WEB应用安全和数据库安全的领航者！

应用安全高级测试

吴卓群 rainman

杭州安恒信息技术有限公司
www.dbappsecurity.com.cn

- 应用安全测试介绍
- 测试工具介绍
- 常规应用漏洞测试
- 业务逻辑测试
- 应用防火墙绕过技术

应用安全测试介绍

- 漏洞测试方式
  - 安全性测试的方式太多，每种方式都有自己的优点和缺点。没有一种方式是绝对正确的，也没有一种方式能够揭示一个给定目标的所有可能漏洞。在较高层次上，有三种方式用来发现漏洞：白盒测试、黑盒测试和灰盒测试。

应用安全测试介绍

- 白盒测试
  - 白盒测试也称结构测试或逻辑驱动测试，它是按照程序内部的结构测试程序，通过测试来检测产品内部动作是否按照设计规格说明书的规定正常进行，检验程序中的每条通路是否都能按预定要求正确工作。 这一方法是把测试对象看作一个打开的盒子，测试人员依据程序内部逻辑结构相关信息，设计或选择测试用例，对程序所有逻辑路径进行测试，通过在不同点检查程序的状态，确定实际的状态是否与预期的状态一致。

应用安全测试介绍

- 黑盒测试
  - 黑盒测试也称功能测试，它是通过测试来检测每个功能是否都能正常使用。在测试中，把程序看作一个不能打开的黑盒子，在完全不考虑程序内部结构和内部特性的情况下，在程序接口进行测试，它只检查程序功能是否按照需求规格说明书的规定正常使用，程序是否能适当地接收输入数据而产生正确的输出信息。黑盒测试着眼于程序外部结构，不考虑内部逻辑结构，主要针对软件界面和软件功能进行测试。

应用安全测试介绍

- 灰盒测试
  - 灰盒测试，是介于白盒测试与黑盒测试之间的，可以这样理解，灰盒测试关注输出对于输入的正确性，同时也关注内部表现，但这种关注不象白盒那样详细、完整，只是通过一些表征性的现象、事件、标志来判断内部的运行状态，有时候输出是正确的，但内部其实已经错误了，这种情况非常多，如果每次都通过白盒测试来操作，效率会很低，因此需要采取这样的一种灰盒的方法。

应用安全测试介绍

- 我们使用的测试方法
  - 黑盒测试

    fuzzing模糊测试

    权限攻击

    流程攻击
  - 灰盒测试

    对关键部分代码进行审计

    协议的加密解密

    关键数据的解密
  - 白盒测试

    自动化工具审计

    代码走读，发现深层次漏洞

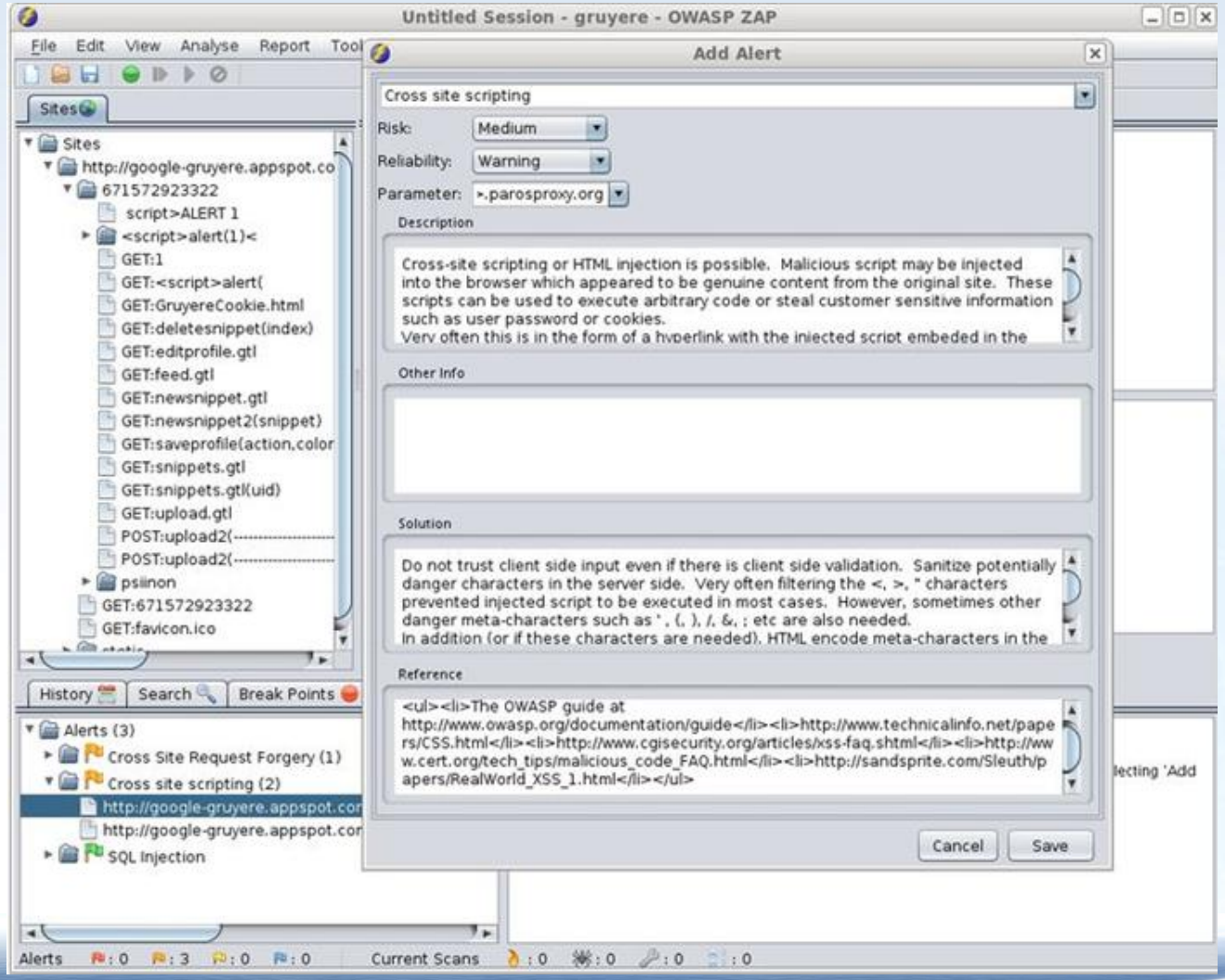
DBAPPSecurity

应用安全测试介绍

- 更关注黑盒测试
  - 看清数据请求的本质
  - 识别所有输入及输出
  - 使用开发者的思路考虑问题
  - 不断假设并否定

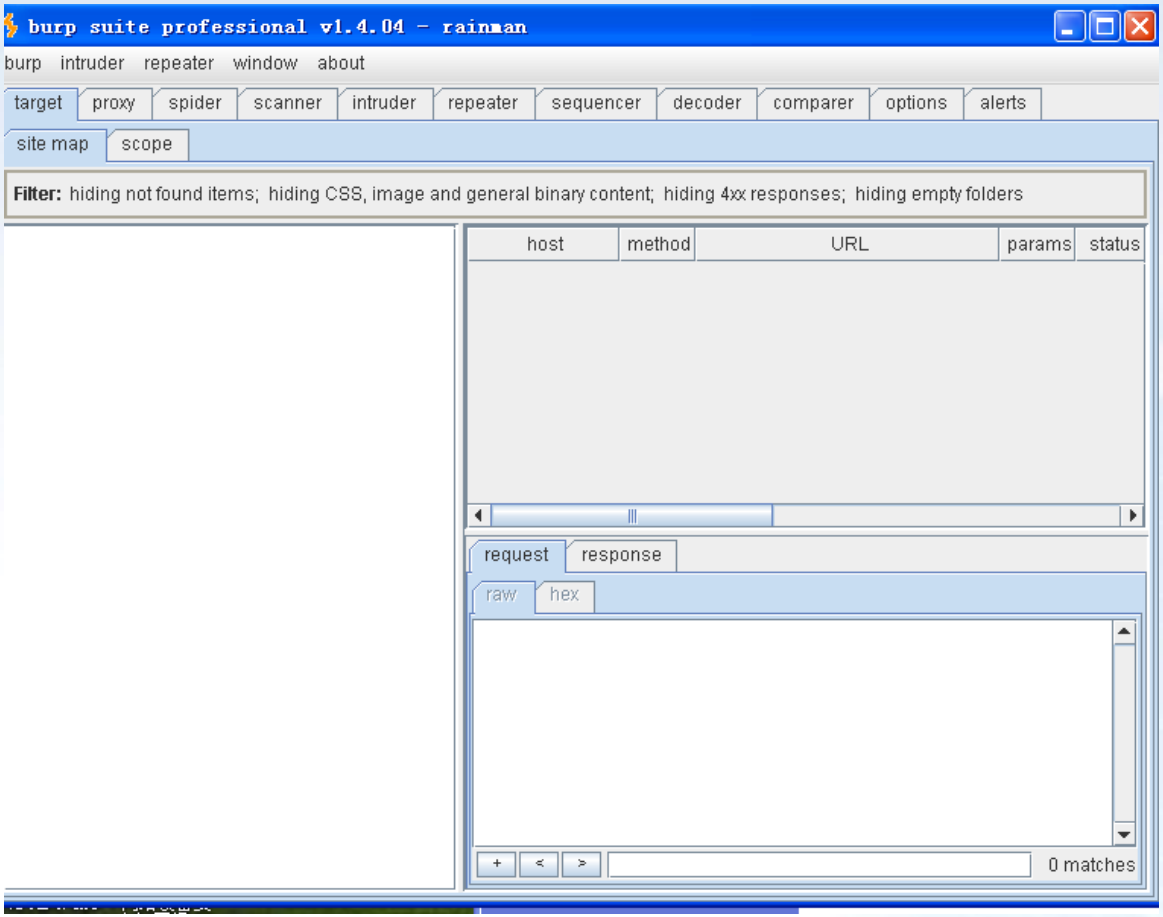
- 应用安全简介绍
- 测试工具介绍
- 常规应用漏洞测试
- 业务逻辑测试
- 应用防火墙绕过技术

- 开源的测试利器**Zed Attack Proxy**



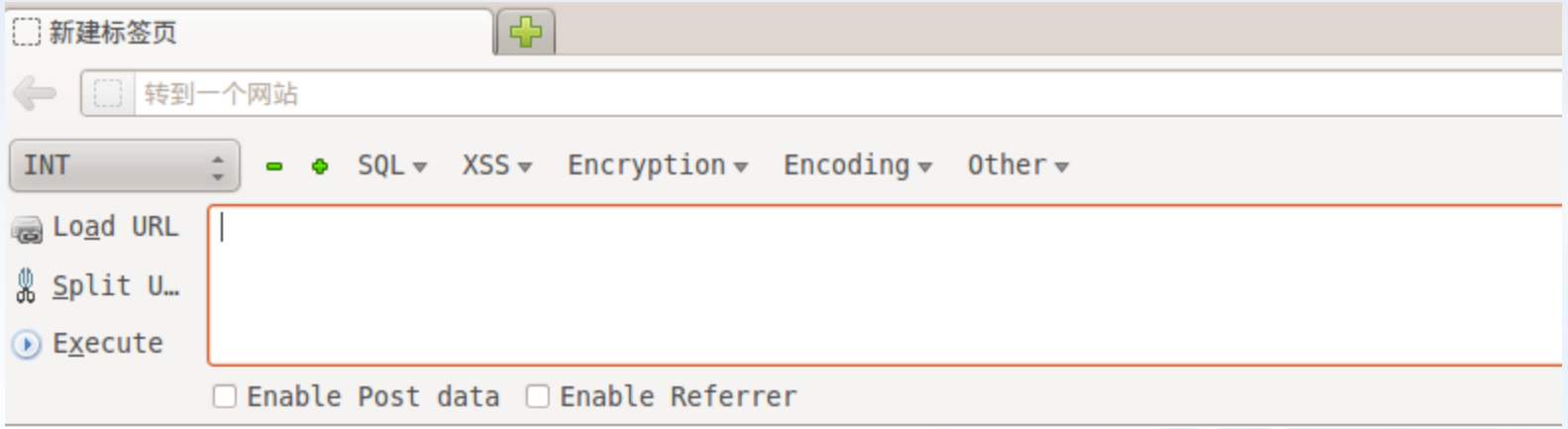
测试工具介绍

- 安全测试神器 BurpSuite



DBAPPSecurity

- 快速辅助测试工具 Hackbar



- 应用安全测试介绍
- 测试工具介绍
- 常规应用漏洞测试
- 业务逻辑测试
- 应用防火墙绕过技术

注入 - Injection

- SQL注入(包括MSSQL MySQL Oracle等)
- SQL注入漏洞，是依靠存在弱点的WEB脚本代码，来实现通过浏览器执行任意SQL语句，从而实现最终获取某种权限的攻击技术。SQL注入的关键部分在于对元数据的利用，所谓元数据即数据库的基础数据。例如我们可以通过database() version()来获得数据库的名称及版本，而我们通过SQL内置函数获得的这些内容都属于数据库元数据的内容。

注入 - Injection

- 虽然还有其他类型的注入攻击，但绝大多数情况下，问题涉及的都是SQL注入。
- 攻击者通过发送SQL操作语句，达到获取信息、篡改数据库、控制服务器等目的。是目前非常流行的Web攻击手段。
- 流行性：常见；危害性：严重
- 示例

•网页中使用如下SQL查询，其中username和password需要用户输入：

- **select * from hacked_Users where username='username' and password='password'**
- **攻击者可以输入任意用户名，并输入' or '1'='1 作为密码，实现如下查询：**
- **select * from hacked_Users where username= 'Andy ' and password=' or '1'='1'**

DBAPPSecurity

注入 - Injection

- 主要防范措施：
  - ➢严格检查用户输入，注意特殊字符：“’ ” “;” “[” “--” ” || ” “xp_” 等
  - ➢转义用户输入内容
  - ➢拒绝已经经过转义的输入
  - ➢使用参数化的查询
  - ➢使用SQL存储过程
  - ➢最小化SQL权限（禁用SA帐号）
  - ➢防止错误页面信息泄露

DBAPPSecurity

跨站脚本

- XSS
- XSS也是一个比较危险的安全隐患，很多国内介绍XSS漏洞的文章大部分在如何欺骗管理员获得后台登陆帐户或者管理员的cookies文件。但这些仅仅是XSS漏洞的简单用法，如果寻找到的XSS漏洞可以任意执行任何的Javascript脚本，那安全性也是不容忽视的。通过Javascript脚本其实也可以做一些恶意的攻击，甚至可以获得一些WEB程序的源代码，当然这个要看大家对Javascript脚本的熟悉程度。例如我们这几天公布的这个可跨站执行任意Javascript脚本的漏洞，最后我也通过这个漏洞给客户演示了如何获取他们的服务器信息，并最终实现得到其一定权限的方法。
- 同时例如session欺骗，目前我也把这些规入了XSS漏洞的范围，当然仅仅研究这两个技术也是很值得大家去深入的进行漏洞挖掘的。
- 该PPT的所有内容都为黑盒子测试的范围，也即使用这些漏洞挖掘规则，大家仅仅需要一个WEB浏览器，如IE Firefox等即可，也无需读取WEB程序的源代码，只要某个规则符合了漏洞规则的要求，即可以采取相关的漏洞攻击技术进行相应的漏洞攻击办法。

跨站脚本 - Cross-Site Scripting（XSS）

- 影响面最广的Web安全漏洞。
- 攻击者通过向URL或其他提交内容插入脚本，来实现客户端脚本执行的目的。
- 可分为三种类型：反射、存储和DOM
- 流行性：极为广泛；危害性：中等
- 示例

•反射型XSS：在搜索引擎中输入含脚本的查询内容后，查询结果页面中会出现脚本内容。

•通常要结合社会工程学欺骗用户执行。

•存储型XSS：在某论坛中攻击者新建一个帖子，在提交文本框时输入了脚本内容。当其他用户打开这个帖子时，其中包含的脚本内容被执行。

•DOM型XSS：网页中包含如下脚本内容，其中含有window.location.href对象

```
•document.write("<input id=\"a\" value=\""+window.location.href+"\">");
```

•攻击者可以直接在原始地址后添加脚本内容

DBAPP Security

跨站脚本 - Cross-Site Scripting（XSS）

- 主要防范措施：
  - 严格检查用户输入
  - 尽量限制在HTML代码中插入不可信的内容
    （可被用户输入或修改的内容）
  - 对于需要插入的不可信内容必须先进行转义
    （尤其对特殊字符、语法符合必须转义或重新编码）
  - 将Cookie设置为HttpOnly，防止被脚本获取

文件包含

- 文件包含类型，如PHP的的远程、本地文件包含漏洞

- 文件包含漏洞是PHP程序特有的一个弱点攻击，原理就是在使用include时没有安全的编程，而能够找到文件包含漏洞则是入侵一个WEB系统的很重要的因素，有了文件包含漏洞则可以很快速的达到上传WEBSHELL，然后本地提升权限的作用。

跨站请求伪造 - Cross-Site Request Forgery

- 攻击者构造恶意URL请求，然后诱骗合法用户访问此URL链接，以达到在Web应用中以此用户权限执行特定操作的目的。
- 和反射型XSS的主要区别是：反射型XSS的目的是在客户端执行脚本；CSRF的目的是在Web应用中执行操作。
- ·流行性：广泛；危害性：中等
- 示例

•某网银在执行用户转账时会提交如下URL请求：
•http://bank.com/transfer.do?acct=BOB&amount=100
•攻击者向用户发送邮件，其中包含如下链接：
•View my Pictures!

DBAPPSecurity

跨站请求伪造 - Cross-Site Request Forgery

- 主要防范措施：
  - 避免在URL中明文显示特定操作的参数内容
  - 使用同步令牌（Synchronizer Token）,检查客户端请求是否包含令牌及其有效性
  - 检查Referer Header，拒绝来自非本网站的直接URL请求

URL访问限制缺失 - Failure to Restrict URL Access

- 某些Web应用包含一些“隐藏”的URL，这些URL不显示在网页链接中，但管理员可以直接输入URL访问到这些“隐藏”页面。如果我们不对这些URL做访问限制，攻击者仍然有机会打开它们。
- 流行性：不常见；危害性：中等
- 示例

•1. 某商品网站举行内部促销活动，特定内部员工可以通过访问一个未公开的URL链接登录公司网站，购买特价商品。此URL通过某员工泄露后，导致大量外部用户登录购买。

•2. 某公司网站包含一个未公开的内部员工论坛（http://example.com/bbs），攻
•击者可以进行一些简单尝试就找到这个论坛的入口地址。

DBAPPSecurity

URL访问限制缺失 - Failure to Restrict URL Access

- 主要防范措施：
  - 对于网站内的所有内容（不论公开的还是未公开的），都要进行访问控制检查
  - 只允许用户访问特定的文件类型，比如.html, .asp, .php等，禁止对其他文件类型的访问
  - 进行渗透测试

未验证的重定向和跳转 - Unvalidated Redirects and Forwards

- 攻击者可能利用未经验证的重定向目标来实现钓鱼欺骗，诱骗用户访问恶意站点。
- 攻击者可能利用未经验证的跳转目标来绕过网站的访问控制检查。
- 流行性：不常见；危害性：中等
- 示例

•利用重定向的钓鱼链接：

•http://example.com/redirect.asp?=http://malicious.com

•更为隐蔽的重定向钓鱼链接：

•http://example.com/userupload/photo/324237/../../../redirect.asp

•%3F%3Dhttp%3A//www.malicious.com

•利用跳转绕过网站的访问控制检查：

•http://example.com/jump.asp?fwd=admin.asp

未验证的重定向和跳转 - Unvalidated Redirects and Forwards

- 主要防范措施：
  - 尽量不用重定向和跳转
  - 对重定向或跳转的参数内容进行检查，拒绝站外地址或特定站内页面
  - 不在URL中显示目标地址，以映射的代码表示（http://example.com/redirect.asp?=234)

- 应用安全测试介绍
- 测试工具介绍
- 常规应用漏洞测试
- 业务逻辑测试
- 应用防火墙绕过技术

业务逻辑测试

- 业务逻辑测试
  - 随着系统功能越来越强大，业务逻辑也更加复杂。逻辑上出现的安全问题也会越来越多。而且当出现问题时，可能比常规的安全漏洞更加严重。

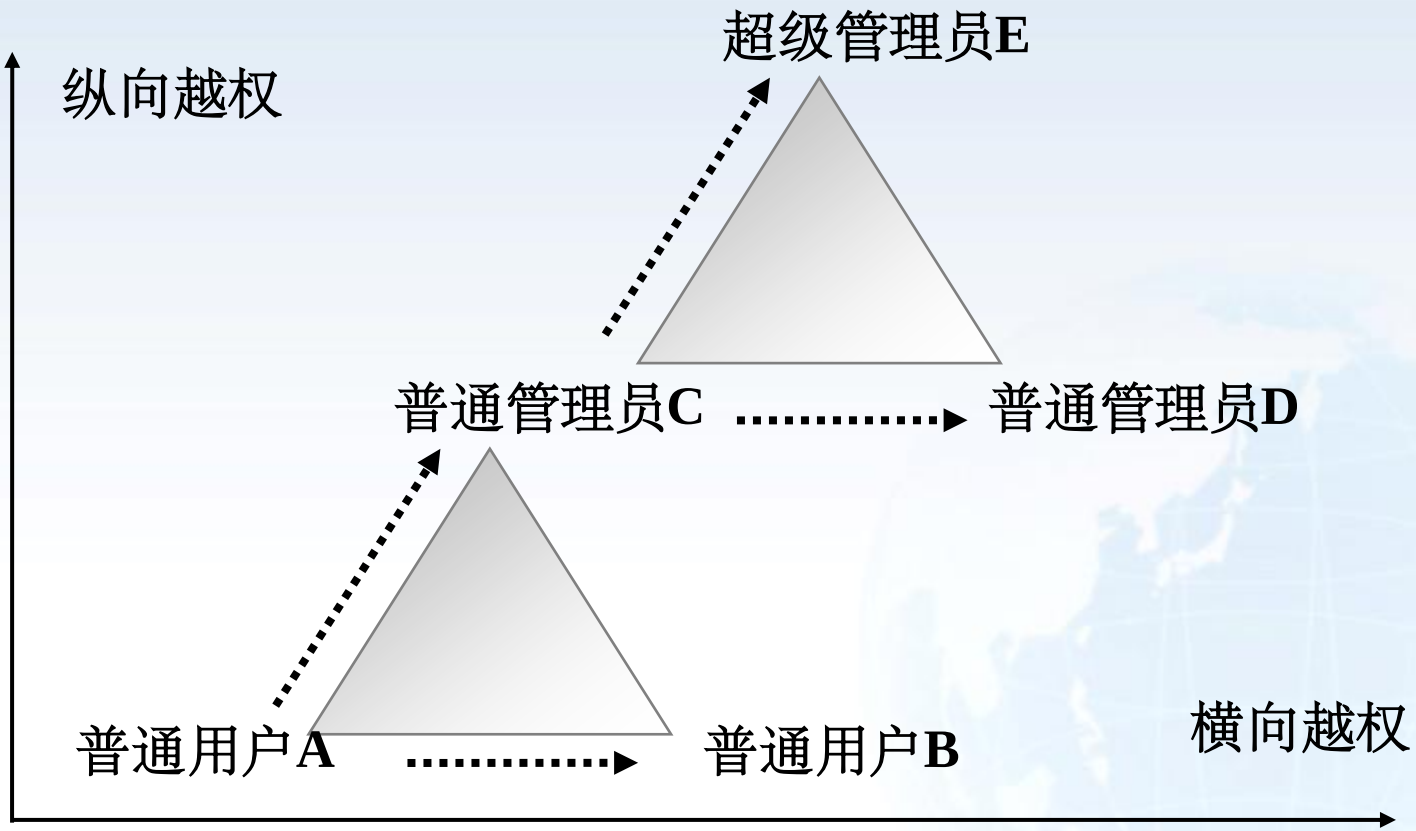
业务逻辑测试

- 应用系统漏洞
  - 权限控制漏洞

    最常见和最普遍的漏洞，重点测试。

    目前存在着两种越权操作类型：横向越权操作和纵向越权操作。前者指的是攻击者尝试访问与他拥有相同权限的用户的资源；而后者指的是一个低级别攻击者尝试访问高级别用户的资源

- 权限控制漏洞



- 权限控制漏洞

  水平权限控制：

  - 查询其他账金额
  - 查询其他账户身份信息
  - 查询其他账户的其他相关信息
  - 修改昵称
  - 修改手机信息
  - 修改业务开通功能

- 权限控制漏洞

垂直权限控制：

- **低权限账户，是否可以操作高权限账户功能**
- **是否可查看高权限账户功能**

业务逻辑测试

- 逻辑顺序测试
  - 逻辑顺序异常，可能导致安全问题

  如：

  外挂账户操作： 1. 判断账户号， 2. 判断密码，
  3. 判断姓名， 4. 完成外挂操作

业务逻辑测试

- 数据校验问题

  使用提交异常数据进行测试

  - 提交负数
  - 提交超大数字溢出

  给别人打款100元，但是提交的参数中将100改成了-100，结果？

- 账户控制
  - 数据转换输入输出账户互换

我转给A用户1000积分，但是在我提交数据时，将输入账户和输出账户进行对调。结果?

业务逻辑测试

- 短信测试
  - 很多系统存在短信功能，短信接口是否存在短信DOS攻击， 伪造短信发送
- 邮件测试
  - 存在发送邮件功能的系统，邮件接口是否存在问题，是否可以不断发送邮件，发送伪造邮件
  - 短信和邮件类似，如果没有限制发送次数就会产生拒绝服务攻击
  - 如果在参数中存在短信、邮件中的可控字段，就可伪造短信和邮件的内容

业务逻辑测试

- 其他测试
  - 是否有可暴力破解的点
    - 通过暴力可猜解出有效信息
    - 无二次验证
    - 无图形验证码
    - 其他可以直接通过工具自动猜测的
  - 验证码安全
    - 无验证码
    - 为字母，而非图片
    - 验证码使用完后没有从服务器缓存中清楚
    - 过于简单的图形验证码

- 其他能想到的漏洞
  - 逻辑漏洞不像传统漏洞，分类相对固定。任何可以利用的方法并可产生危害的方法都可以称为漏洞。所以在测试过程中，明确输入输出，只要任何输入都认为是不可信的，都需要进行检测和审计。结合业务特点，可能会发现大量的安全漏洞。

- 应用安全测试介绍
- 测试工具介绍
- 常规应用漏洞测试
- 业务逻辑策略
- 应用防火墙绕过技术

- 背景
  - WEB应用防火墙越来越普遍
  - 通过网络层面防护应用攻击
  - 目前大部分的应用防火墙都存在绕过的威胁

- 存在的问题
  - 应用防火墙规则问题
  - 应用防火墙构架导致问题
  - 应用防火墙解析方式和服务器解析方式的差异

- 针对规则的测试
  - 使用模糊化测试手段
    - BurpSuite
  - 根据waf的返回结果判断waf的规则
    - 使用替换关键字:

      1. 字母大小写
      2. /*xx*/替换空格
      3. 使用代替语句代替拦截数据

- 应用防火墙构架导致问题
  - IPS模式
    - 分片问题
    - 巨大的Content-Length，导致设备bypass
    - 解码较弱
  - 代理模式
    - 使用高并发流量，导致设备切换bypass

- 解析方法差异
  - 正则的%00、%0a、 ascii 00截断
  - 重复变量的绕过, 重复变量的变体
  - 绕过针对域名保护
  - 超大数据包绕过防护
  - 变换变量位置绕过
  - 利用不同POST的解析方式
  - 畸形数据包结构

- 正则的%00、%0a、 ascii 00截断

POST /member.php HTTP/1.1
Accept: */*
Accept-Language: zh-cn
User-Agent: Mozilla/4.0
Content-Type: application/x-www-form-urlencode
Accept-Encoding: gzip, deflate
Host: bbs.chinaunix.net
Content-Length: 118
Pragma: no-cache

aaa=x%00&username=fsdf&password=sdf&loginsubmit=true&retur
n_type=

- 重复变量的绕过，重复变量的变体

POST /member.php HTTP/1.1
Accept: */*
Accept-Language: zh-cn
User-Agent: Mozilla/4.0
Content-Type: application/x-www-form-urlencode
Accept-Encoding: gzip, deflate
Host: bbs.chinaunix.net
Content-Length: 118
Pragma: no-cache

username=fsdf' and ''='&username=fsdf&password=sdf&loginsubmit=true&return_type=

- 绕过针对域名保护

POST /member.php HTTP/1.1
Accept: */*
Accept-Language: zh-cn
User-Agent: Mozilla/4.0
Content-Type: application/x-www-form-urlencode
Accept-Encoding: gzip, deflate
Host: xxx
Content-Length: 118
Pragma: no-cache

username=fsdf&password=sdf&loginsubmit=true&return_type=

- 超大数据包绕过防护

POST /member.php HTTP/1.1
Accept: */*
Accept-Language: zh-cn
User-Agent: Mozilla/4.0
Content-Type: application/x-www-form-urlencode
Accept-Encoding: gzip, deflate
Host: xxx
Content-Length: 118
Pragma: no-cache

a=x…{10000}&username=fsdf&password=sdf&loginsubmit=true&re
turn_type=

- 变换变量位置绕过

POST /member.asp HTTP/1.1
Accept: */*
Accept-Language: zh-cn
User-Agent: Mozilla/4.0
Content-Type: application/x-www-form-urlencode
Accept-Encoding: gzip, deflate
Host: xxx
Content-Length: 118
Pragma: no-cache
Cookie: username=fsdf

password=sdf&loginsubmit=true&return_type=

- 利用不同POST的解析方式

POST / HTTP/1.1
Accept: */*
Accept-Language: zh-cn
User-Agent: Mozilla/4.0
Content-Type: multipart/form-data; boundary=--------------------------7dc33b8148092e
Accept-Encoding: gzip, deflate
Content-Length: 253
Host: xxxxx

----------------------------7dc33b8148092e
Content-Disposition: form-data; name="username"

xxxx
----------------------------7dc33b8148092e
Content-Disposition: form-data; name="password"

ddddd
----------------------------7dc33b8148092e--

- 畸形数据包结构

POST /member.asp HTTP/1.1
Accept: */*
Accept-Language: zh-cn
User-Agent: Mozilla/4.0
Content-Type: application/x-www-form-urlencode
Accept-Encoding: gzip, deflate
Host: xxx
Content-Length: 118
Pragma: no-cache

Username%20=fsdf&username=fsdf
&password=sdf&loginsubmit=true&return_type=

WEB应用安全和数据库安全的领航者！

•Thank You !

杭州安恒信息技术有限公司
www.dbappsecurity.com.cn