



华中科技大学

HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

网络安全安全学院



构建安全软件与安全开发 生命周期

网络安全安全学院 慕冬亮

Email : dzm91@hust.edu.cn

软件开发生命周期

软件开发生命周期可以有很多模型，如瀑布模型、原型模型等。但是一般说来，软件开发生命周期可以包括以下5个阶段：

(1) 分析阶段。软件需求分析，实际上是回答“软件需要完成什么功能”。它的主要工作是，通过研讨或调查研究，对用户的需求进行收集，然后去粗取精、去伪存真、正确理解，最后把它用标准的软件工程专业语言(需求规格说明书)表达出来，供设计人员参考。

该阶段首先是在用户中进行调查研究，和用户一起确定软件需要解决的问题，此阶段的重要工作有：
建立软件的逻辑模型；
编写需求规格说明书文档，根据用户需求，通过不断沟通，反复修改，并最终得到用户的认可。

软件开发生命周期

(2) **设计阶段**。一般说来，软件设计可以分为概要设计和详细设计两个阶段。该阶段的最终任务是将软件分解成一个个模块(可以是一个函数、过程、子程序、一段带有程序说明的独立的程序和数据，也可以是可组合、可分解和可更换的功能单元)，并将模块内部的结构设计出来。

该阶段的主要工作有：

- 利用结构化分析方法、数据流程图和数据字典等方法，根据需求说明书的要求，设计建立相应的软件系统的体系结构；
- 进行模块设计，给出软件的模块结构，用软件结构图表示，将整个系统分解成若干个子系统或模块，定义子系统或模块间的接口关系；
- 设计模块的程序流程、算法和数据结构，设计数据库；
- 编写软件概要设计和详细设计说明书，数据库或数据结构设计说明书，组装测试计划。



软件开发生命周期

(3) 编码阶段。该阶段主要把软件设计转换成计算机可以接受的程序，选择某一种程序设计语言，编写出源程序清单。

此阶段的主要工作有：

- 基于软件产品的开发质量的要求，充分了解软件开发语言、工具的特性和编程风格；
- 进行编码；
- 提供源程序清单。

软件开发生命周期

（4）**测试阶段**。软件测试的目的是以较小的代价发现尽可能多的错误。要实现该目标，关键在于设计一套出色的测试用例。不同的测试方法有不同的测试用例设计方法，目前常见的测试方法有：

- 白盒测试法。该测试法的对象是源程序，依据的是程序内部的逻辑结构来发现软件的编程错误、结构错误和数据错误(如逻辑、数据流、初始化等错误)，其用例设计的关键，是以较少的用例覆盖尽可能多的内部程序逻辑结果。
- 黑盒测试法。依据的是软件的功能或软件行为描述，发现软件的接口、功能和结构错误。黑盒法用例设计的关键同样也是以较少的用例覆盖模块输出和输入接口。

此阶段的主要工作是：

- 设计测试用例，进行测试；
- 写出测试报告，提交修改部门；
- 继续测试。



软件开发生命周期

(5) **维护阶段**。本阶段主要根据软件运行的情况，对软件进行适当修改，以适应新的要求；以及纠正运行中发现的错误。本阶段工作在已完成对软件的研制(分析、设计、编码和测试)工作并交付使用以后进行，一般所做的工作是编写软件问题报告、软件修改报告。

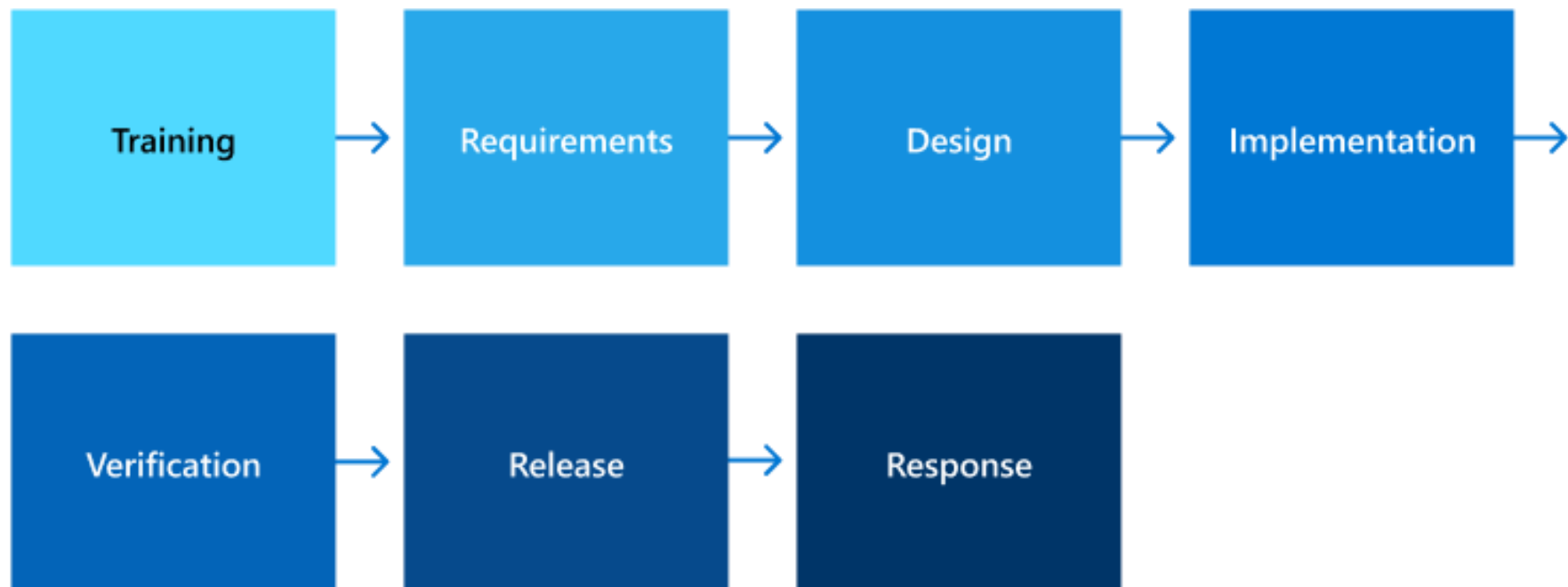
维护阶段的成本是比较高昂的，设计不到位或者编码测试考虑不周全，可能会造成软件维护成本的大幅提高。**事实上，和软件开发工作相比，软件维护的工作量和成本都要大得多。**

在实际开发过程中，软件开发并不一定是从第一步进行到最后一步，而是在任何阶段，在进入下一阶段前一般都有一步或几步的回溯。如在测试过程中的问题可能要求修改设计，用户可能会提出一些需要来修改需求说明书等。

安全并非开发完成后的附属工作

- 开发安全的软件时，安全和隐私不应是事后才考虑的问题，而是必须制定一个正式的过程，以确保在产品生命周期的所有时间点都考虑安全和隐私。
- Microsoft 的安全开发生命周期 (SDL)
- 华为的安全开发生命周期 (SDL)

Microsoft安全开发生命周期 (SDL)



七组件（五个核心阶段和两个支持安全活动）

- 五个核心阶段是要求、设计、实现、验证和发布。
- 两个支持安全活动（培训和响应）

Microsoft安全开发生命周期 (SDL)



必需的安全活动

如果确定对某个软件开发项目实施 SDL 控制，则开发团队必须成功完成十六项必需的安全活动，才符合 Microsoft SDL 过程的要求。这些必需活动已由安全和隐私专家确认有效，并且会作为严格的年度评估过程的一部分，不断进行有效性评析。

培训阶段

- 软件开发知识以技术角度至少一

基本软件安全培训应涵盖的基础概念包括：

- 安全设计，包括以下主题：
 - 减小攻击面
 - 深度防御
 - 最小权限原则
 - 安全默认设置
- 威胁建模，包括以下主题：
 - 威胁建模概述
 - 威胁模型的设计意义
 - 基于威胁模型的编码约束
- 安全编码，包括以下主题：
 - 缓冲区溢出（对于使用 C 和 C++ 的应用程序）
 - 整数算法错误（对于使用 C 和 C++ 的应用程序）
 - 跨站点脚本（对于托管代码和 Web 应用程序）
 - SQL 注入（对于托管代码和 Web 应用程序）

全基础
开发的
页参加

要求阶段

SDL 实践 2：安全要求

- “预先”考虑安全和隐私是开发安全系统过程的基础环节。为软件项目定义信任度要求的最佳时间是在初始计划阶段。尽早定义要求有助于开发团队确定关键里程碑和交付成果，并使集成安全和隐私的过程尽量不影响到计划和安排。

SDL 实践 3：质量门/Bug 栏

- 质量门和 Bug 栏 用于确立安全和隐私质量的最低可接受级别。在项目开始时定义这些标准可加强对安全问题相关风险的理解，并有助于团队在开发过程中发现和修复安全 Bug。

要求阶段

SDL 实践 4：安全和隐私风险评估

安全风险评估 (SRA) 和隐私风险评估 (PRA) 是必需的过程，用于确定软件中需要深入评析的功能环节。这些评估必须包括以下信息：

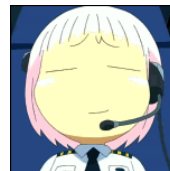
- （安全）项目的哪些部分在发布前需要威胁模型？
- （安全）项目的哪些部分在发布前需要进行安全设计评析？
- （安全）项目的哪些部分（如果有）需要由不属于项目团队且双方认可的小组进行渗透测试？
- （安全）是否存在安全顾问认为有必要增加的测试或分析要求以缓解安全风险？
- （安全）模糊测试要求的具体范围是什么？
- （隐私）隐私影响评级如何？

设计阶段

- **SDL 实践 5：设计要求**
- 影响项目设计信任度的最佳时间是在项目生命周期的早期。在设计阶段应仔细考虑安全和隐私问题，这一点至关重要。
- **SDL 实践 6：减小攻击面**
- 减小攻击面通过减少攻击者利用潜在弱点或漏洞的机会来降低风险。减小攻击面包括关闭或限制对系统服务的访问、应用最小权限原则以及尽可能进行分层防御。
- **SDL 实践 7：威胁建模**
- 威胁建模用于存在重大安全风险的环境之中。

软件设计阶段威胁建模

- ❖ 软件在设计阶段达到的安全性能，将是软件整个生命周期的基础。如果在设计阶段没有考虑某些安全问题，那么在编码时就几乎不被考虑。这些隐患将可能成为致命的缺陷，在后期以更高的代价的形式爆发出来。安全问题，应该从设计阶段就开始考虑，设计要尽可能完善。
- ❖ 传统的软件设计过程中，将工作的重点一般放在软件功能的设计上，没有非常详细地考虑到安全问题。因此，在软件设计阶段，针对安全问题，应该明确以下方面：
 - 安全方面有哪些目标需要达到；
 - 软件可能遇到的攻击和安全隐患；等等。



软件设计阶段威胁建模

- ❖ 一般采用**威胁建模**的方法来在软件设计阶段加入安全因素的考量。威胁建模除了和设计阶段的其他建模工作类似的地方外，更加关心安全问题，是一种比较好的安全问题的表达方法。
- ❖ 分析系统中可能存在的威胁，可能是一件比较繁重的工作，因为很多**威胁是不可预见**的。但是，在设计阶段就尽可能多地将威胁考虑到，在编写代码前修改方案，代价比较小。

软件设计阶段威胁建模

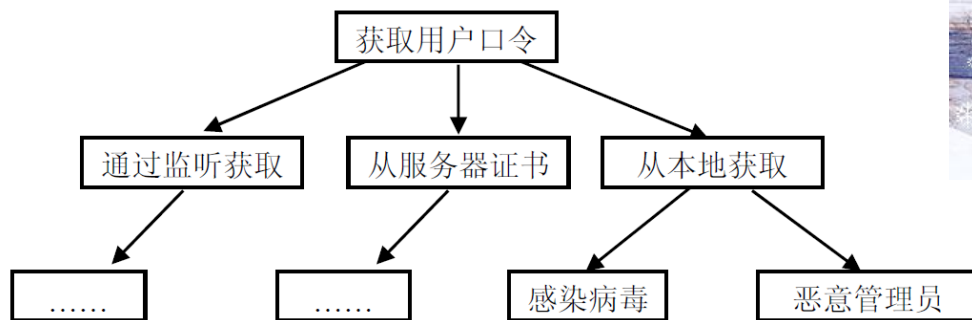
❖威胁建模过程一般如下：

- **在项目组中成立一个小组。**在此过程中，需要从项目组内挑选对安全问题比较了解的人，这些人可能不一定要懂得怎样去编写程序，但是要懂得程序运行的过程中，可能会出现哪些安全问题，或者可能受到什么样的攻击。也可以请用户中的某些人来参与该小组。
- **分解系统需求。**本过程中，可按照需求规格说明书和设计文档中的内容，站在安全角度，分析系统在安全方面的需求。当然，传统的软件工程的一些工具也可以使用，如：数据流图(DFD)、统一建模语言(UML)等。关于这些内容，大家可以参考相应文献。
- **确定系统可能面临哪些威胁。**系统可能遇到的威胁有很多种，在这里可以首先将威胁进行分类，如系统缓冲区溢出、身份欺骗、篡改数据、抵赖、信息泄露、拒绝服务、特权提升等。由于同类的安全问题可以用类似的方法解决，因此该过程可以减小后期工作量。



软件设计阶段威胁建模

❖另外，对威胁进行分类之后，可以画出威胁树，其目的是对软件可能受到的威胁进行表达。如图是一个针对用户口令安全问题画出的威胁树：



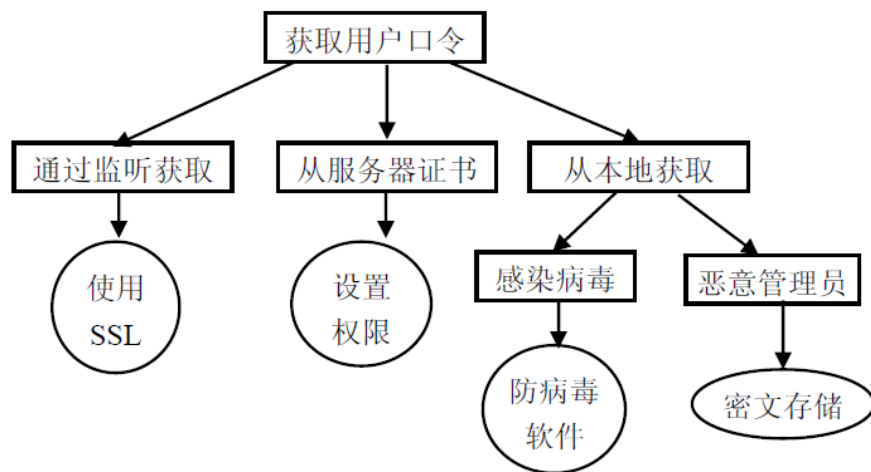
以上威胁树中画出了用户口令不安全中的各种原因，我们就可以针对不同的原因采用相应的措施。

软件设计阶段威胁建模

- **选择应付威胁或者缓和威胁的方法。**很显然，针对不同的安全问题，我们可以选择应付威胁或者缓和威胁的方法。一般说来，可以应付或缓和威胁的方法有很多，但是考虑到实施的成本，根据威胁可能的危害程度，还是有所选择。在面对威胁时，我们可以采用的方法有：
 - **不进行任何处理。**这是不建议的方法。
 - **告知用户。**如果某些威胁无法通过软件工程师自身在产品上施加某种技术来解决，可以告知用户。如提醒用户要杀毒等。
 - **排除问题。**在软件中施加某种技术来避免出现安全问题。
 - **修补问题。**某些问题如果无法预见和解决，可以提供修补接口，待出现问题之后进行扩展。不过这种方案的代价是比较大的，对软件的设计提出了较高的要求。

软件设计阶段威胁建模

- **确定最终技术。**在各种备选的方案中，确定最终选用的技术。一般可以将最终选用的技术，直接在威胁树中描述或者用图表画出来。如图所示的就是针对用户口令安全的威胁树进行的修改：



安全代码的编写

- 实际上，在设计阶段如果尽可能多地将问题考虑周到，在软件的代码编写阶段，只是针对这些问题进行实现而已。不过，在编码过程中也需要考虑一些技巧。如：
 - 内存安全怎样实现？
 - 怎样保证线程安全？
 - 如何科学地处理异常？
 - 输入输出安全怎样保障？
 - 怎样做权限控制？
 - 怎样保护数据？
 - 怎样对付篡改和抵赖？
 - 怎样编写优化的代码？

安全代码的编写

SDL 实践 8：使用批准的工具

- 所有开发团队都应定义并发布获准工具及其关联安全检查的列表，如编译器/链接器选项和警告。

SDL 实践 9：弃用不安全的函数

- 许多常用函数和 **API** 在当前威胁环境下并不安全。

SDL 实践 10：静态分析

- 项目团队应对源代码执行静态分析。

测试阶段

SDL 实践 11：动态程序分析

- 为确保程序功能按照设计方式工作，有必要对软件程序进行运行时验证。

SDL 实践 12：模糊测试

- 模糊测试是一种专门形式的动态分析，它通过故意向应用程序引入不良格式或随机数据诱发程序故障。

SDL 实践 13：威胁模型和攻击面评析

- 应用程序经常会严重偏离在软件开发项目要求和设计阶段所制定的功能和设计规范。因此，在给定应用程序完成编码后重新评析其威胁模型和攻击面度量是非常重要的。

软件的安全性测试

- ❖ 测试是软件发布前所做的重要工作，一方面，需要对软件的可用性进行测试，另一方面，也要对软件的安全性进行最大限度的保障。所以，测试工作决定着软件的质量，是软件质量保证的关键手段。
- ❖ 在充分考虑安全性问题的前提下，安全性测试显得尤为重要。安全测试和普通的功能性测试主要目的不同。普通的功能测试的主要目的是：
 - 确保软件不会去完成没有预先设计的功能；
 - 确保软件能够完成预先设计的功能



软件的安全性测试

- ❖ 安全测试是软件生命周期中一个重要的环节。实际上，安全测试就是一轮多角度、全方位的攻击和反攻击。因此，进行安全测试，需要精湛的系统分析技术和反攻击技术，其目的就是要抢在攻击者之前尽可能多地找到软件中的漏洞，以减少软件遭到攻击的可能性。
- ❖ 安全测试有如下特点：
 - 非常灵活，测试用例没有太多的预见性；
 - 没有固定的步骤可以遵循；
 - 工作量大，并且不能保证完全地加以解决。

发布阶段

SDL 实践 14：事件响应计划

- 受 SDL 要求约束的每个软件发布都必须包含事件响应计划。即使在发布时不包含任何已知漏洞的程序也可能面临日后新出现的威胁。

SDL 实践 15：最终安全评析

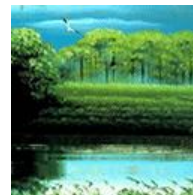
- 最终安全评析 (FSR) 是在发布之前仔细检查对软件应用程序执行的所有安全活动。

SDL 实践 16：发布/存档

- 发布软件的生产版本 (RTM) 还是 Web 版本 (RTW) 取决于 SDL 过程完成时的条件。此外，必须对所有相关信息和数据进行存档，以便可以对软件进行发布后维护。

漏洞响应和产品的维护

- ❖ 在软件开发的过程中，即使在设计、代码编写和测试过程中考虑了安全因素，最终的软件产品仍可能存在漏洞。漏洞一般在用户使用的过程中被发现，此时，迅速**确认、响应、修复漏洞，是非常重要的。**
- ❖ 由于软件的维护是一个长期的过程，因此，软件的维护和跟踪要及时持续，也要花费较大的成本。大型软件公司都会有自己的安全响应队伍，专职处理安全事件，在**发现漏洞后的第一时间采取措施**，以保护客户的利益不被侵害。



漏洞响应和产品的维护

一般来说，正常的漏洞响应可以大致分为以下四个阶段：

- **发现漏洞通知厂商**。在该阶段，漏洞首先由用户报告给厂商所设置的安全响应中心，响应中心经过初步的鉴定，如果确信是一个漏洞，安全响应队伍向漏洞上报者确认已经收到漏洞报告。
- **确认漏洞和风险评估**。安全响应队伍会联系上报者和相关产品的开发部门，以获得更多的技术细节，有时甚至会将上报者和开发团队召集在一起进行讨论。当漏洞被成功重现后，为漏洞定一个威胁等级。
- **修复漏洞**。安全响应队伍和开发队伍协商决定解决方案，并确定响应工作的时间表。开发部门开始修复漏洞。补丁完成后，进行严格的测试。
- **发布补丁及安全简报对外公布安全补丁**。通知所有用户修补该漏洞，在网站上发布安全简报，其中会特别感谢上报漏洞和协助修复漏洞的安全研究人员。

可选的安全活动

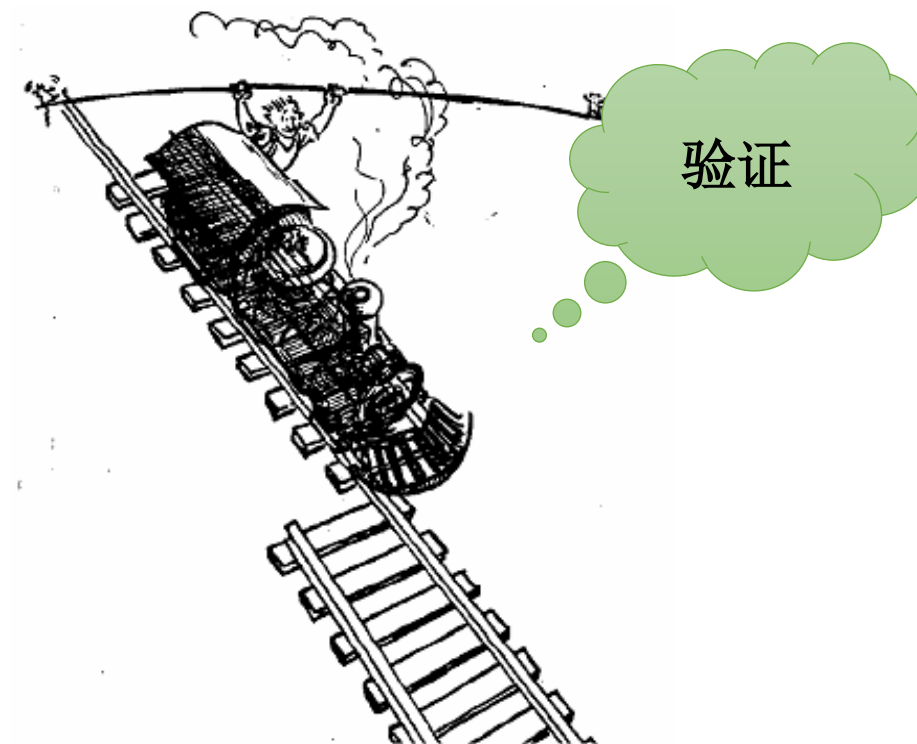
人工代码评析

- 人工代码评析是 **SDL** 中的可选任务，通常由应用程序安全团队中具备高技能的人员或由安全顾问执行。

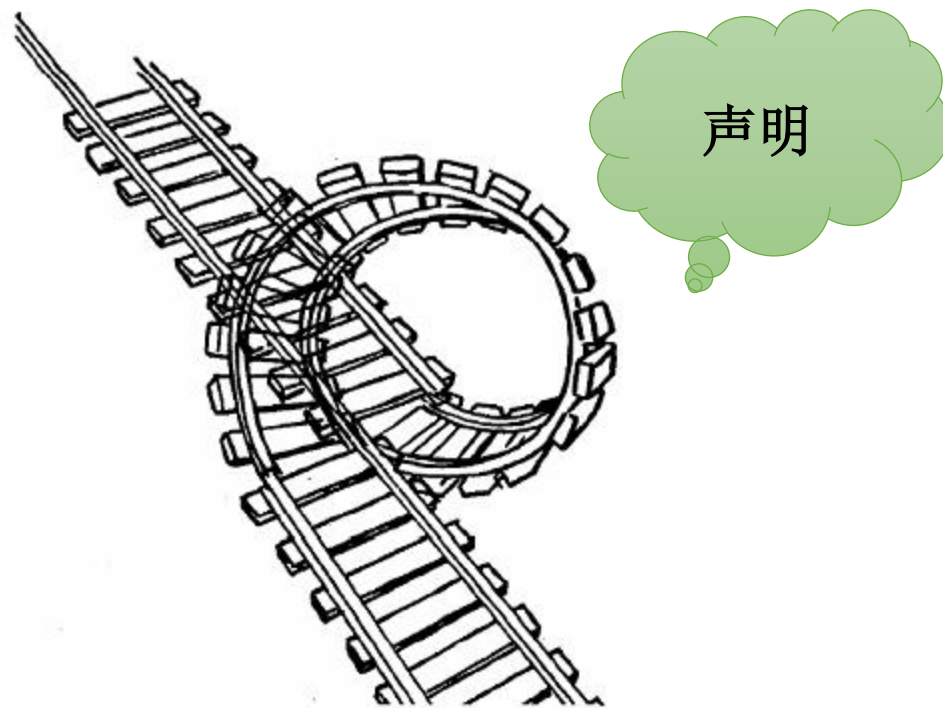
渗透测试

- 渗透测试是对软件系统进行的白盒安全分析，由高技能安全专业人员通过模拟黑客操作执行。
-

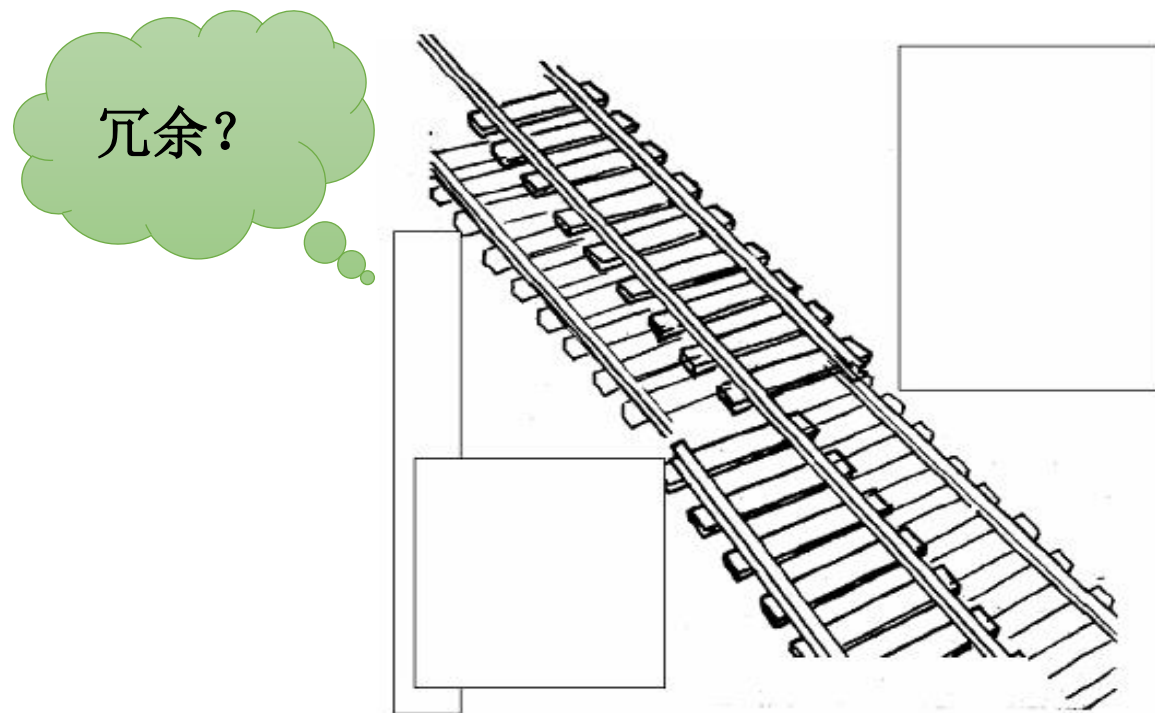
如何处理这些错误和缺陷？ (1/4)



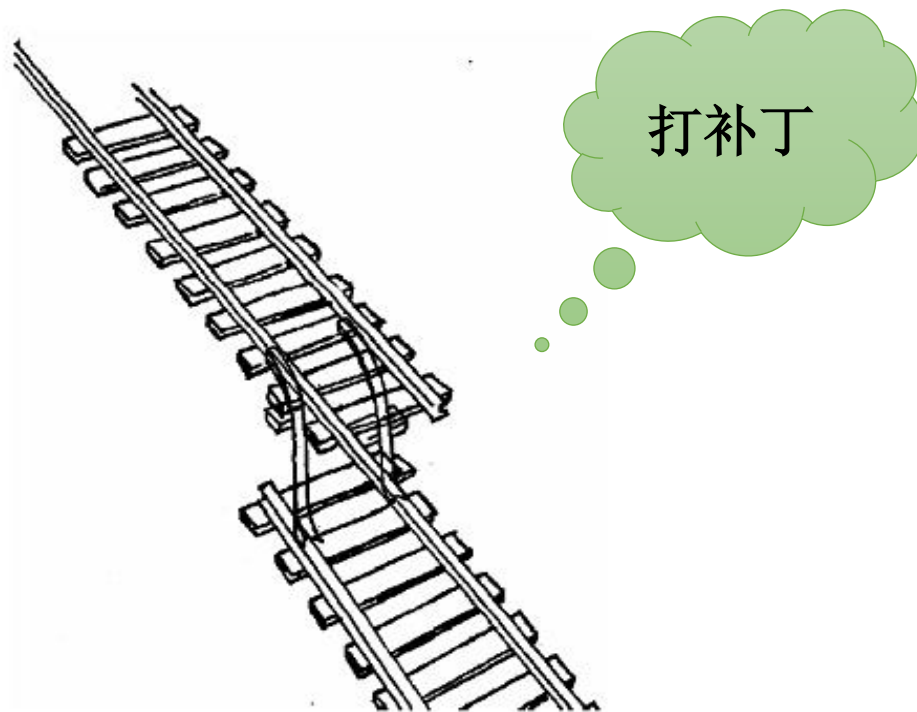
如何处理这些错误和缺陷？ (2/4)



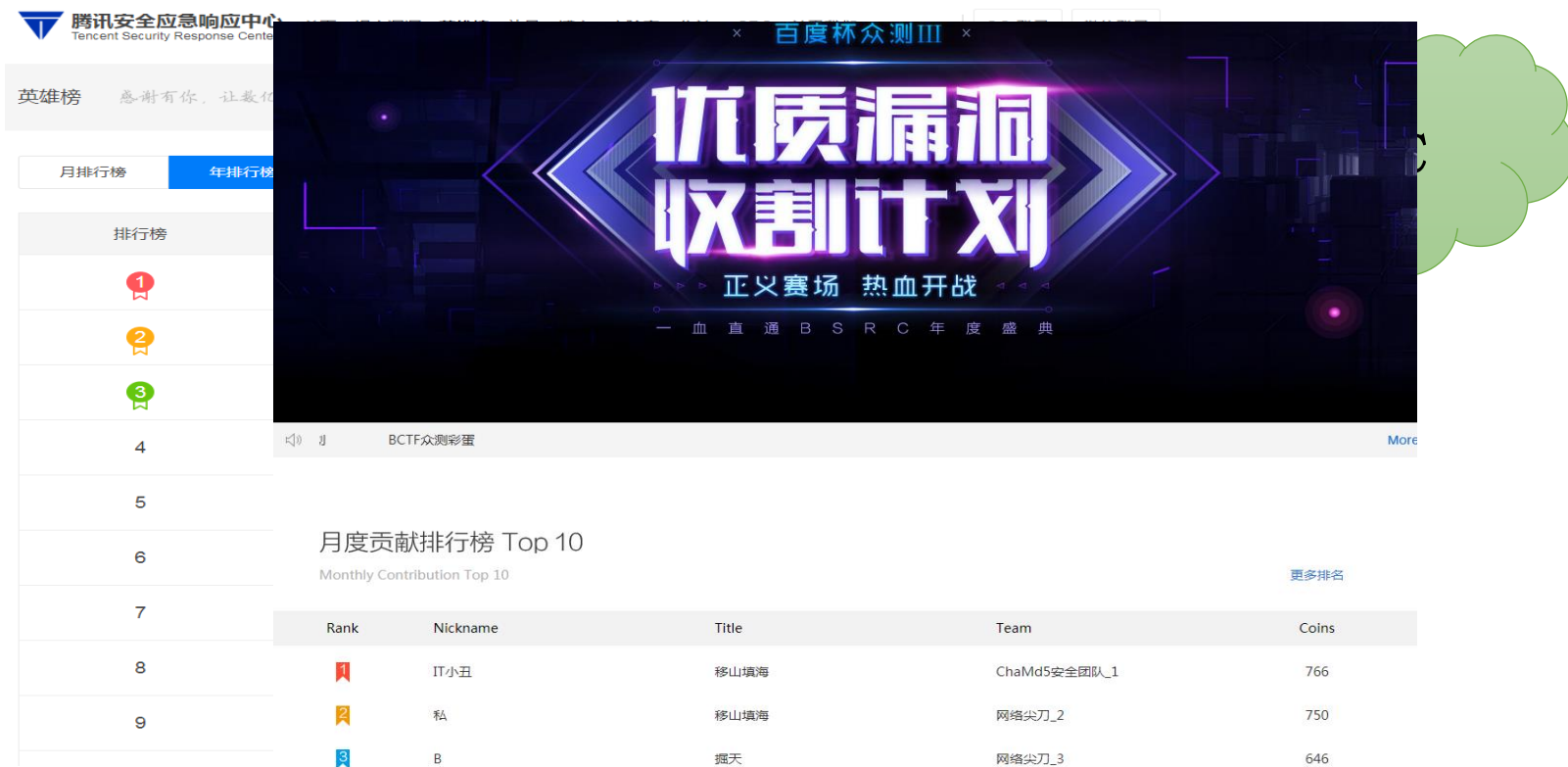
如何处理这些错误和缺陷？ (3/4)



如何处理这些错误和缺陷？（4/4）



如何处理这些错误和缺陷？（4/4）



The screenshot displays the Tencent Security Response Center (腾讯安全应急响应中心) website. On the left, there is a sidebar with navigation links: "英雄榜" (Hero List), "月排行榜" (Monthly Ranking), and "年排行榜" (Annual Ranking). The main content area features a large video player with a dark, futuristic theme. The video title is "百度杯众测III" (Baidu Cup CTF III). The video content shows a large, stylized title "优质漏洞收割计划" (High-Quality Vulnerability Harvesting Plan) in the center, with the subtitle "正义赛场 热血开战" (Justice Arena, Blood-Heating Battle) below it. At the bottom of the video frame, it says "一血直通BSRC年度盛典" (First Blood Direct to BSRC Annual Grand Ceremony). Below the video player, there is a section titled "月度贡献排行榜 Top 10" (Monthly Contribution Ranking Top 10) with the subtitle "Monthly Contribution Top 10". To the right of this title is a link "更多排名" (More Rankings). Below the title is a table with 5 columns: Rank, Nickname, Title, Team, and Coins. The table lists the top 10 contributors.

Rank	Nickname	Title	Team	Coins
1	IT小丑	移山填海	ChaMd5安全团队_1	766
2	私	移山填海	网络尖刀_2	750
3	B	搬天	网络尖刀_3	646