关于优化互联网系统安全开发方法的研究

马全斌

（天津市住房公积金管理中心，天津 300042）

摘　要　随着互联网系统的不断发展，应用安全问题越来越重要，专业的安全检测通常处于项目的中后期，面临着架构修改困难，整改、测试工作量大等问题，因此需要有针对性地进行调整。本文通过对通用互联网项目安全开发情况的分析，有针对性地提出了优化方案，减少了项目后期由于需要修改安全问题而引发的大量问题，提高了互联网系统的安全性。

关键词　互联网　信息系统　安全开发

中图分类号：U293　　**文献标识码**：A　　**文章编号**：1007-0745(2022)09-0016-03

1 互联网系统安全开发过程中的常见手段

随着我国信息化产业的不断发展，互联网系统规模的不断加大，安全问题的重要性不容忽视，开发安全更是重中之重。互联网系统在开发过程中必须要遵守适用的安全开发规范，保证在系统设计的时候能够抵御可预见的安全风险。在集成测试时，应进行源代码扫描和基础渗透测试，提早发现问题的时间，保证项目的如期上线。因此，在实施互联网项目时，在开发阶段和测试的早期阶段，应通过遵守安全开发规范，及时、多次进行源代码扫描和基础渗透测试来保证系统安全。

1.1 安全开发规范

在系统设计时应符合安全开发规范，安全开发规范主要包含以下相关内容：

1.1.1 验证输入内容

有许多方式可以对用户的输入进行处理，不同的输入类型以及不同的应用场景适用不同的验证方式，必要的情况下还可以进行组合验证。

1. 黑名单。黑名单包含一系列具有风险的字符串清单，可以解析已知的攻击模式，通过对用户的输入内容进行匹配，一旦匹配命中，系统会拒绝用户的请求，保证系统安全。如果有新的攻击字符串产生，黑名单需要立刻更新，因此时效性比较差。

2. 白名单。白名单包含一系列安全的字符串清单，验证用户输入是否符合安全内容，符合的请求才能被处理，其他数据会被阻止。这种方式可以有效地解决安全问题，但是通用性较差。

1.1.2 加密或者编码输出内容

对输出内容加密或者进行编码可以有效地防止跨站攻击，同时也可以保证输出内容的完整性以及正确性，对输出内容加密或者进行编码主要有以下几种方式：

1. 对 HTML 代码的输出进行过滤。

2. 对 HTML 代码的输出进行加密或者编码。

3. 对非 HTML 代码的输出，先对内容进行加密或者编码，再输出到页面。

4. 对特殊符号进行编码形式的转换。

1.1.3 安全审计

安全审计分为白盒安全审计和黑盒安全审计，具有以下特点：

1. 白盒安全审计。对系统代码应进行安全审计，及时发现由代码造成的安全问题，并进行修改。白盒审计通常采取自动静态分析技术对应用程序的源代码进行扫描，定位可能存在问题的环节以及数量，验证应用程序的安全性。执行自动静态分析时，不需要运行待分析的系统，因此在查询已知或未知跨站攻击以及 SQL 注入漏洞时更为高效，同时白盒审计过程能与系统开发并行，可以节约大量研发时间并且节省巨大的成本。

2. 黑盒安全审计。在信息化项目的实施过程中，应对已经开发完成的系统进行黑盒安全审计。黑盒安全审计不需要了解系统内部程序逻辑，仅需要模拟用户操作行为，分析程序输入数据与输出数据的对应关系，针对目前主流的安全漏洞进行检测和验证。

1.2 源代码扫描

源代码扫描通常采用静态分析方法。静态分析是指不运行被测程序本身，通过工具扫描和人工确认来检查源程序的语法、结构、过程、接口等来检查程序的正确性。找出程序中存在的漏洞和缺陷，例如不匹配的参数、逻辑错误的循环嵌套、不允许的递归、未使用过的变量和可疑的计算等。

静态分析不需要代码的运行环境，因此适合在项目早期的编码阶段，可以在编码阶段找出可能存在的安全风险代码，这样开发人员可以在早期解决潜在的

安全问题。

通常聘请第三方安全公司对系统进行安全扫描，会有明确的扫描次数，一般是两次扫描：初测和复测。因此为了扫描得更全面，初测一般放在系统测试中后期进行，这个阶段系统开发工作已经结束，测试工作也到了收尾阶段，很少会因为系统修改导致新的安全问题，但是缺点也很明显，就是一旦测试出需要较大改动的安全问题，修改的时间通常不够。

1.3 渗透测试

渗透测试是动态分析，在测试的实施过程中，通常会从以下几个维度来进行：

1. 数据传输及身份认证。

暴力破解：认证方式存在漏洞，可以被绕过，或者没有抵御暴力破解的策略。

密码安全：默认密码未修改，或者密码复杂度不足。

传输安全：隐私数据传输未加密，或者加密的方式存在风险。

信息泄露：敏感信息未经转换就提示给用户，造成信息泄露。

密码修改：修改密码不需要认证，修改密码不需要提供原始密码，修改密码时可穷举原始密码。

登录漏洞：登录存在 SQL 注入。

密码找回：找回密码问题简单，找回密码问题的答案可被穷举，依据找回密码的步骤限定逻辑，找回的密码以非安全方式通知用户。

管理后台泄漏：管理后台对外开放，管理后台密码复杂度无要求。

2. 会话管理。令牌安全：令牌可被猜测，会话令牌固定。

3. 数据验证。路径遍历：恶意的文件读写，以及文件上传、下载漏洞。

XSS：存储型，反射型。

4. 访问越权。用户可访问其他同级别用户的数据内容，低权限用户可访问高权限用户的数据内容。

5. 跨站脚本攻击（XSS/CSS）。

6. 配置管理。使用了不安全的 HTTP 方法，造成遍历网站目录，取得中间件的版本以及服务器开放的端口。

7. 伪造跨站请求（CSRF）。

1.4 系统缺乏安全设计

以电子公积金项目为例，该项目的安全评测报告问题如下：

1. 源代码扫描问题。

2. 漏洞扫描问题。

3. 渗透测试问题。

其中跨站请求伪造，参数硬编码，框架问题，密码加密存储，使用弱加密算法，敏感信息泄露，身份认证，数据验证和配置管理均为安全设计问题，应在安全架构时，充分考虑这些安全问题，在开发过程中注意避免。

2 互联网系统安全开发的优化方案

2.1 加强系统安全设计

系统承建商在进行系统设计时，应该充分考虑系统的安全需求，有针对性地进行设计，系统设计阶段通常包括概要设计和详细设计，当系统规模较小时也可以合并为一个阶段，该阶段的主要安全性工作是进行系统的安全性设计，系统安全设计工作应与常规的设计紧密结合，贯穿在系统设计过程始终。此外，随着系统细节的展开，还要进一步展开安全性分析以发现新的危险，补充系统安全性需求。

在系统建设时，需要考虑该系统涉及了哪些相关系统和硬件，实现了哪些数据的采集、传递、加工和处理功能，系统之间是否有明确的界限，是否存在敏感数据丢失的疑虑，系统管理员是否可信，系统如何初始化，如何建立一个可信的口令机制，如何确认某个操作人员的行为等问题。为了解决这些不确定因素，系统在设计时应建立相应的安全模型，包括：

1. 系统用户应划分为：普通用户、管理员和系统管理员。管理员负责应用配置管理，系统管理员负责管理员和系统后台的管理。

2. 系统划分为以下三个部分：用户界面逻辑、业务逻辑和异常处理。以用户界面逻辑为例，用户界面逻辑主要包括两个部分：数据访问和登录控制。系统在启动时，用户首先要登录系统，通过验证后才可以进行数据访问。通过登录控制界面，可以完成的主要功能包括：口令验证、口令修改、口令相关数据加密、记载登录日志。登录控制在设计时应考虑使用用户 ID，用户输入用户 ID，从权限数据表中提取出相应的用户名，在密码输入时，应考虑使用密码控件，对密码数据进行加密并防止用户远程登录，同时获取一定的客户端信息，建立系统黑名单数据。对于用户的口令修改，应由客户自行完成，而不是系统管理员，系统管理员对用户授权，但是口令由用户在用户界面输入，并加密存储至后台数据库中，避免系统管理员获取用户口令造成泄密，口令的加密方式应使用不可逆的算法，推荐使用 SHA256 或者复合算法。对于初始口令，应要求用户第一次登录时必须更改，强制要求口令为数字加字母加符号的组合，越复杂越长越好。

3. 在系统安全方面要实现的功能包括：访问控制，抗抵赖、数据保密、身份鉴别、授权机制、日志审计。

4. 对应开发规范，应充分考虑系统的身份认证、会话管理、访问授权、数据验证、配置管理、跨站脚

本攻击 (XSS/CSS) 和跨站请求伪造 (CSRF) 的相关设计。在设计时：第一，要确定系统的安全目标，目标清晰有助于将注意力集中在威胁建模活动上，以及确定后续步骤的工作量。第二，创建系统应用程序概述，逐条列出系统应用程序的重要特征和参与者，有助于确定相关威胁。第三，分解系统应用程序，全面了解系统应用程序的结构，尽量发现更相关、更具体的威胁。第四，确定威胁，使用步骤二和步骤三中的详细信息来确定与系统应用程序方案相关的威胁。第五，确定漏洞，检查系统应用程序的各层以确定与威胁有关的弱点。

2.2 定期进行安全开发和测试培训

互联网系统应用安全开发是一项系统工作，安全意识的培养、技术的学习和应用、思路和眼界的开拓三方面内容必不可少，因此需要对相关人员进行必要的培训。

2.2.1 安全意识的培养

从安全事件切入，使相关人员重视系统安全问题，培养安全意识，在工作中充分考虑系统的安全问题。

2.2.2 技术的学习和应用

通过应用来验证自身技术的掌握情况，制定短期目标和长期目标，通过短期目标的不断积累，完成从量变到质变的过程。技术的学习一定要与应用实践相结合，找到差距，认清需求，通过培训不断提高。

2.2.3 思路和眼界的开拓

通过一定的技术和应用实践的积累，摆脱被动接受培训的阶段，对系统应用安全有一定的见解，思路清晰，并向既定目标不断探索和前进。

2.3 自主开展安全开发和测试

2.3.1 提前进行测试，尽早发现问题

与第三方安全评测公司合作，存在一定的约束，主要是评测次数和评测时间需要与公司进行协调，通常情况是两次，第一次安排在系统测试后期，这个阶段系统功能开发基本完毕，大部分测试问题也得到解决，代码相对固定，做安全测试比较具有代表性。第二次安排在上线前一周左右，主要针对第一次测试出现的问题，检查其修改情况。这种安排受公司资源的约束和次数的限制，发现问题较晚，通常不能全部修改，而是根据问题的优先级，着重解决互联网系统的高危问题，时间紧任务重，压力较大。

为彻底解决以上约束，建议自主开展安全开发和测试，不受公司限制，使用内部人员，在开发阶段按照开发的实际情况进行安全测试，包括源代码扫描和渗透测试，其中渗透测试会相对晚一些，需要一定的部署后才能进行，但仍早于公司测试，这样可以提前发现问题，即便是设计问题，也有充足的时间进行调整，早发现早解决。同时，使用第三方安全评测公司进行配合，形成双保险，解决系统的安全开发问题。

2.3.2 熟练使用工具，结合人工筛查

自主开展安全开发和测试，必要的工具是不可缺少的，根据经验，通常使用以下工具：Fortify SCA、AppScan、Kali Linux、Nessus 等，需要注意的是，工具毕竟是自动化测试，会有一定的误报率和漏报率，因此需要定期升级。对于测试结果，需要手工核对，并且按照经验进行手工渗透，完善安全测试结果。

2.4 加强 APP 项目更新的灵活性

APP 项目与普通的 web 项目不一样，APP 升级会对用户体验造成一定的影响，在设计开发时，使用智能更新技术可以解决这个问题。智能更新基于类加载技术，无需 Root 权限自动完成对 APP 更新资源包的检测，下载和资源文件替换，整个更新过程对终端用户的透明化，让 Android 用户享有 IOS 的体验，提升了用户的留存率和转化率。

使用智能更新技术可以带来的好处有：

1. 易操作：上传移动应用后，快速接受并处理，下发处理后程序供用户发布。

2. 可更新范围广：APK 包中的除 androidmanifest.xml 文件和图标文件外的所有文件都可以更新。

3. 增强安全性：可实现对原包 APK 整包加密，并对 APK 包内的所有文件进行单个加密，运行时在内存中文件动态解密，解密后的数据不落地。

4. 强大的版本控制：随时更新、强制更新。可以不经过用户同意进行后台静默更新。

5. 更新包体积小：更新包为差异更新包，能最小化减少下载时间和下载流量，增强用户体验，减缓服务器压力。

6. 不需要权限：无论用户手机是否 root，都能正常运行。

7. 增强用户体验：用户只需安装一次，后续更新无需再次安装即可达到更新效果。

8. 零风险：防止应用被非法加入病毒、计费 SDK、广告 SDK、恶意代码，防止非法篡改。

3 结语

随着互联网的不断发展，系统的安全开发工作越来越重要，本文总结了互联网系统安全开发过程中的常见手段，并从加强系统安全设计、定期进行安全开发和测试培训、自主开展安全开发和测试和加强 APP 项目更新的灵活性四个方面提出了优化互联网系统安全性的方法，希望可以对互联网信息化建设工作有所帮助。