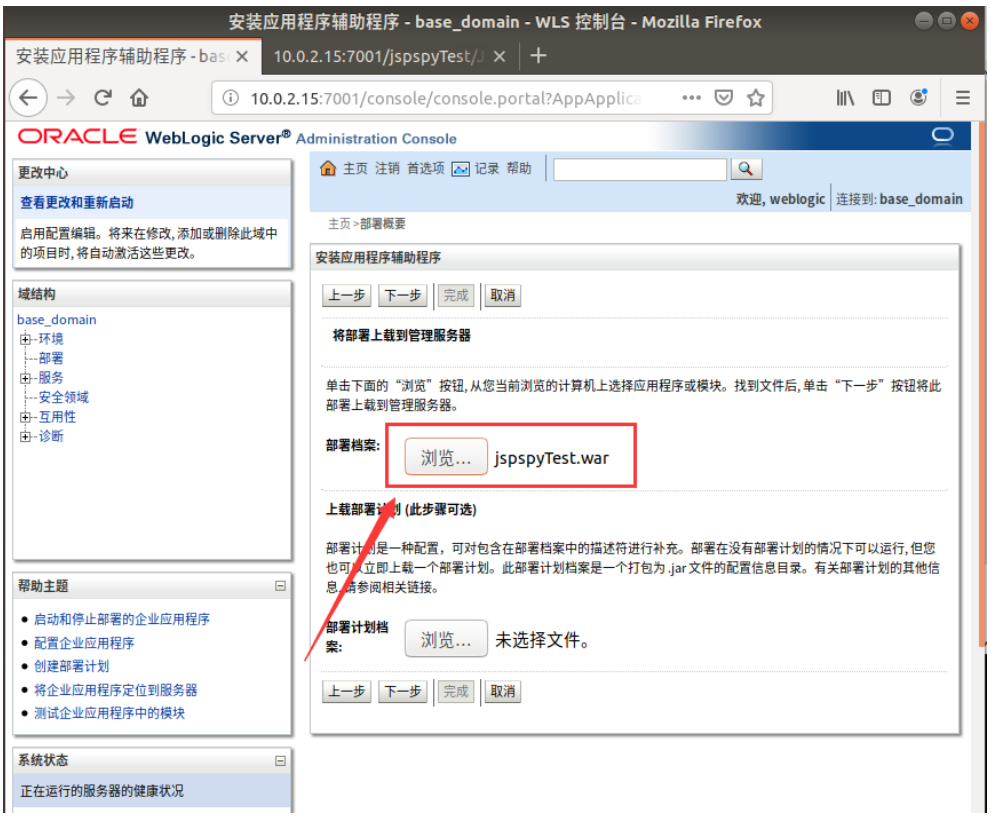
a） 进入网址：

a） “部署”➔“安装”：

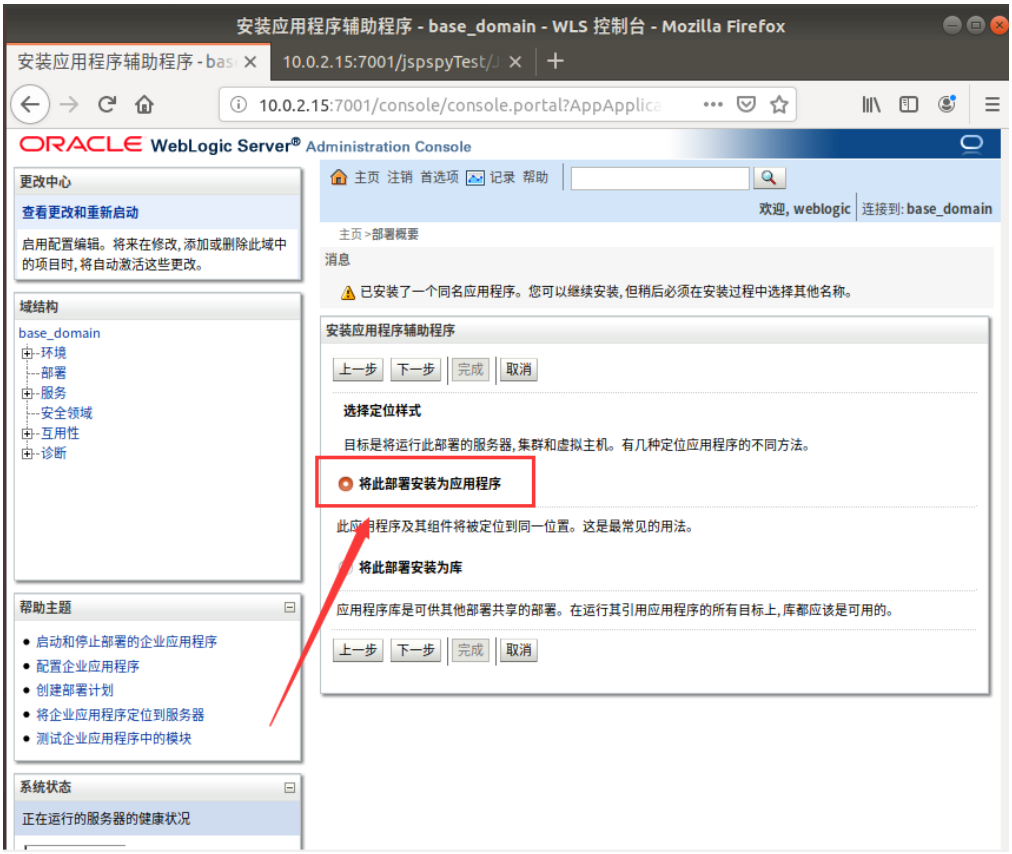


b） “上传文件”➔“选择文件”➔“下一步”：

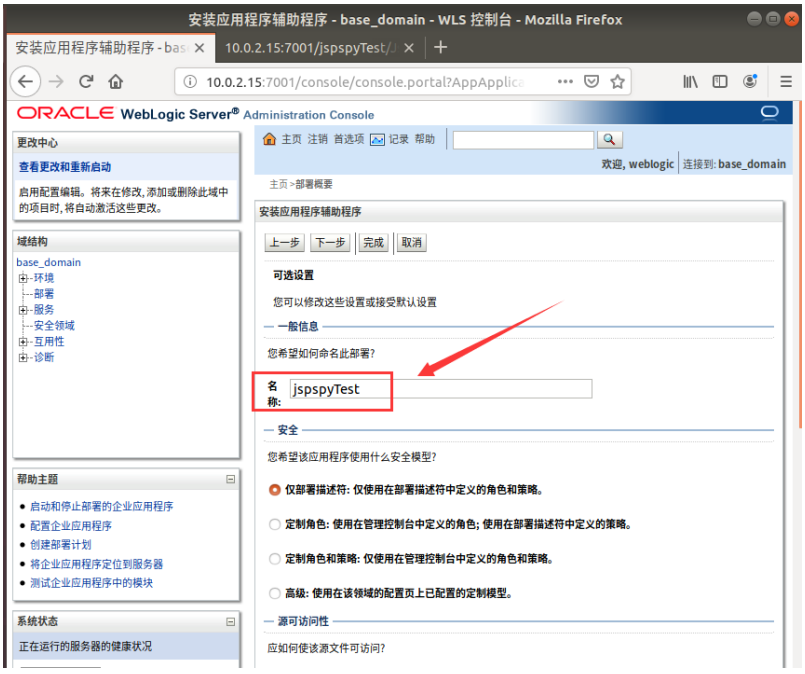

安装应用程序辅助程序 - base_domain - WLS 控制台 - Mozilla Firefox
10.0.2.15:7001/console/console.portal?_nfpb=true
ORACLE WebLogic Server® Administration Console
更改中心
查看更改和重新启动
域结构
base_domain
帮助主题
系统状态
正在运行的服务器的健康状况
主页 注销 首选项 记录 帮助
欢迎, weblogic 连接到: base_domain
主页 >部署概要
安装应用程序辅助程序
上一步 下一步 完成 取消
找到要安装的部署并准备部署
为要安装的应用程序根目录, 档案文件, 展开的档案目录或应用程序模块描述符，选择文件路径。您还可以在“路径”字段中输入应用程序目录或文件的路径。
注: 以下只显示有效文件路径。如果您找不到部署文件, 则请上载文件和/或确认您的应用程序包含所需的部署描述符。
路径:
/root/Oracle/Middleware/user_projects/domains/base_domain
最近使用的路径:
/root/Oracle/Middleware/user_projects/domains/base_domain/servers/AdminServer/upload
当前位置:
10.0.2.15 / root / Oracle / Middleware / user_projects / domains / base_domain / servers / AdminServer / upload
Jspspy.war
Test.war
jspspyTest.war
test.war


安装应用程序辅助程序 - base_domain - WLS 控制台 - Mozilla Firefox
ORACLE WebLogic Server® Administration Console
更改中心
查看更改和重新启动
域结构
base_domain
帮助主题
系统状态
正在运行的服务器的健康状况
欢迎, weblogic 连接到: base_domain
主页 >部署概要
安装应用程序辅助程序
上一步 下一步 完成 取消
将部署上载到管理服务器
单击下面的“浏览”按钮, 从您当前浏览的计算机上选择应用程序或模块。找到文件后, 单击“下一步”按钮将此部署上载到管理服务器。
部署档案:
浏览...
jspspyTest.war
部署计划档案:
浏览...
未选择文件。

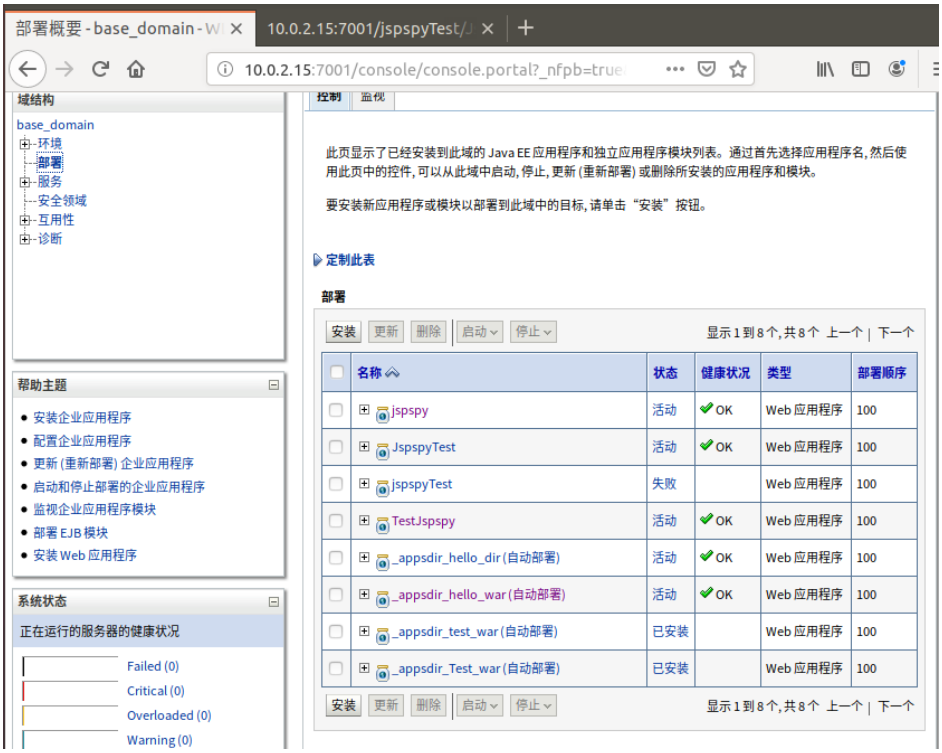
c) “安装”➡“下一步”：



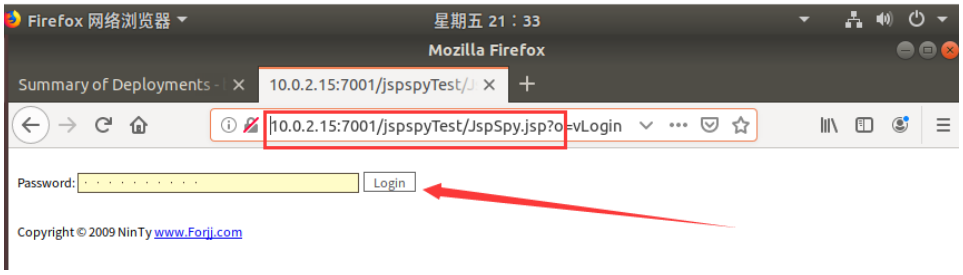
d) 填写应用名称：jspspyTest ➡ 下一步：



e） 完成：



3） 成功 getshell：



四、实验总结（此部分必须有）

1.遇到的问题及解决方法

1） 在终端执行"docker version"命令，出现报错：

```
liangll@liangll-VirtualBox:~$ docker version
Client:
 Version:           18.09.6
 API version:       1.39
 Go version:        go1.10.8
 Git commit:        481bc77
 Built:             Sat May  4 02:35:57 2019
 OS/Arch:           linux/amd64
 Experimental:      false
Got permission denied while trying to connect to the Docker daemon socket at un
ix:///var/run/docker.sock: Get http://%2Fvar%2Frun%2Fdocker.sock/v1.39/version:
 dial unix /var/run/docker.sock: connect: permission denied
```


解决方法：详见个人博客记录：

https://blog.csdn.net/liangllhahaha/article/details/92077065

2）在终端执行命令：sudo apt install python-pip 命令，出现报错：

```
liangll@liangll-VirtualBox:~$ sudo apt install python-pip
E: 无法获得锁 /var/lib/dpkg/lock-frontend - open (11: 资源暂时不可用)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), is a
nother process using it?
```

解决方法：执行命令:sudo rm /var/lib/dpkg/lock-frontend

```
nother process using it?
liangll@liangll-VirtualBox:~$ rm /var/lib/dpkg/lock-frontend
rm: 是否删除有写保护的普通空文件 '/var/lib/dpkg/lock-frontend'?
liangll@liangll-VirtualBox:~$ sudo apt install python-pip
E: 无法获得锁 /var/lib/dpkg/lock-frontend - open (11: 资源暂时不可用)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), is a
nother process using it?
liangll@liangll-VirtualBox:~$ sudo rm /var/lib/dpkg/lock-frontend
liangll@liangll-VirtualBox:~$ sudo apt install python-pip
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
将会同时安装下列软件：
  build-essential cpp cpp-7 dpkg-dev fakeroot g++ g++-7 gcc gcc-7 gcc-7-base
  gcc-8-base libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libasan4 libatomic1 libc-dev-bin libc6-dev libcc1-0
  libcilkrts5 libexpat1-dev libfakeroot libgcc-7-dev libgcc1 libgomp1 libitm1
  liblsan0 libmpx2 libpython-all-dev libpython-dev libpython-stdlib
  libpython2.7 libpython2.7-dev libpython2.7-minimal libpython2.7-stdlib
  libquadmath0 libssl1.1 libstdc++-7-dev libstdc++6 libtsan0 libubsan0
  linux-libc-dev make manpages-dev python python-all python-all-dev
  python-asn1crypto python-cffi-backend python-crypto python-cryptography
  python-dbus python-dev python-enum34 python-gi python-idna python-ipaddress
  python-keyring python-keyrings.alt python-minimal python-pip-whl
  python-pkg-resources python-secretstorage python-setuptools python-six
```

3）在终端执行命令“docker-compose up -d”时，出现报错：

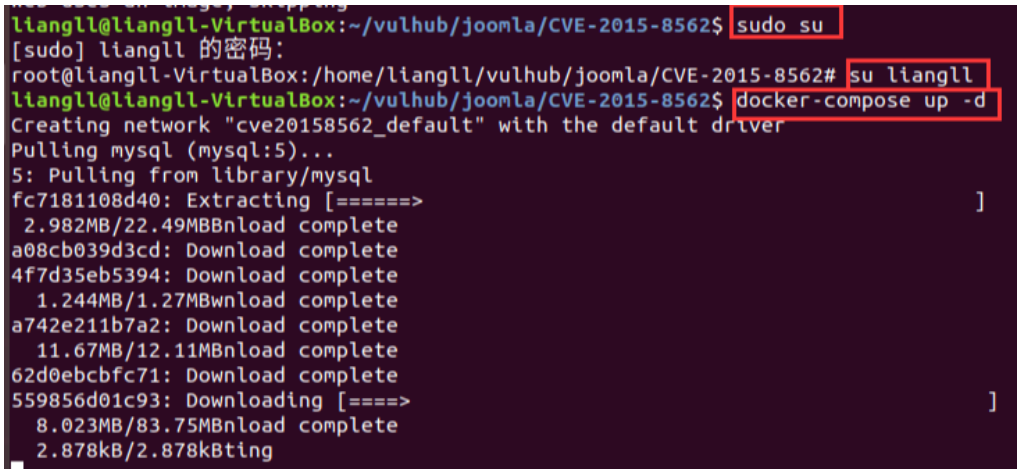
```
liangll@liangll-VirtualBox:~/vulhub/joomla/CVE-2015-8562$ docker-compose up -d
ERROR: Couldn't connect to Docker daemon at http+docker://localunixsocket - is
it running?

If it's at a non-standard location, specify the URL with the DOCKER_HOST enviro
nment variable.
```

解决方法：执行命令：

sudo su

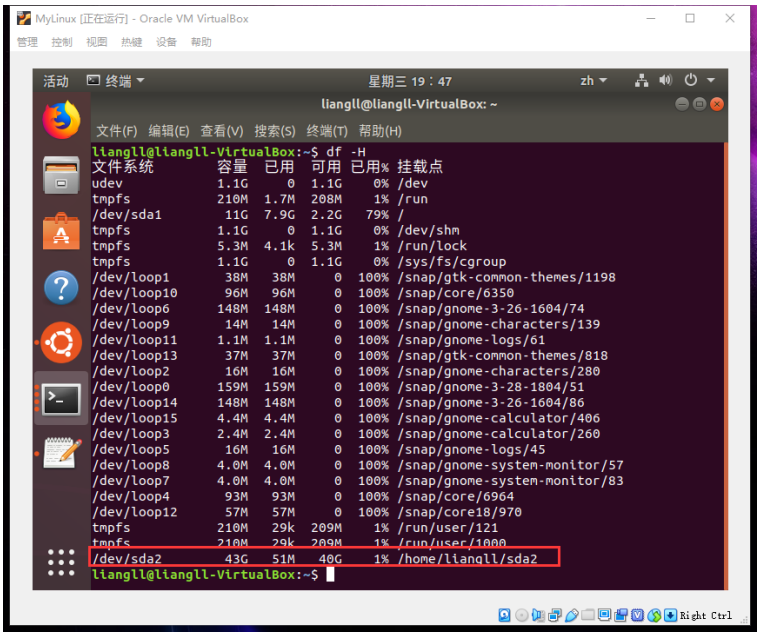
su liangll



4) 在装 ubuntu 系统时，给虚拟电脑分配的硬盘内存过小（只有 10G），导致后续虚拟电脑出现电脑内存不够的情况；

解决方法：参照网上博客链接解决：

https://blog.csdn.net/ouyang_peng/article/details/53261599



2.收获与总结

1) **实验题目的选择：**刚开始拿到老师的实验题目时，因为我想挑战难题，又对 Web 常见漏洞方面感兴趣，因此选择了四星难度的 vulhub 靶场渗透实验。打开老师在实验题目下附带的五个链接时，我完全不知道这个实验到底是要干什么，vulhub 到底是什么？实验的最终结果应该是怎么样的？这些对于我来说都是完全未知的东西，经过一天的苦苦研究与查阅资料，终于对实验有了头绪，权衡之下，我选择了 weblogic 的靶场实验。

2) **实验环境的选择：**在做本次实验之前，我从未接触过 Linux，Linux 系统对我来说是完全陌生的东西。虽然之前在一些博客上看到常见的 Linux 命令，诸如$sudo 等，但它对我来说还是一个黑洞。在查阅资料的过程中，我也发现了一些在 Windows 操作系统中模拟的靶场实验博客，但是想到在以后的学习过程中还是不可避免需要用到 Linux 环境（如部署服务器的系统选择），因此最终我还是选择用 Linux 完成本次实验。初上手 Linux，安装系统中遇到了很多坑，几乎是每前进一步就遇到一个坑，印象最深刻的就是在 VB 上安装好 Ubuntu 后，输入密码进入系统，系统提示“抱歉，认证失败。请重试”，我一直无法成功进入系统。但是和我用同样环境的同学输入密码却能立刻进入系统，这让我很崩溃，于是沉住气，去询问度娘，经过几番折腾终于进入了系统。此外还有不少坑，诸如“设置虚拟

电脑硬盘内存过小导致电脑空间不够”、在终端输入命令报各种错、进行weblogic靶场实验时找到的jspspy大马文件不好使等各种坑，但经过查阅资料，自己尝试解决后，看到自己的实验能成功完成，拿到了webshell，还是十分开心。

3) **对Web漏洞的思考：**之所以选择vulhub靶场渗透实验，很大原因是这个是关于 Web 漏洞安全的实验。上大学两年来，虽然自己也做了一些网页（网站），但是一直都没有思考过我做的东西安不安全。老师在课上曾和我们介绍在网页登录时可能会遇到的安全攻击，这给我提了一个醒，做网页不止是要思考网页的前端设计多么协调完美，后端功能多么齐全，还应思考这个网页是否足够安全，能抵挡住黑客的攻击。以后我会多关注一些 web 漏洞安全方面的知识，多丰富一下自己的知识面。