1. 安全漏洞的分类、描述、解决方案

|  |  |  |  |
| --- | --- | --- | --- |
| **漏洞列表** | **漏洞描述** | **风险等级** | **解决方案** |
| SQL注入漏洞 | 主要是SQL注入漏洞，对用户提交CGI参数数据未做充分检查过滤，用户提交的数据可能会被用来构造访问后台数据库的SQL指令，从而非授权操作后台的数据库，导致敏感信息泄漏、破坏数据库内容和结构、甚至利用数据库本身的扩展功能控制服务器操作系统。通常在SQL查询语句、LADP查询语句、XPATH查询语句、OS命令，XML解析器、SMTP头、程序参数等中找到。 | 高 | 使用安全的API,避免使用解释器；对输入的特殊字符进行Escape转义处理；使用白名单来规范化的输入验证方法。sql语句全部使用参数形式调用，不拼sql语句，对输入都要验证：客户端验证+服务器端验证 |
| 跨站脚本漏洞XSS | 跨站脚本漏洞，即XSS，通常用Javascript语言描述，利用的是客户端的弱点，常见3种漏洞，1）存储式；2）反射式；3）基于DOM。由于动态网页的web应用对用户提交请求参数未做充分的检查过滤，允许用户在提交的数据中加入HTML、JS代码，未加编码地输出到第三方用户的浏览器，恶意攻击者可以利用Javascript、VBScript、ActiveX、HTML语言甚至Flash应用的漏洞，发送恶意代码给另一个用户，因为浏览器无法识别脚本是否可信，从而跨站漏洞脚本便运行并让攻击者获取其他用户信息。攻击者能盗取会话cookie或session、获取账户、模拟其他用户身份，甚至可以修改网页呈现给其他用户的内容。 | 高 | 对应用系统源代码进行优化，对用户可控参数，进行严格的后台检测和过滤，对特殊字符进行转义，不能简单的进行JS过滤。  对所有web应用输入参数进行过滤，建议过滤出所有以下字符：  [1] |（竖线符号）  [2] &（&符号）  [3];（分号）  [4] $（美元符号）  [5] %（百分比符号）  [6] @（at 符号）  [7] '（单引号）  [8] "（引号）  [9] \'（反斜杠转义单引号）  [10] \"（反斜杠转义引号）  [11] <>（尖括号）  [12] ()（括号）  [13] +（加号）  [14] CR（回车符，ASCII 0x0d）  [15] LF（换行，ASCII 0x0a）  [16] ,（逗号）  [17] \（反斜杠）  或者对特殊字符进行html编码，例如php中的htmlspecialchars()这样的函数。 |
| 会话预测漏洞 | 攻击者推断或猜测出会话凭证，劫持用户会话或仿冒用户，从而获得非授权访问。如在给用户传送的cookies时，用户有可能利用该信息猜测到会话ID。如果成功利用该安全漏洞，恶意用户就可以获取其它合法用户的基于cookie的认证，从而利用该用户的权限进行破坏活动。 | 高 | 1.不进行默认身份验证，2.使用强大的认证，3.不要在URL中暴露会话ID。 |
| 会话管理漏洞 | Web应用程序允许重复使用会话凭证或会话ID，导致攻击者冒充其他用户身份。如1、用户更改密码之前不验证用户，而是依靠会话的IP地址；2、没有会话超时限制，如用户使用公共计算机访问网站。离开时，该用户没有点击退出，而是直接关闭浏览器；3、用户忘记密码后，密码找回功能太过简单。4、机票预订应用程序支持URL重写，把会话ID放在URL里。4、内部或外部攻击者进入系统的密码数据库. 存储在数据库中的用户密码没有被加密 | 高 | 1、区分公共区域和受限区域，站点的公共区域允许任何用户进行匿名访问。受限区域只能接受特定用户的访问，而且用户必须通过站点的身份验证。  2、对最终用户帐户使用帐户锁定策略，3、支持密码有效期，4、能够禁用帐户，5、不要在用户存储中存储密码，可以存储一个单向哈希值，然后使用用户所提供的密码重新计算哈希值。为减少对用户存储的词典攻击威胁，可以使用强密码，并将随机 salt 值与该密码结合使用。  6、要求使用强密码，7、不要在网络上以纯文本形式发送密码  8、保护身份验证 Cookie9、使用 SSL 保护会话身份验证 Cookie  10、对身份验证 cookie 的内容进行加密  11、限制会话寿命，12、避免未经授权访问会话状态 |
| 不安全的直接对象引用 | 指一个已经授权的用户，通过更改访问时的一个参数，从而访问到了原本其并没有得到授权的对象。Web应用往往在生成Web页面时会用它的真实名字，且并不会对所有的目标对象访问时来检查用户权限，所以这就造成了不安全的对象直接引用的漏洞。常见的错误暴露数据的位置是URL和超链接、隐藏表单域、ASP.NET的未保护视图状态、直接列表框、JavaScript代码和客户端对象(如Java Applet)。 | 高 | 防止这种漏洞的最佳方法是使用随机、不可猜测的ID、文件名和对象名，而且不要暴露对象的真实名称。方法有两种：1使用非直接的对象引用——这防止了攻击者直接访问其并未授权的对象，通过一种mapping或是其他的方法让攻击者无法直接访问。2. 检查访问——对每一个来自于不信任的源的直接对象引用都必须包含访问控制检查，从而确信该用户对该对象拥有访问权。每次访问敏感文件或内容时，都要验证访问数据的用户已获得授权 |
| 跨站请求伪造 | 跨站请求伪造，即CSRF，CSRF利用的是网站服务器的弱点 | 高 | CSRF漏洞防御主要可以从三个层面进行，即服务端的防御、用户端的防御和安全设备的防御。1.给每个HTTP请求添加一个不可预测的令牌，并保证该令牌对每个用户会话来说是唯一性。最好的办法是将独有的令牌包含在隐藏字段中，通过HTTP请求发送，避免在URL中暴露出来。2.设定短暂的可信用户会话时间。 3.每次提出一个可信行为时，对发出请求的用户进行验证。 4.在URL和表单中增加的每个请求提供基本会话令牌以外的每个请求用户验证。  5.从应用程序中删除所有XSS脆弱性。攻击者通过XSS可获得有用的攻击信息。例如，攻击者可以伪造一个提示用户输入身份信息的表单 |
| URL跳转漏洞 | 攻击者挟持用户在当前已登录的web应用程序执行非本意的操作，url跳转属于跨站请求伪造的一种，我们需要对传入的URL做有效性的认证，保证该URL来自于正确的地方，对web程序不恰当的状态管理可能导致攻击者绕过页面正常访问顺序的攻击 代码编程类 | 高 | 限制的方式同防止csrf一样可以包括：  1 referer的限制  如果确定传递URL参数进入的来源，我们可以通过该方式实现安全限制，保证该URL的有效性，避免恶意用户自己生成跳转链接  2 加入有效性验证Token  我们保证所有生成的链接都是来自于我们可信域的，通过在生成的链接里加入用户不可控的Token对生成的链接进行校验，可以避免用户生成自己的恶意链接从而被利用，但是如果功能本身要求比较开放，可能导致有一定的限制。 |
| 任意文件包含漏洞 | 严格来说，文件包含漏洞是“代码注入”的一种，其原理就是注入一段用户能控制的脚本或代码，并让服务端执行。“代码注入”的典型代表就是文件包含，文件包含漏洞可能出现在JSP、PHP、ASP等语言中，原理都是一样的，分为本地文件包含漏洞（LFI）和远程文件包含漏洞（RFI），能够打开并包含本地文件的漏洞，被称为本地文件包含漏洞。利用本地文件包含漏洞，可以查看系统任意文件内容，如果具备一些条件，也可以执行命令文件。包含漏洞的产生原因是在通过引入文件时，由于传入的文件名没有经过合理的校验，或者校检被绕过，从而操作了预想之外的文件，就可能导致意外的文件泄露甚至恶意的代码注入。当被包含的文件在服务器本地时，就形成了本地文件包含漏洞 代码编程类 | 高 | 一般的做法是将用户文件访问限制在指定的区域或者是通过白名单的方式包含文件。具体是从代码层和Web服务器安全配置两个方面来防范。首先来从代码层来讲，在开发过程中应该尽量避免动态的变量，尤其是用户可以控制的变量。一种保险的做法是采用“白名单”的方式将允许包含的文件列出来，只允许包含白名单中的文件。还有一种做法是将文件包含漏洞利用过程中的一些特殊字符定义在黑名单中，对传入的参数进行过滤，但这样有时会因为过滤不全，导致被有经验的攻击者绕过。在Web服务器安全配置方面可以通过设定php.ini中open\_basedir的值将允许包含的文件限定在某一特定目录内，这样可以有效避免利用文件包含漏洞进行的攻击 |
| 任意文件上传漏洞 | 由于文件上传功能实现代码没有严格限制用户上传的文件后缀以及文件类型，导致允许攻击者向某个可通过 Web 访问的目录上传任意PHP文件，并能够将这些文件传递给 PHP 解释器，就可以在远程服务器上执行任意PHP脚本 | 高 | 1.检查是否判断了上传文件类型及后缀2.定义上传文件类型白名单，即只允许上传的文件类型3.文件上传目录禁止脚本解析 |
| 任意文件读取漏洞（包括了任意文件下载漏洞） | 攻击者可以通过或者后台登录之后读取任意文件包括数据库配置文件.可导致入侵者获得数据库权限并利用数据库提权进一步获得系统权限。如系统未对传入的URL进行限制，并且部分服务器（192.168.7.2、192.168.7.4）上的weblogic以root权限运行，导致攻击者可以下载任意文件。 | 高 | 将文件下载的路径限制在可控的范围之内，例如：专用于文件下载的目录。 |
| 未验证的重定向和转发 | Web应用程序在重定向至其他网页和网站时，接收不受信任的数据来确定目标网页，攻击者可将用户重定向至钓鱼或恶意网站，或访问未经授权的网页 | 高 | 尽可能的避免使用重定向和转发机制；  如果使用了，那么在定义目标url的时候不要包含用户参数；  如果一定要包含用户的参数，那么，对每个参数都必须进行验证以确保它的正确性和合法性；或是在服务器端提供映射机制，将用户的选择参数转变为真正的目标页面； |
| 拒绝服务漏洞 | 攻击者想办法让目标机器停止提供服务或资源访问，是黑客常用的攻击手段之一。这些资源包括磁盘空间、内存、进程甚至网络带宽，从而阻止正常用户的访问。。典型的拒绝服务攻击有如下两种形式：资源耗尽和资源过载。拒绝服务攻击大多是由于错误配置或者软件弱点导致的。 | 高 | 1.配置方面：对网络防火墙等方面找专业人员进行配置。2.软件引起的漏洞只能依靠打补丁。 |
| 组件漏洞 | 指的是开发人员使用的组件含有漏洞，这些漏洞能被自动化工具发现和利用。然后攻击者根据需要定制攻击代码并实施攻击。 | 高 | 尽量使用最新版本的组件，如果没有使用最新版本组件，建议项目有以下流程1.标识正在使用的所有组件及其版本。  2.及时关注这些组件的安全信息并保证他们是最新的。  3.建立使用组件的安全策略，禁止使用未经安全评估的组件。  4.在适当情况下，对组件进行安全封装，精简不必要的功能，封装容易受攻击部分。 |
| XML注入漏洞 | 如同HTML脚本注入一样，在输出中包含攻击者提供的数据的地方，XML是容易受到攻击的。三种最常见的XML注入攻击是：XML数据注入、可扩展样式表语言转换（Extensible Stylesheet Language Transformation，XSLT）注入和XPath/XQuery 注入。 | 高 | 检查所使用的底层xml解析库，默认禁止外部实体的解析，同时增强对系统的监控，防止此问题被人利用 |
| 暴力破解 | 因认证强度低于业务安全要求或登录页面未使用验证码校验机制，攻击者可以通过穷举方式自动猜测用户登录身份凭证，会话凭证以及未公开目录及文件名， | 中 | 增加校验码验证机制，不要存在用户名与密码一样的用户。 |
| 自动表单提交 | 攻击者使用表单提交工具，向登录、注册服务等表单自动提交信息的恶意行为。 | 中 | 在登录或注册页面增加例如验证码、随机问题等来防止恶意行为。 |
| 权限绕过 | 攻击者绕过权限认证对敏感内容或功能进行非授权访问 | 中 | 所有页面都要增加认证控制。 |
| 不安全密码恢复机制 | 网站采用了不安全的密码恢复机制，攻击者可以通过暴力破解、安全问题猜测等手段绕过密码恢复机制，从而非法获得、更改或回复他人的密码。 | 中 | 找回密码等密码恢复地方时，不要暴露用户名或用户密码等用户信息。 |
| 非授权访问漏洞 | 简单说就是未经授权可以直接访问后台页面，具体说是因用户账号设计不合理或服务端未做权限校验，web应用未能有效执行授权检查，攻击者可以违反安全策略，执行权限之外的功能或者访问权限之外的数据或功能。 | 中 | 1.添加用户会话检查及权限控制代码  2.检查管理权限的过程并确保能够容易进行升级和审计，切记硬编码。  3.默认缺省情况下，应该拒绝所有访问的执行权限。对于每个功能的访问，需要明确角色授权。  4.检查每个功能分配的权限合理有效。 |
| 不恰当的错误处理 | 基于后台数据库的web应用在出现错误时会输出过于丰富的信息，比如错误类型，出现错误脚本的绝对路径，网页主目录的绝对路径，出现错误的sql语句及参数，软件的版本，系统的配置信息等。信息泄漏本身并不太重要，但结合其他漏洞可极大地提高攻击入侵的效率。 | 中 |  |
| URL猜测 | 攻击者通过猜测不在正常网站目录树的URL链接，进一步实现越权访问 | 中 |  |
| 敏感信息泄漏 | 攻击者在实施攻击前收集应用程序信息，包括web服务器软件类型及版本，web应用程序编程语言、数据库版本、文件路径等信息 | 中 | 1.避免将服务器的信息泄露到程序的任何界面中，包括异常信息。  2.预测一些威胁，加密数据的存储和传输。  3.尽快清除没有必要存放的重要的敏感数据。  4.确保使用合适强大的标准算法和密钥，并且密钥管理到位。  5.确保使用密码专用算法存储密码  6.禁用自动收集敏感数据，禁用包含敏感数据的页面缓存。 |
| 目录遍历攻击漏洞 | 服务端将Web服务器路径直接输出到页面的隐藏表单中或其他地方，攻击者操纵输入参数使应用系统执行或透露任意文件内容，或对服务器任意文件目录进行读写删除等操作 | 中 | 避免将服务器的路径信息泄露到客户端页面中 |
| 安全审计功能缺失漏洞 | 缺乏有效的安全审计功能，无法对应应用程序重要安全事件进行审计 | 中 | 增加审计功能 |
| 后门 | 后门是攻击者在参与应用程序开发的过程中，特意留下可绕过系统安全控制的功能或超级密码。在系统上线后利用后门非授权访问系统的数据或功能 | 中 | 修改完善系统，去掉后门程序 |
| 系统弱口令漏洞 | 存在弱口令如aaaaaa,123456等。 | 中 | 修改系统中的帐号口令，要求满足复杂性要求（包含大小写字母、数字、特殊符号中的任意三种，至少6位以上）。 |
| OS命令注入漏洞 | OS命令注入和SQL注入差不多，只不过SQL注入是针对数据库的，而OS命令注入是针对操作系统的。OS命令注入即能够在服务器上执行任意命令 | 中 | 1.不要调用外部程序2.过滤调、;，[，]，　，<，>，\之类的符号。  　　3.设置用户的权限。 |
| 服务器端请求伪造漏洞SSRF | SSRF(Server-Side Request Forgery:服务器端请求伪造) 是一种由攻击者构造形成由服务端发起请求的一个安全漏洞。一般情况下，SSRF攻击的目标是从外网无法访问的内部系统。（正是因为它是由服务端发起的，所以它能够请求到与它相连而与外网隔离的内部系统）。  SSRF 形成的原因大都是由于服务端提供了从其他服务器应用获取数据的功能且没有对目标地址做过滤与限制。比如从指定URL地址获取网页文本内容，加载指定地址的图片，下载等等 | 中 | 1,过滤返回信息，在返回给用户之前先验证返回的信息是否符合标准。  2, 自定义错误信息，防止根据错误信息来判断远端服务器的端口状态。  3,限制请求的端口为http常用的端口，比如，80,443,8080,8090。  4,黑名单内网ip。避免应用被用来获取获取内网数据，攻击内网。  5,禁用不需要的协议。仅仅允许http和https请求。可以防止类似于file:///,gopher://,ftp:// 等引起的问题。  6,如果web应用提供的功能是获取某一种文件类型，比如图片，则可以限制为jpg等 |
| 逻辑漏洞 | 一般指业务逻辑存在BUG，可能会导致账户余额可能被消费，隐私被泄露等。 | 中 |  |
| 代码执行漏洞 | 用户通过浏览器提交执行命令，由于服务器端没有针对执行函数做过滤，导致在没有指定绝对路径的情况下就执行命令，可能会允许攻击者通过改变 $PATH 或程序执行环境的其他方面来执行一个恶意构造的代码。命令注入攻击中WEB服务器没有过滤类似system(),eval()，exec()等函数是该漏洞攻击成功的最主要原因。 | 中 | 建议假定所有输入都是可疑的，尝试对所有输入提交可能执行命令的构造语句进行严格的检查或者控制外部输入，系统命令执行函数的参数不允许外部传递。  ·不仅要验证数据的类型，还要验证其格式、长度、范围和内容。  ·不要仅仅在客户端做数据的验证与过滤，关键的过滤步骤在服务端进行。  ·对输出的数据也要检查，数据库里的值有可能会在一个大网站的多处都有输出，即使在输入做了编码等操作，在各处的输出点时也要进行安全检查。  ·在发布应用程序之前测试所有已知的威胁。 |
|  |  |  |  |
|  |  |  |  |