

软件安全测评

北京邮电大学计算机学院
信息安全系

张淼



火龙果•整理
uml.org.cn



第四讲 软件安全设计

- 1 软件安全设计目标与主要任务
- 2 软件安全设计原则
- 3 软件安全设计方法
- 4 安全设计审查策略



1 软件安全设计目标与主要任务

- 如今计算机安全性的最大问题之一就是软件不够健壮。软件的安全性往往只是事后才被考虑到，很多时候其都被遗忘



1 软件安全设计目标与主要任务

- **Windows** 操作系统不是为今天的网络环境设计的；它是在 **PC** 还没有进入互联网时代时设计的，在设计之初没有很好的考虑到安全性。



1 软件安全设计目标与主要任务

- 等到发现软件的安全风险再对软件的安全性进行改进可行吗？
- 几乎每一个程序员都在项目开发时遇到过自己软件的安全性bug，然后就不停地进行修补。



1 软件安全设计目标与主要任务

- 软件安全并不是一个随时都可以添加到系统中的功能部件，而是一个需要有效、仔细地规划和设计的遍布于整个系统的特性。因此，需要采用更为主动的方法来改进软件的安全性



1 软件安全设计目标与主要任务

- 应该从软件设计伊始就要考虑软件的安全性
- 可以通过在设计和开发软件时运用一系列设计原则与方法来规避所有已知的风险。



2 软件安全设计原则

- 为了进行安全的软件设计，我们将给出**10**条用于设计安全的软件的通用性原则。这些原则的目标是要提炼出在设计和构建安全的软件系统时应该记住的要点。遵循这些原则将帮助您避免许多常见的安全性问题。



2 软件安全设计原则

——原则1：保护最薄弱的环节

- 安全是一个木桶，能盛多少水是由木桶最短的木板决定的，安全取决于最短的木板
- 如果进行适当的风险分析，则会比较容易的发现一些软件中比较薄弱的组件。应该首先消除那些比较严重的风险，而不是那些比较轻微的风险。



2 软件安全设计原则

——原则2：纵深防御

- 使用多重防御策略来管理风险，以便在一层防御被攻破时，另一层防御将会在一定程度上阻止破坏。

2 软件安全设计原则

——原则2：纵深防御





2 软件安全设计原则

——原则3：防止软件错误导致信息泄漏

- 任何复杂的系统都会有错误，这是很难避免的，但是可以避免的与错误有关的安全问题。在目前，许多软件出现错误时，都会导致不安全行为的发生。



2 软件安全设计原则

——原则3：防止软件错误导致信息泄漏

- 例如需要支持不安全的旧版软件而出现问题
- 有经验的黑客可以在数据经过网络时，通过篡改数据来迫使两台新客户端都认为对方是旧客户端。
- 软件的原始版本十分简单，完全没有使用加密。



2 软件安全设计原则

——原则3：防止软件错误导致信息泄漏

- 对这一问题的一种较好解决方案是从开始就采用强制升级方案进行设计；使客户端检测到服务器不再支持它。
- 当然这会得罪一些老用户，但是安全性得到了很大的提高。



2 软件安全设计原则

——原则4：最小特权

- 最小特权原则规定：只授予执行操作所必需的最少访问权，并且对于该访问权只准许使用所需的最少时间。



2 软件安全设计原则

——原则4：最小特权

- 例如，如果您没养宠物，只需要一位朋友偶尔收取您的邮件，那么您应当只给他邮箱钥匙。
- 即使您的朋友可能找到滥用那个特权的方法，但至少您不必担心出现其它滥用的可能性。
- 如果您不必要地给出了房门钥匙，那么所有一切都可能发生。



2 软件安全设计原则

——原则5：隔离

- 隔离原则背后的基本思想是如果我们将系统分成尽可能多的独立单元，那么我们可以将对系统可能造成损害的量降到最低



2 软件安全设计原则

——原则5：隔离

- 例如，潜水艇拥有许多不同的船舱，每个船舱都是独立密封，就应用了这个原则；
- 如果船体裂开了一个口子而导致一个船舱中充满了水，其它船舱不受影响。
- 船只的其余部分可以保持其完整性，人们就可以逃往潜水艇未进水的部分而幸免于难。



2 软件安全设计原则

——原则6：简单性

- 复杂性增加了软件的风险，这似乎在任何软件中都不可避免。很明显，软件的设计和实现应该尽可能地简单。
- 只要一个组件的质量是良好的，就应该考虑尽可能重用该组件。



2 软件安全设计原则

——原则7：保护隐私

- 如果用户认为您的软件不能很好地保护隐私信息，则您可能很快失去用户的信任。
- 维护隐私可能经常要与可用性进行平衡。



2 软件安全设计原则

——原则7：保护隐私

- 用户隐私不是您应该考虑的唯一隐私类型。恶意的黑客往往根据容易从您的系统中收集的信息发起攻击。



2 软件安全设计原则

——原则8：难以隐藏的机密

- 安全性通常是有关保守机密的。用户不想让其个人数据泄漏出去，所以您必需使用加密密钥以避免窃听或篡改。
- 面对的主要威胁：对二进制进行逆向工程、网络的安全性和公司内部泄密



12月21日

CSDN 600万用户密码被黑客公布，涉及包括网易邮箱、QQ邮箱、人人网等多家网站



12月22日早间

黑客爆出多玩网、178、7k7游戏用户资料，涉及的用户在5000万以上

CSDN

表示被盗的号码主要是2009年4月以前的用户，用户当时是采用明文密码，2010年8月底已清理掉所有明文密码，从2010年9月开始全部是安全的，9月之前的有可能不安全

网易、人人公司

表示已经对存在风险的用户进行弹窗提示，要求用户通过修改密码、绑定手机等方式进行隐私保护





2 软件安全设计原则

——原则9：不要轻易地扩展信任

- 不能相信最终用户会按照他们期望的那样使用客户机。我们还强烈要求您不要太相信您自己的服务器，以防止数据被窃取。



2 软件安全设计原则

——原则9：不要轻易地扩展信任

- 记住信任是可转移的。一旦您信任某个实体，就会暗中将它扩展给该实体可能信任的任何人。出于这个原因，可信的程序决不应该调用不可信的程序。



2 软件安全设计原则

——原则10：信任公众

- 无失败地重复使用会增进信任。公众的监督也可增进信任。
- 最好信任已被广泛使用并且被广泛监督的安全性数据库。



2 软件安全设计原则 ——原则10：信任公众

- 原则10与原则9 是不是冲突了呢？
- 究竟应该应用哪些原则呢？
- 这些原则的使用效果怎么样呢？



2 软件安全设计原则 ——总结

- 当使用这些原则时，应用好的软件风险管理技术很重要。
- 仅当这种策略告诉您这样做很有效时，您才应该应用这些原则。另外，应用这些原则有时候也会适得其反，所以这种策略更关键。
- 最后，我们当然不指望这组原则会适用于您可能遇到的每种情况，但通常它肯定会干得很出色。



3 软件安全设计方法

- 下面我们在这些原则的基础上介绍一下安全设计的具体方法



3 软件安全设计方法

■ 软件的弱点及由于不完善的设计引起的问题

漏洞类型	由于不完善的设计引起的问题
输入验证	由在查询字符串，表单字段，cookie以及HTTP标头中恶意嵌入的字符串发起的攻击。它们包括命令行的执行、跨站点脚本（XSS）、SQL注入和缓冲区溢出攻击
认证	身份欺骗，密码破解，提升特权，未经授权访问
授权	存取机密或受限数据，篡改，和执行未经授权的行动
配置管理	未经授权的访问管理员接口，更新配置数据，擅自获取用户帐户和帐户配置文件
敏感数据	机密信息泄露，数据被篡改



3 软件安全设计方法

会话管理	会话标识符的捕获导致会话被截获和身份欺骗
加密技术	存取机密数据或帐户凭据，或两者兼得
参数篡改	除上述情况之外的路径遍历攻击，命令执行和绕道访问控制机制，会导致信息泄露，提升特权及拒绝服务攻击。
异常管理	拒绝服务攻击和披露敏感的系统级的细节
审核和日志	无法记录入侵现象，无法证明用户的活动，在问题诊断上出现困难



3 软件安全设计方法

——输入验证

- 适当的输入验证是防范面向应用程序攻击的一种有效的措施。
- 适当的输入验证是一项有效的方法，可以阻止**XSS**，**SQL注入**，**缓冲溢出**和其他的输入攻击



3 软件安全设计方法

——输入验证

下列做法可以改善应用程序的输入验证：

- 假设所有的输入都是恶意的
- 集中进行输入验证
- 不要依赖于客户端验证
- 小心规范化带来的问题
- 将输入进行约束，拒绝和处理



3 软件安全设计方法

——认证

- 认证是决定访问者身份的过程，主要要考虑以下三个方面：
- 确定在应用程序中，哪些地方需要进行认证。
- 验证访问者是谁。
- 在随后的请求中识别用户。



3 软件安全设计方法

——认证

- 以Web应用程序为例：
- 用户名和密码是通过一个不安全的信道以明文形式发送的吗？
- 证书是如何保存的？
- 如何验证证书？
- 在初始登录后如何识别已认证用户的身份？



3 软件安全设计方法

——认证

下面的做法可以改善应用程序的认证：

- 将公共区域和限制区域分离
- 对终端用户帐户使用帐户锁定策略
- 支持密码有效期
- 能够禁用帐户
- 不在用户区保存密码
- 需要强有力的密码。
- 不要用明文发送密码
- 保护认证**Cookie**



3 软件安全设计方法

——授权

- 授权决定经过认证的身份可以做些什么和可以访问哪些资源。
- 不正确的或是较差的授权会导致信息泄露和数据被篡改。
- 纵深防御是适用于应用程序的授权策略的关键安全原则。



3 软件安全设计方法

——授权

下列做法可以改善应用程序的授权：

- 使用多重把关。
- 限制用户访问系统级的资源。
- 考虑授权的粒度



3 软件安全设计方法

——配置管理

- 仔细考虑一下应用程序的配置管理功能。
- 如果支持远程管理，应当如何保护管理界面呢？
- 违反了管理界面的安全的后果是相当严重的，因为攻击者经常用管理员权限中止运行并能直接接触到整个网站。



3 软件安全设计方法

——配置管理

下列做法可以改善应用程序的配置安全：

- 保护管理界面
- 保护配置存储
- 保持单独的管理特权
- 使用最少的特权进程和服务帐户



3 软件安全设计方法

——敏感数据

- 保护应用程序执行所使用的机密，诸如密码和数据库连接字符串。
- 如果应用程序要处理用户的私人信息，应采取特别措施以确保该数据仍然是私有的，并且不发生改变。



3 软件安全设计方法

——敏感数据

以下做法可以改善应用程序处理机密的安全性：

- 如果能避免就不要存储机密
- 不要用代码来存储机密
- 不要用明文来存储数据库连接，密码和密钥
- 避免在本地安全授权（LSA）存储机密
- 使用数据保护API（DPAPI）来加密机密



3 软件安全设计方法

——敏感数据

下列做法可以改善应用程序的敏感的用户个人数据的安全性：

- 按需检索敏感数据
- 加密数据或保护通信信道
- 不要把敏感数据保存在永久性cookie中
- 不要通过HTTP-GET协议来传递敏感数据



3 软件安全设计方法

——会话管理

- 会话管理是应用程序级的责任。而会话安全是整个应用程序安全的关键部分。

下列做法可以改善应用程序的会话管理的安全性：

- 使用**SSL**来保护会话认证**cookie**
- 给认证**cookie**的内容加密
- 限制会话生存周期
- 保护未授权访问的会话状态



3 软件安全设计方法

——加密技术

加密技术的基本形式规定如下：

- 隐私（保密性）。这项服务始终为机密保密。
- 不可抵赖（真实性）。这项服务保证用户不能否认发送过某特定消息。
- 防篡改（完整性）。这项服务防止数据被更改。
- 认证。这项服务确定消息发送者的身份。



3 软件安全设计方法

——加密技术

下列做法可以在使用加密技术时改善应用程序的安全性：

- 不开发自己的加密技术
- 将未加密的数据存储在算法附近
- 使用正确的算法和正确的密钥长度
- 保证加密密钥的安全



3 软件安全设计方法

——参数篡改

- 通过进行篡改参数攻击，攻击者可以修改在客户端和应用程序之间发送的数据。这些数据可能使用查询字符串，表单，**cookie**或在**HTTP**标头里发送。

以下做法有助于保护应用程序的操作不被篡改：

- 加密敏感的**cookie**状态
- 确保用户不能绕过检查
- 验证所有来自客户端的值
- 不要信任**HTTP**标头信息



3 软件安全设计方法

——异常管理

- 安全异常处理可以用来阻止某些应用程序级的拒绝服务攻击，并且它也可以被用来防止对攻击者非常有用的宝贵的系统级信息返回到客户端。
- 一个好的方法是设计一个集中的异常管理和日志方案，并考虑提供挂接到异常管理系统，以支持设备和集中监控，从而来帮助系统管理员。



3 软件安全设计方法

——异常管理

以下做法有助于保护应用程序的异常管理：

- 不要泄露信息给客户端
- 日志要详细记录错误消息
- 捕捉异常



3 软件安全设计方法

——审核和日志

- 应该在应用程序级主动地进行审核和记录日志。
- 通过日志，可以检测看起来可疑的活动。它常常能提供一个全面攻击的早期迹象，
- 日志也能帮助消除用户否认他们的行动的否认威胁。



3 软件安全设计方法

——审核和日志

下列做法可以改善应用程序的安全性：

- 审核和日志访问在应用程序级进行
- 考虑标识流
- 记录关键事件
- 保护日志文件
- 备份并定期分析日志文件



3 软件安全设计方法 ——总结

- 安全应该贯穿于产品开发生命周期的每个阶段，而且应该是一个应用程序设计的重点。
- 这一节中的步骤能够解决在设计和建立安全的应用程序中遇到的这样或那样的挑战。



4 安全设计审查策略

- 建立一个安全的应用程序，需要一个适当的架构和设计。但设计之后的审查也是必不可少的。
- 这可以使在漏洞被利用之前找出潜在的风险，并在花费大量费用重新开发之前就修正这些潜在的风险。



4 安全设计审查策略 ——架构和设计审查过程

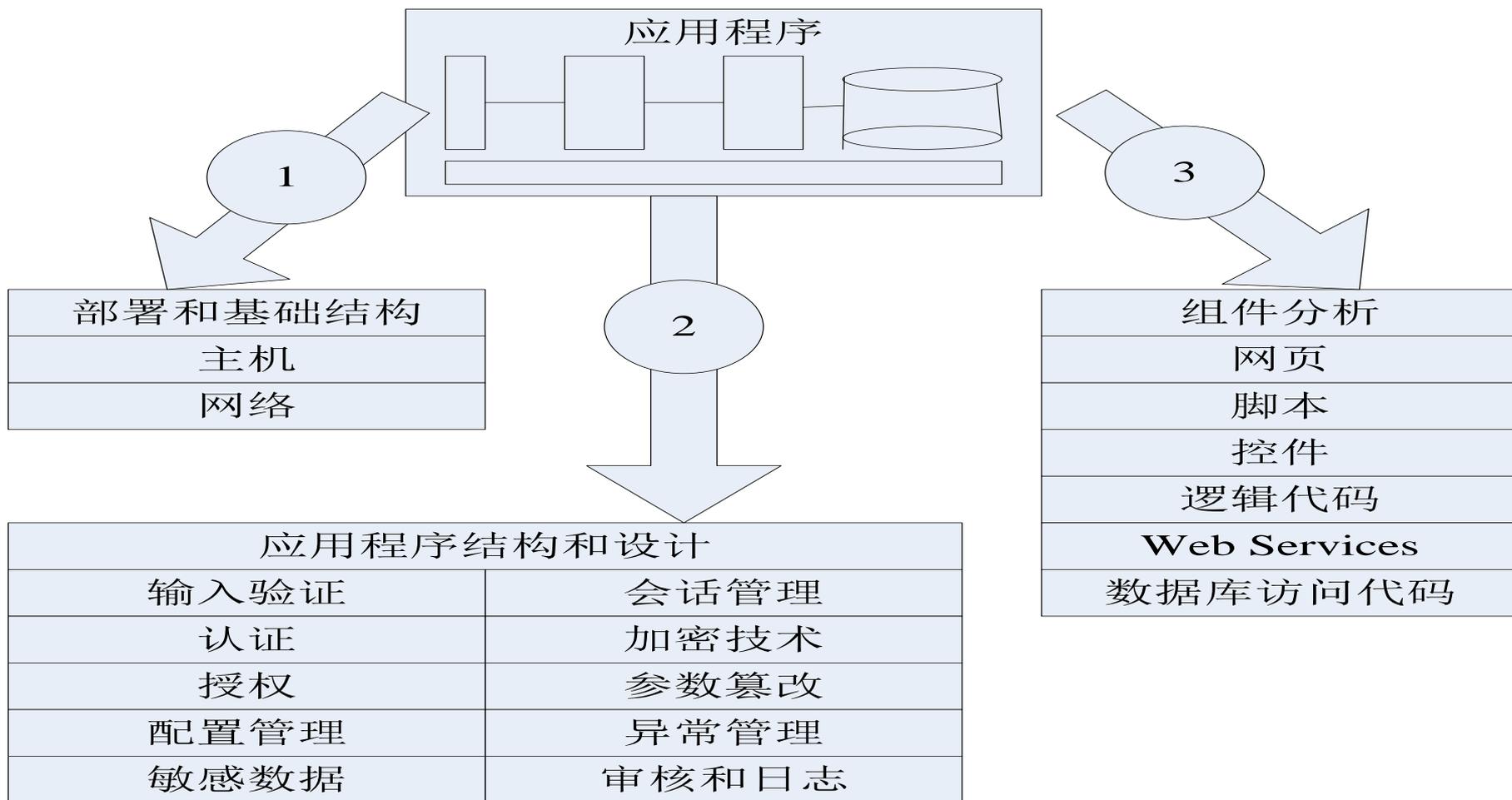
当审查应用程序的架构和设计时考虑以下几个方面：

- 部署和基础结构。
- 应用程序的架构和设计。
- 组件分析。



4 安全设计审查策略

——架构和设计审查过程





4 安全设计审查策略 ——部署和基础结构

审查下列问题以找出潜在的部署和基础结构问题：

- 网络是否提供安全的通信？
- 应用拓扑结构包括一个内部防火墙吗？
- 应用拓扑结构包括一个远程应用服务器吗？
- 基础结构的安全要求会带来什么限制？
- 是否考虑过**WEB**容器问题？
- 目标环境支持什么级别的信任？



4 安全设计审查策略 ——输入验证

- 检查应用程序是如何验证输入的，因为许多应用程序攻击都使用故意畸形的输入。

审查下列问题，以找出潜在的输入验证安全问题：

- 怎样验证输入？
- 怎样处理输入？

漏洞

表现



在超文本标记语言（HTML）输出流中未验证的输入

应用程序可能受到XSS攻击

用来产生SQL查询的未验证的输入

应用程序可能受到SQL注入攻击

依赖于客户端验证

客户端验证很容易被绕过

使用输入文件名，URL 或者用户名来决定安全性

应用程序容易犯规范化错误，从而导致安全漏洞

只为应用程序设计了过滤恶意输入功能

这几乎不可能正确地实现，因为潜在的恶意输入的范围非常大。应用程序应该限制，拒绝和处理输入。



4 安全设计审查策略 ——认证

审查下列问题以找出应用程序执行认证时的漏洞：

- 是否将公共访问和限制访问分离？
- 是否确定了服务帐户的要求？
- 怎样认证访问者？
- 怎样用数据库来认证？
- 是否执行强有力的帐户管理做法？



漏洞	表现
弱密码	增加了受到密码破解和字典攻击的风险
配置文件中的明文证书	能够访问服务器的内部人员和利用主机漏洞下载了配置文件的攻击者立刻就能访问该证书
在网络中传输明文证书	攻击者可以监控网络以窃取认证证书和欺骗身份
越权帐户	增加了泄露进程或帐户的风险
长会话	增加了会话被截获的风险
将认证与个人数据混淆	个人数据适用于永久cookie。认证cookie不应该是永久性的



4 安全设计审查策略 ——授权

审查下列问题，以验证应用程序设计中的授权策略：

- 怎样授权终端用户？
- 怎样授权数据库中的应用程序？
- 怎样限制对系统级资源的访问？

漏洞

表现



依赖于单一的防护手段

如果防护手段被绕过或正确地配置出来了，用户未经授权就可以进行访问

对应用程序的身份未能锁定系统资源

攻击者可以强迫应用程序访问受限的系统资源

未限制程序对数据库的访问

攻击者使用SQL注入攻击来截获，篡改或销毁数据

权限分离的不够

没有制度或能力来为每个用户单独授权



4 安全设计审查策略 ——配置管理

审查下列问题可以帮助验证应用程序配置管理部分的设计方法：

- 是否支持远程管理？
- 是否保护配置存储？
- 是否将管理员权限进行了分离？



4 安全设计审查策略 ——敏感数据

审查下列的问题来帮助验证应用程序处理敏感的数据的方法：

- 是否存储机密？
- 怎样存储敏感数据？
- 是否在网络中传递敏感数据？
- 是否记录敏感数据？



4 安全设计审查策略 ——会话管理

使用下列问题以帮助验证应用程序处理敏感的数据的方法：

- 如何交换会话标识符？
- 是否限制了会话的生存周期？
- 怎样保护会话状态存储的安全？



4 安全设计审查策略 ——加密技术

审查下列问题，以帮助验证应用程序处理敏感数据的方法：

- 是否使用了特别的算法？
- 怎样保证密钥的安全？



4 安全设计审查策略 ——参数篡改

审查下列问题，以帮助确保设计不容易受到参数篡改攻击：

- 是否验证了所有的输入参数？
- 是否在参数中传递敏感数据？
- 是否使用**HTTP**标头数据来提供安全性？



4 安全设计审查策略 ——异常管理

审查下列问题以帮助确保设计不容易产生异常管理安全漏洞：

- 是否使用了结构化异常处理？
- 是否泄露了太多信息给客户端？



4 安全设计审查策略 ——审核和日志

审查下列问题，以帮助验证应用程序使用的审核和日志方法：

- 是否确定过应该重点审核的活动？
- 是否考虑过如何掩盖原始访问者身份？
- 是否考虑过安全的日志文件管理策略？



本节小结

- 1 软件安全设计目标与主要任务
- 2 软件安全设计原则
- 3 软件安全设计方法
- 4 安全设计审查策略