ISTQB® 高级模块

安全性测试

2016 版

国际软件测试认证委员会

高级工作组: Mike Smith （主席）

高级安全性测试人员教学大纲工作组: Randall Rice （主席）, TarunBanga,TazDaughtrey, FransDijkman, Prof. Dr. Stefan Karsch, Satoshi Masuda, RaineMoilanen, Joel Oliveira, Alain Ribault, Ian Ross, KwangikSeo, Dave van Stein, Dr. Nor Adnan Yahaya, WenqiangZheng.

版本历史

版本	日期	备注
0.1	2015/04/24	基线版本的创建来自现有的专家安全性测试人员教学大纲草案 3.9
0.2	2015/06/15	经过 Oslo 会议后整合作者输入
1.0 - Beta	2015/09/20	Beta 发布 – alpha 发布合并评论
1.0 – GA 候选人	2016/04/03	经过考试工作组评审后，更改了 K2 和 K3 的学习目标 4.1.2，并适当的重新措辞。文档已经充分支持 K3 的学习目标
1.0 - GA	2016/03/18	GA 发布 – beta 发布合并评论

目录

致谢

本文档由国际软件测试认证委员会高级工作组的核心团队编写。

核心团队感谢审查小组和所有国家委员会的建议和意见。

在完成本单元的高级程度课程大纲时，安全性测试高级工作组拥有以下成员：

本课程的核心团队作者：Randall Rice（主席），休 TazwellDaughtrey（副主席），FransDijkman，Joel Oliveira，Alain Ribault。

以下人员参加了本课程大纲的审查，评论和投票
（字母顺序）：TarunBanga，Clive Bates，Hugh TazwellDaughtrey（副主席），FransDijkman（作者），Christian Alexander Graf，Wenda Hu，Matthias Hamburg，Stefan Karsch 教授，Sebastian Malyska，Satoshi Masuda，Gary Mogyorodi，RaineMoilanen，Joel Oliveira，MeilePosthuma，Alain Ribault，Randall Rice（主席），Ian Ross，KwangikSeo，Dave van Stein，Ernst vonDüring，Attila Toth，Wei Xue，Nor Adnan Yahaya 博士，杨晓峰，郑文强，左平。

此外，我们承认并感谢专家级工作组的领导和成员提供的早期和持续指导：Graham Bath（专家级工作组主席），Judy McKay（专家级工作组副主席）。

本文件于 2016 年 3 月 18 日由 ISTQB®大会正式发布。

本课程大纲中文版翻译参与者（按姓氏拼音排序）

程国青、胡文达、蒲冬梅、腾康、王轶昆、王帅、左平（组长）

本课程大纲中文版 QA 评审参与者（按姓氏拼音排序）

李华北、徐文叶

0 大纲介绍

0.1 本文档的目的

本大纲构成了专家模块“安全性测试人员”的高级软件测试认证基础。 ISTQB®提供如下教学大纲：
1.向国家委员会转化为当地语言并认证培训机构。国家委员会可能会根据其特定的语言需要调整教学大纲，并修改参考文献以适应当地的出版物。
2.对考试委员会，根据每个单元的学习目标，用当地语言推导考题。
3.培训提供者，制作课件并确定适当的教学方法。
4.将认证考生作为准备考试的资源。
5.致力于国际软件和系统工程界，推动软件和系统测试行业，并作为书籍和文章的基础。

ISTQB®可以允许其他实体将本课程大纲用于其他目的，前提是他们需要事先获得书面许可。

0.2 概览

高级安全性测试工程师资格是针对那些在软件测试方面已经取得高级职位的人员，并希望进一步发展他们在安全性测试方面的专业知识。高级级别提供的模块涵盖了广泛的测试主题。

要获得“安全性测试人员”模块中的高级认证，考生必须持有认证测试人员基础级认证证书，并且满足考试委员会的要求，有足够的实践经验可以获得高级认证，具有至少三年相关学术、实践或咨询经验。请参考相关考试委员会确定其具体的实践经验标准。

0.3 考试

本模块的高级考试都基于本高级安全性测试大纲。

考试形式由 ISTQB®的高级考试指南定义。

考试可以作为认可培训课程的一部分或独立进行（例如在考试中心）。考试可以采取书面或电子方式进行，但所有考试必须进行监督/观察（由国家或考试委员会授权的人监督）。

0.4 本大纲的组织

共有十章。标题显示本章的时间。例如：

1.安全性测试基础 105 分钟。

第 1 章旨在 105 分钟用于教授本章的内容。

具体的学习目标在每章开始时列出。

0.5 定义

软件文献中使用的许多术语可以互换使用。虽然基础和高级水平考试的考生可能只会被问到基于 ISTQB® 标准术语表的问题，但是这一级别的考生应该了解并能够使用不同的定义。

注:“信息保障”(IA)只在第 2.4 条提述。在 2.4.3 中引用之后，断言 IA 应该被视为比“安全性测试”更广泛，就像 QA 应该被视为比软件测试更广泛一样。
“信息安全”一词在第 2.2、2.3.1、2.7.2、6(背景)、6.1.3 及整个第 9 章均有使用。
这里没有使用“网络安全”一词，在某些地方现在被称为 IA。
在本高级课程大纲的每一章开头列出的关键字，要么是在软件测试中使用的标准术语表中定义的，要么是由 ISTQB®发布的，要么是在参考文献中提供的。

0.6 细节层次

本教学大纲的详细程度允许进行国际一致的教学和考试。为了实现这一目标，教学大纲包括：
•描述高级水平意向的总体教学目标
•学习每个知识领域的目标，描述认知学习的结果和要实现的心态
•要传授的信息列表，包括描述，以及必要时对其他来源的引用
•描述要教授的关键概念，包括已接受的文献或标准等来源
•本教学大纲中可能提及某些工具，方法和商标。本教学大纲并非旨在宣传或推荐任何特定的安全解决方案。

教学大纲内容不是高级安全性测试人员整个知识领域的描述; 它反映了高级安全性测试人员培训课程中所涉及的详细程度。

0.7 学习目标/知识层级

即使没有在学习目标中明确提及，至少应该记住（K1）和理解（K2）本大纲的所有内容，术语和所列标准的主要内容（目的）。

以下学习目标被定义为适用于本教学大纲。课程大纲中的每个主题都将根据学习目标进行检查。

等级 1：记住（K1）
考生会认识，记住并回忆一个术语或概念。
关键词：记住，回忆，认识，了解。
例子
可以认识到“风险”的定义如下：
•“可能导致未来负面后果的因素;通常表示为影响和可能性。“

等级 2：理解（K2）

考生可以选择与该主题相关的陈述的原因或解释，并且可以对事实（例如比较术语），测试概念，测试程序（解释任务序列）进行总结，区分，分类和举例。

关键词：总结，分类，比较，映射，对比，例证，解释，翻译，表示，推断，总结，分类。
例子
解释为什么应尽早设计安全性测试的原因：
•在寻求更便宜的解决方案时查找安全缺陷和漏洞
•避免构建易于不断修补安全漏洞的系统或应用程序

等级 3：应用（K3）
考生可以选择一个概念或技术的正确应用，并将其应用于给定的上下文。 K3 通常适用于程序性知识。没有涉及的创造性行为，例如评估软件应用程序或为给定的软件程序创建模型。当提供模型时，教学大纲解释了从该模型创建测试用例所需的程序步骤，这就是 K3。
关键词：执行，执行，使用，遵循程序，应用程序。
例子
•使用安全性测试用例创建的通用过程来从给定状态转换图中选择测试用例，以涵盖所有转换。

等级 4：分析（K4）
候选人可以将与程序或技术有关的信息分解为其组成部分以便更好地理解，并且可以区分事实和推论。典型的应用是分析文档，软件，项目情况并提出适当的措施来解决问题或任务。
关键词：分析，区分，选择，结构，重点，属性，解构，评估，判断，监控，协调，创造，综合，产生，假设，计划，设计，构建，生产。
例
•分析产品安全风险并提出预防性和纠正性缓解措施。
•选择安全性测试工具，在过去的安全失败的特定情况下最合适。

参考（对于学习目标的认知水平）
Bloom，B. S.（1956）。教育目标的分类，手册 I：认知领域，David McKay，Co. Inc.
Anderson，L.W。和 Krathwohl，D.R。（编）（2001）。学习，教学和评估的分类学：修订布卢姆的教育目标分类法，Allyn＆Bacon。

1 安全性测试基础-105 分钟

关键词

数据隐私，道德黑客，信息安全，渗透测试，风险评估，风险揭露，风险缓解，安全攻击，安全审计，安全策略，安全程序，安全风险

安全性测试基准的学习目标

1.1 安全风险

AS-1.1.1 (K2) 理解风险评估在提供安全性测试计划的信息中所扮演的角色且设计和调整安全测试的业务需求

AS-1.1.2 (K4) 识别重要的资产需要保护，每个资产的价值和数据需要评估每个资产所需的安全级别

AS-1.1.3 (K4) 在特定情况下，分析风险评估技术的有效利用来确定当前和未来的安全威胁

1.2 信息安全政策和程序

AS-1.2.1 (K2) 理解安全政策和程序的概念和它们是如何应用于信息系统

AS-1.2.2 (K4) 分析一个给定的一组安全政策和程序以及安全性测试结果来确定有效性

1.3 安全审计以及它在安全性测试中的角色

AS-1.3.1 (K2)了解安全审计的目的

功能测试是基于各种各样的项目，如风险，需求，用例和模型。安全性测试是基于这些规范的安全方面并且试图核实和验证安全风险，安全进程和政策，攻击行为和已知的安全漏洞。

1.1 安全风险

1.1.1 风险评估在安全性测试中的角色

安全性测试目的是基于安全风险的，通过执行安全风险评估来识别这些风险。一般的风险管理技术在[ISTQB®_FL_SYL] 和[ISTQB®_ATM_SYL]有所描述。

风险是衡量一个实体受到潜在情况或者事件的威胁程度，典型的功能是：

- 如果情况或事件发生，这种负面影响会上升
- 发生的可能性

信息安全风险是指由于信息或信息系统的保密性、完整性或可用性的丧失而产生的风险，反映出对组织运行(即组织资产、个人、其他组织和国家。(NIST 800 - 30)

安全风险评估的作用是让组织了解哪些领域和资产可能处于风险中，并确定每个风险的大小。对于安全性测试人员，安全风险评估可以是安全性测试计划和设计的丰富信息来源。此外，可以使用安全风险评估来确定安全性测试的优先级，以便最高级别的测试严谨性和覆盖率可以集中在风险敞口很大的领域。

通过基于安全风险评估对安全性测试进行优先级排序，测试将与业务安全目标保持一致。然而，要实现这种一致性，安全风险评估必须准确地反映组织的安全威胁、受影响的涉众和要保护的资产。

重要的是要理解，任何风险评估(安全性或其他)只是给定时间点上的一个快照，并且基于可能导致无效假设和结论的有限信息。自从新的威胁每天出现以来，安全风险在组织和项目中不断变化。因此，应该定期进行安全风险评估。执行安全风险评估的确切时间间隔根据组织和 it 经历的更改程度而变化。一些组织以 3 到 6 个月为基础进行安全风险评估，而其他组织则以每年为基础进行评估。

风险评估的另一个问题是参与者的知识水平。由于缺乏详细的信息，可能会遗漏风险。此外，如果人们不了解安全威胁和风险，就可能忽略风险。因此，最好征求各种人的意见，并仔细注意他们提供的信息中所包含的详细程度。

这是一种现实的期望，即错误的假设可能会导致在评估中遗漏重要的安全风险。处理丢失或不完整风险信息的可能性的方法包括使用已建立的安全风险评估方法作为检查表，并从多个人员获得输入。这种方法可以在[NIST 800-30]中找到。

1.1.2 资产识别

并不是所有的信息都是数字格式，如复制文件（合同，规划，书面笔记。书面形式的登录和密码）。虽然不是数字格式的信息，但可能是重要信息。因此,问题是,哪些信息是数字的,哪些不是?也许保护的资产是数字和物理共同的格式。当识别担保资产时,应该问下面的问题：

哪些资产对组织有价值？

高价值的敏感信息的例子包括：

- 客户数据
- 商业计划
- 公司开发的专有软件
- 系统文献
- 公司财产的图片和图表
- 知识产权(如流程、商业机密）
- 金融电子表格
- 演讲和培训课程
- 文件
- 邮件
- 雇员记录
- 税务报表

虽然很多资产信息化，但是一些资产有可能是在一个组织中的物理或无形的形式，这些资产的例子包括：

- 正在开发的新设备的物理原型
- 提供服务的能力
- 公司的声誉和信任

资产有多大价值？

许多敏感资产有有形资产的价值，其他的在成本和损失的后果方面更慎重。例如,竞争对手的商业计划会有什么影响？

很难评估确定的价值,然而,一些方法来确定数字资产的价值包括：

- 未来资产产生的收入
- 竞争者可能获得的信息的价值
- 重新创建资产所需的时间和精力
- 当需要时不能产生所需的信息的罚款和处罚,例如,审计或诉讼
- 客户数据损失的罚款和处罚

数字资产在哪里？

过去,数字资产被存放在服务器、台式电脑外围设备,如磁盘或光盘，虽然这是一个过时的和无组织的方法,可能仍然有敏感数据在旧 CD、DVD 和 USB 驱动器。一个更安全存储数据资产的方法是使用安全的企业服务器，对敏感数据加密。在安全的服务器上访问敏感数据,应该需要身份验证和授权。此外,其他可能需要安全保护,如数字证书在互联网上用于访问敏感信息。

存储在改变。现在可以在移动设备上存储大量的业务数据,如智能手机、平板电脑和 U 盘。当数字信息被迁移到云存储,有一组新的基于数据访问的安全问题规范。数据存储主要的问题来自过去的案例，在这个案例中受托人带着敏感数据简单地走出公司大楼并且硬盘装满了私人客户和业务数据。在美国这样一个案件牵涉到政府安全机构的安全区域内一个硬盘被盗，包括工资和超过 100000 名现任和前任员工银行信息。[Washington Post, 2007]。

如何访问这些数字资产？

- 访问数字资产的常用方法包括：
- 计算机在局域网或无线网络的访问
- 远程访问通过一个虚拟专用网(VPN)或云驱动器
- 通过物理数据存储(CD、DVD、USB 驱动器)从一个人到另一个人这是一个低技术含量的但很常见的做法
- 通过电子邮件发送文件

如何保护数字资产？

- 有很多的方法保护数字资产包括：
- 加密（什么类型和强度，谁有密码）
- 身份认证和令牌（是否需要数字证书？密码策略是否足够并被遵守？）
- 权限（哪个级别的特权被授予处理数字资产的用户）

1.1.3 分析风险评估技术

- 安全风险评估过程非常类似于一个标准的风险评估,主要区别是关注于安全相关的领域
- 安全风险评估应该包括外部安全性测试的利益相关者的观点（即公司外部的参与项目/产品相关的人或当事人，在项目/产品的安全性上有明确的利害关系）这些利益相关者包括:
- 客户和用户——有助于理解，安全性测试投入，建立良好的沟通
- 公众和社会——重要的是信息安全是全体共同的努力和责任
- 监管机构——确保遵守适用关于信息安全的法律是必要的
- 做一个风险评估任务包括以下准备：
- 确定评估的目的
- 确定评估的范围
- 确定与评估相关的假设和约束
- 确定作为投入评估的信息源
- 确定在评估中使用风险模型和分析方法（即评估和分析方法)
- 进行风险评估包括以下具体的任务：
- 识别与组织相关的威胁源

- 确定威胁事件可能产生的来源
- 确定组织内部的漏洞，这些漏洞可能通过特定的威胁事件和诱发条件被威胁源利用，诱发条件可能会影响到这种成功利用
- 确定被识别出的威胁源发起特定威胁事件的可能性和威胁事件成功的可能性
- 确定漏洞被威胁利用所产生的对组织业务，资产，个人，其他组织和国家的不利影响交流和分享信息包括以下具体的任务[NIST 800-30]：

沟通风险评估结果

- 在风险评估执行过程中共享信息，以支持其他风险管理活动

1.2 信息安全策略和程序

1.2.1 理解安全策略和程序

在信息安全策略中根据业务模型改变组织机构是常见的，特定的行业和组织面临独特的安全风险。即使有各种各样的变化，安全策略的目标是相似的。所有安全策略的基础应该是安全风险评估，它检查特定的安全威胁以及它们对组织的影响[Jackson, 2010]

安全策略的例子包括,但不限于以下内容[Jackson, 2010]：

可以使用——该策略定义了计算机系统用户必须遵循的操作，以便符合组织安全策略和程序。该策略涵盖了使用数字资源（如网络、网站和数据）的可接受和不可接受的行为。此外，该策略也适用于于组织系统的内部和外部用户。对于系统的用户来说，在任何时候都要理解并遵循这个策略是很重要的。为了防止混淆和意外违反策略，它应该定义关于可接受行为、不可接受行为和必需行为的特定规则。

最小使用——该策略定义了需要执行特定的任务最低级别的访问，这一政策的目的是阻止人们被授予访问权限超过他们执行的任务所需。当访问权限高于需要有可能会无意或有意滥用用户权限。

网络访问——该策略定义了访问各种类型网络的标准，如局域网（lan）和无线网络。此外，该策略可以在网络上定义什么是允许的和什么是不允许的。这一政策通常禁止用户向网络添加未经授权的设备，如路由器和热点。

远程访问——这一策略要求这样远程网络访问可以授予内部员工和外部用户(非雇员)。VPN 使用往往是涵盖在这个策略中。

互联网接入——这一策略定义了组织员工和组织客人对互联网的使用许可。这一策略的范围包括可以和不可以被访问的网站类型，如赌博或色情网站，也涉及是否允许非商业性使用互联网。尽管策略中涉及的一些内容也可以在可以使用策略中得到解决，但一些组织选择单独定义这一策略，因为在互联网上开展业务的人数很多。

用户账号管理——该策略定义了用户账号的创建，维护和删除。该策略还包括对用户账号的定期审计，以确保遵守策略。

数据分类——从安全的角度来看有很多方法可以分类数据。在本大纲里,敏感数据被用作通用术语指为避免损失任何必须被保护的数据。数据分类策略定义了被认为是敏感且必须被保护的不同类型的数据。通过有数据分类策略，组织可以创建控件，以根据其对组织及其客户的价值来保护数据。通常,业务领域中创建数据基于标准的分类结构。

下面是一个数据分类例子（来于业务环境）：

- 公开：任何人, 无论组织内或组织外,可以查看这些数据(例如,对外的文件和网页）
- 保密：这通常是创建内部文件时的缺省分类。这些文件可以包括电子邮件、报告和在组织内部的演讲，销售报表就是一个例子。只有经过授权的用户应该能够处理这种级别的信息，在与第三方如顾问，分享这类信息之前,保密协议往往是需要的。
- 高度机密：这是对敏感信息更高层级的保密,应该只提供给组织里的特定人员。这些信息将包括如商业机密、战略规划、产品设计和非公开财务数据。分享这种类型的数据是不允许的,除非数据的所有者明确允许。
- 私有：这些信息通常被限制在组织的官员中，他们必须被特别授权才能访问它。如果披露,这些信息可能产生重大负面影响,比如财务伤害。因为是和损失相关的高风险,必须极其谨慎的保护私用信息。这些数据可以包括研究和发展的信息，合并和收购计划，以及客户信息,如信用卡和账户信息。
- 秘密：在公司背景下，这是一个组织为了变革从外部获取的信息，无论组织内或组织外都不允许变得尽人皆知。公司背景下的一个例子是由一名顾问创建的一份关于一种新型技术的设计文档，涉及与其他公司合作，每一个公司都必须将此信息保密，直到技术准备好被披露为止。它可以与高度机密相媲美，它可能对组织本身没有任何实际价值。在这方面,它不同于商业秘密。然而,秘密信息的披露可能对组织，其他组织或国家造成伤害。在军方和政府的背景下，这些信息可以被开发或获得,但只有通过安全调查的人才可以知道。这将包括科学细节、包含新技术发展的研究项目、对国防至关重要的直接应用于军事的技术。

配置和变更管理——这种策略可以有一个正常的操作上下文，例如描述如何管理和配置系统的变更，以防止由于意外的影响而导致停机。从安全的角度来看，配置管理控制如何将安全设置应用于安全设备和应用程序。风险在于，未经授权的对安全设备的更改可能导致安全漏洞，而这种漏洞可能不会被发现。

另一个风险是未经授权的更改代码或应用程序配置，可能会造成安全漏洞。该策略包括要使用的标准配置、所有更改的审批流程以及出现问题时的回滚过程。该策略适用于组织中的所有 IT 服务、应用程序和设备。

服务器安全——该策略向服务器所有者传达遵守公司安全实践的责任，在安装、配置和操作服务器和系统方面其遵循公司安全措施以及其业界最佳实践。此外,基线配置是强制进行定义和维护的。这一策略实践中描述的示例包括安全需求,备份和恢复,并限制运行应用程序所必需的活动服务。这一策略可能也包括监控和审计的要求,确保服务器配置和更新正确。

移动设备——移动设备有一组独特的安全问题，因此为移动设备制定一个单独的策略是需要的。例如,笔记本电脑和智能手机很容易丢失或被盗,导致公司和私人数据的损失。这些设备有很高的接触恶意软件的风险。这些风险需要特定的规则和采取必须的预防措施来降低风险和限制组织面临的安全威胁。这一策略要求可能包括必须加密数据，安装和维护软最新版本的反恶意软件，访问设备时需要密码。此外，可以在移动设备上保存的组织信息的类型也在该策略中定义。物理安全也可以解决,比如为笔记本电脑安装电缆锁，有报告丢失或被盗设备的程序。

访客访问——该策略定义了应该采取的保护组织的实践，同时允许公司在组织网络上接待来宾和其他人。这一策略的一个方面是要求客人阅读和同意许可条约后再允许他们访问网络。这个政策可以以多种方式实现,如有客人签署许可条约,然后提供一个临时代码访问。这一策略的主要目的还是强化组织的安全标准和提供程序允许客人访问网络或互联网。

物理安全——该策略定义了物理设施所需的控制，因为在物理上接近安全设备会增加安全漏洞的风险。这一策略也可以涵盖其他风险,如功率损耗、盗窃、火灾和自然灾害，还指出是哪些设备可以带出或带进公司,特别是对于那些有敏感信息的区域。

密码策略——该策略定义了强密码和其他安全密码实践的最低要求，如强制密码更改之间的时间间隔，人们如何保护他们的隐私密码（如不使用浏览器的"记住密码"功能，禁止共享密码和禁止通过电子邮件传播密码）这一策略适用于应用程序,用户账户和任何其他需要密码的地方。

恶意软件防护——这个策略定义了一个防御和行为的框架来防止、检测和删除恶意软件。由于恶意软件和间谍软件来源各种各样，这对组织中的每个人来说都是一项重要策略，可以理解和遵循。例如,这一策略可能会限制使用 USB 驱动器。

事件响应——这一策略描述了如何应对与安全相关的事件，这些事件的范围可以从恶意软件和违反可以使用策略到未经授权访问敏感数据。重要的是在一个事件发生之前制定好这个策略，避免不得不根据具体情况确定适当的答复。这一策略还涉及沟通,包括媒体反应和执法通知。

审计策略——这一策略授权审计员为进行审计的目的请求访问系统，审计团队可能需要访问日志数据,网络流量记录和其他法务数据。

软件授权——这一策略指出组织如何获得和使用许可软件，如果违反了商业软件许可证，该组织将面临罚款和法律诉讼的风险。由于这个原因,许可证识别和监控是重要的。在这个策略里，下载和安装未经批准的软件是一个关键的禁令。

电子监控和隐私——组织有权利和责任来监控整个公司电子通讯硬件和资源，这包括电子邮件和社交媒体，该策略概括了组织执行哪些监测，哪些数据需要收集，不同国家之间的法律不同,因此写这个策略之前需要法律顾问。

安全程序

安全程序指定在实施特定政策或控制时应采取的步骤，以及针对特定安全事件采取的步骤。正式的、有文件的程序有助于实施安全策略和强制控制。

策略、标准和指导方针描述了应该到位的安全控制，一个过程描述具体细节并解释如何一步一步地实现安全控制。例如,一个程序可以书面解释如何授予用户访问级别，详细描述每个需要采取的步骤来确保获得正确的访问级别，以便用户权限满足适用的政策、标准和指导方针。

1.2.2 分析安全策略和程序

评估安全策略和程序之前，确定评估的目的和定义一组标准来判断策略和程序的充分性是重要的。在某些情况下,标准可以由诸如 COBIT 之类的标准来定义(COBIT)、ISO27001[ISO27001]或 PCI(PCI)。

此外,必须要定义的是：

- 评估特定领域的技能和知识需要哪些资源
- 如何衡量策略和程序的充分性
- 如何衡量和评估(如有效性、效率、可用性,采用性）
- 在组织中策略和程序的位置
- 一份指导评估和增加一致性的清单

检查表作为指南,指导审计人员在哪里查看和期望什么?诸如密码审计工具之类的工具可能有助于测试某些控制，以确定它们是否实现了目标，并生成可以在稍后进行风险评估时使用的数据。审计人员寻求在政策、控制和标准方面找到“证据”。下面列表中的一些任务本质上是静态的，而其他的任务，例如观察运转中的流程是动态的。审计人员执行以下操作：

- 检查系统文档
- 调查人们对策略和程序的有效性的看法
- 访谈被控制过程涉及到的关键人员
- 执行证人系统和流程
- 分析以往的审计结果来发现趋势
- 分析日志和报告
- 审查技术控制配置,如防火墙配置和入侵检测系统配置
- 交易异常或可疑交易的样本数据

控制：

安全控制是一种技术或行政保障或对策，用来避免、抵消或减少因以他们的匹配漏洞相威胁造成的损失或不可用性,即安全风险[Northcutt, 2014]。例如,安全控制在工资系统可能是两个人必须分别批准对员工工资率信息的更改。安全性测试人员必须了解他们组织中的特定控制，并在安全性测试中包含对他们的测试。

安全控制的主要类型是管理、技术和物理。在每个类别下,特定的控制,可以实现预防,侦探,纠正或恢复。这些控制类型一起工作,一般来说,每个类别必须提供控制来有效保护资产。

关键安全控制前 20 名的列表可以在 www.sans.org 上找到。

安全性测试:

与安全策略和过程的静态分析相比，安全性测试的主要区别在于使用了专门的测试设计，其测试结果用于检验或验证安全策略和过程有效性。这些测试关注的是安全策略可能存在的，随之而来的风险，但在保护资产方面没有效果。

在执行安全策略和程序评估时，也有可能被告知做某项任务。对这些任务的安全性测试可以帮助确定安全策略和程序实际上在实践中的有效性。例如,在纸面上看起来密码策略和程序似乎是合理的和有效的,但当使用一个密码破解工具,这个程序可能达不到目标。

安全策略和程序可以是一个安全性测试的来源，然而,安全性测试人员必须记住,攻击总是在不断演变的。新的攻击会出现在任何软件应用程序,新的缺陷会变得明显-所有这些都是从攻击者的思维模式中执行安全性测试的理由。

1.3 安全审计及其在安全性测试中的作用

安全审计是一个手动的检查和评估,确定一个组织安全流程和基础设施的弱点。程序级别的安全审计（例如，审查内部控制）可以手动执行。架构级别的安全审计通常使用安全审计工具执行，这些工具可能与特定的供应商解决方案相一致，用于网络、服务器架构和工作站。

就像安全性测试,安全审计并不能保证所有漏洞被发现。然而,审计是一个更安全的活动过程来识别问题并指出需要修复的地方。

在一些安全审计方法中，测试是执行审计过程的一部分。然而，安全审计的范围远远大于安全性测试。安全审计通常会检查难以直接测试的过程、策略和控制等领域。安全性测试依靠技术保障来确保安全，如防火墙配置，正确的身份验证和加密应用程序和应用程序的用户权限。

有五个安全审计支柱:

评估——评估文件和识别潜在的威胁，关键资产，公司的政策和程序，管理对风险的容忍度。评估并不是一次性的事件。因为环境和业务不断在变化,评估必须定期执行。这也提供了机会,知道安全策略仍然是适用的和有效的。

预防——这超越技术，包括行政管理、运营和技术控制。预防不是仅仅通过技术来完成，而且通过政策、过程和意识。虽然预防所有的攻击是不现实的，但防御的组合可以使攻击者更难以成功。

检测——检测是如何侦测一个安全漏洞或入侵。没有足够的检测机制，就不知道网络的风险已经受损。侦测可以识别安全事故并在网络提供可见性活动。对事件的早期发现有助于快速恢复服务。

反应——良好的安全防护和检测机制可以大幅缩短反应时间。虽然安全漏洞是坏消息，但重要的是是否发生。快速的反应时间能减少事件的暴露。快速反应需要良好的预防性防御和检测机制来提供响应所需的数据和上下文。事件反应的速度和效率是一个组织的有效性的关键指标。

恢复——首先了解恢复发生了什么，这样系统就不会再次出现同样的错误。恢复期间并不意味着恢复系统，还有根本原因分析，决定需要做出什么来改变过程、程序和技术来减少相同类型的漏洞。审计师必须确保组织有一个恢复计划,包括防止未来类似事件的方法。

1.3.1 安全审计的目的

下面列出的事项可能会发现在一个安全审计中：

- 物理安全不足。安全策略可能需要对存储和传输中的所有客户数据进行加密。例如，在审计过程中，发现每周通过物理报告向所有经理发送一次客户信息文件，该报告每周都会被丢弃。但有人发现，有些经理不小心将物理报告丢到垃圾桶里，任何翻找垃圾桶的人都可以在那里找到这些报告（即“垃圾搜索”）。
- 密码维护不足。安全策略可能要求每个用户每 30 天更改一次密码。安全审计显示密码已更改，但许多用户每月只是简单在“密码 A”和“密码 B”之间切换。（密码历史记录是密码审核工具中的一个常见功能。）
- 对用户权限的控制不足和放任权限，比如，用户访问权限已超过他们需要执行的工作权限。另一个例子是，单个用户的文件在网络上共享时，他们应该是私有的。尤其需要关注使用笔记本电脑的用户，那些通过 WIFI 来访问内部网连接的情况，无论是在家里或公共场合。
- 服务器级安全不足。具体审计领域包括：
  - 端口分配和安全
  - 保护数据
  - 保护用户账号（登录和其他敏感信息）
- 应用程序供应商安全更新的不足
- 入侵检测机制不足
- 出现安全漏洞时的响应计划不足

1.3.2 风险识别、评估和缓解

一旦审计确定了问题区域，就必须评估风险并制定改进计划。审计师的报告可能包括建议以及其他风险领域。从这一点开始，可以规划风险识别，评估和缓解活动。

风险识别是记录风险或风险领域的过程。在 IT 安全方面，风险与安全相关。风险评估是为已识别的风险分配价值的活动。重要的是要了解传统的 IT 风险评估模型不足以解决 IT 安全风险。任何安全风险评估模型或方法都应专门针对 IT 安全风险概况。

安全风险通常以风险敞口来衡量。风险暴露的计算方法是将潜在影响或损失乘以发生损失的可能性。例如，如果一个客户的账户信息受到损害，会产生什么影响？如果该客户有 1 亿美元的存款资产怎么办？

发生的可能性可以通过应用安全风险评估模型来确定，例如在 NIST 出版物 800-30，进行风险评估指南[NIST 800-30]中找到的模型。另一个执行安全风险评估的优秀指南是 OWASP 风险评级方法[OWASP2]。以下信息摘自[NIST 800-30]。

风险模型定义了要评估的风险因素以及这些因素之间的关系。风险因素是风险模型中使用的特征，作为确定风险评估风险水平的输入。风险因素也广泛用于风险沟通，以突出强烈影响特定情况，环境或背景下风险水平的因素。

典型的风险因素包括威胁，脆弱性，影响，可能性和诱发情况。风险因素可以分解为更详细的特征（例如，威胁分解为威胁源和威胁事件）。这些定义对于组织在进行风险评估之前进行记录非常重要，因为评估依赖于明确定义的威胁，漏洞，影响和其他风险因素属性来有效地确定风险。

威胁

威胁是指任何可能通过未经授权的访问，破坏，披露或修改信息和/或拒绝服务而通过信息系统对组织运营和资产，个人，其他组织或国家产生负面影响的情况或事件。

威胁事件是由威胁源引起的。威胁源的特征是：

- 针对利用漏洞的目的和方法；
- 或者可能意外利用漏洞的情况和方法。

通常，威胁源的类型包括：

- 敌对的网络或物理攻击
- 遗漏或委托人为错误
- 组织控制资源的结构故障（例如，硬件，软件，环境控制）
- 自然和人为灾害，事故和组织无法控制的失败。

已经开发了各种威胁源分类法。一些威胁来源的分类法将不利影响的类型用作组织原则。多个威胁源可以启动或导致相同的威胁事件，例如，拒绝服务攻击、恶意系统管理员的故意行为、管理错误、硬件故障或电源故障都可以使配置服务器宕机。

脆弱性和易感性条件

漏洞是信息系统，系统安全程序，内部控制或可由威胁源利用的实施的弱点。

大多数信息系统漏洞可能与未应用（有意或无意）或已应用但仍保留一些弱点的安全控制相关联。但是，随着组织任务/业务功能的发展，运营环境的变化，新技术的激增以及新的威胁的出现，随着时间的推移可能会出现紧急漏洞，这也很重要。在这种变化的背景下，现有的安全控制可能变得不充分，可能需要重新评估其有效性。随着时间的推移，安全控制可能会降低有效性的趋势，这强化了在整个软件生命周期中维护风险评估的必要性，以及持续监控程序对于获得对组织安全状况的持续态势感知的重要性。

漏洞不仅仅在信息系统中被识别。在更广泛的背景下查看信息系统，可以在组织治理结构中发现漏洞（例如，缺乏有效的风险管理战略和充分的风险框架，机构间通信不良，关于任务/业务职能的相对优先级的不一致决策，或未对准企业架构，以支持任务/业务活动）。

漏洞也可以在外部关系中找到（例如，对特定能源，供应链，信息技术和电信提供商的依赖），任务/业务流程（例如，定义不明确的流程或不具有风险意识的流程），以及企业/信息安全体系结构（例如，糟糕的体系结构决策导致组织信息系统缺乏多样性或弹性）。

影响

威胁事件的影响程度是指由于未经授权的信息披露、未经授权的信息修改、未经授权的信息销毁、信息或信息系统可用性的损失而可能造成的伤害程度。各种组织和非组织利益相关者都可能遭受这种伤害，包括：

- 机构负责人
- 任务和业务负责人
- 信息所有者/管理者
- 任务/业务流程所有者
- 信息系统所有者
- 个人/团体在公共或私人部门：依靠组织——在本质上就是任何一个在组织的运营、资产，或个人的既得利益，包括其他组织与组织合作，或一个国家。

以下信息应明确记录在一个组织中：

- 用于决定行为影响的过程
- 与影响决定相关的假设
- 获取影响信息的来源和方法
- 对影响决定的基本原理

组织可以明确定义已建立的优先级和价值如何指导识别高价值资产和对组织的利益相关者的潜在不利影响。如果未定义此类信息，则与识别威胁源目标和相关组织影响相关的优先级和价值通常可从战略规划和策略中得出。例如，安全分类级别显示了妥协不同类型信息所带来的组织影响。

可能性

发生的可能性解决了威胁事件将导致不利影响的概率（或可能性），无论可预期的伤害程度如何。这是一个加权风险因素，基于对给定威胁能够利用给定漏洞（或一组漏洞）的概率的分析。可能性风险因子将威胁事件将被发起的可能性的估计与对影响的可能性的估计（即，威胁事件将导致不利影响的可能性）相结合。

对于对抗性威胁，对发生可能性的评估通常基于：

- 对手意图
- 对手能力
- 对手定位

对于除对抗性威胁事件之外，使用历史证据，经验数据或其他因素估计发生的可能性。注意，将针对特定时间范围（例如，接下来的六个月，下一年或直到达到指定里程碑的时段）评估威胁事件将被发起或将要发生的可能性。

如果威胁事件几乎肯定会在（指定或隐含）时间范围内发起或发生，则风险评估可以考虑事件的估计频率。威胁发生的可能性也可以基于组织的状态（包括，例如，其核心任务/业务流程，企业架构，信息安全架构，信息系统以及这些系统运行的环境。易感条件和还应考虑部署的安全控制的存在和有效性，以防止未授权/不良行为，检测和限制损害，和/或维护或恢复任务/业务能力。

确定安全风险等级

可以将发生评估的可能性和影响评估结合起来计算风险的总体严重性。具体的评估分数可以用作完成风险矩阵的基础。在其他情况下，可以使用估计（低，中或高）。

风险矩阵的评分可以基于 0-9 的等级，其中数值由特定标准确定。例如，风险可能性标准可以通过以下方式评估数据隐私：

0 - <3（低）专用数据不存储在本地设备上，并在存储在安全介质上时加密。
3 - <6（中）私有数据可能驻留在笔记本电脑等设备上，但是是加密的。
6 - 9（高）尚不确切知道私有数据是否驻留在本地设备上。加密无法保证。

同样，风险影响标准可以根据具体标准在相同的 0-9 级评估。例如：

0 - <3（低）私人数据的妥协将影响少于 200 人。
3 - <6（中）私人数据的妥协将影响 200 至 1,000 人。
6 - 9（高）私人数据的妥协将影响超过 1,000 人。

然而，测试者得出可能性和影响估计值，估计值可以组合成风险项目的最终严重等级。如果有良好的业务影响信息，则应使用该信息而不是技术影响信息。如果没有关于业务的信息，那么技术影响是下一个最好的事情。

以下是风险矩阵的示例视图，可用于确定个别风险的严重程度。

整体风险严重程度				
风险影响	高	中	高	临界
	中	中	中	高
	低	低	低	中
		低	中	高
		风险的可能性		

在上面的示例矩阵中，如果可能性中等且影响很大，则总体严重性很高。

此外，风险评估报告应确定风险是否持续。持续的风险表明损失发生的可能性增加。

风险的严重程度决定了降低风险的相对重要性。风险严重程度越高，对响应的要求就越快。任何特定风险评估中提供的详细程度与风险评估的目的和支持后续可能性和影响确定所需的输入类型一致。

1.3.3 人员、流程和技术

组织的 IT 实践还有三个组成部分：人员，流程和技术。所有这些都会对安全产生影响。克里斯杰克逊在他的书“网络安全审计”[杰克逊，2010]中说，“所有安全事件，从闯入到丢失的客户记录，通常都可以追溯到可归因于人员，流程或技术的缺陷。“。

人员：人员可以包括最终用户，系统管理员，数据所有者和组织的经理。每个人都有不同程度的技能，态度和议程，这会影响安全性对他们的影响，以及它们如何影响安全控制的有效性。无论是否存在安全策略，程序和控制，如果人们不遵守它们，它们将无效。如果人们不遵守安全策略，则可能需要进行补救措施，例如需要进行安全意识培训或对违规行为进行处罚。组织结构和安全策略通常由组织内部和外部的人员驱动。

流程：流程定义如何交付 IT 服务，包括与安全相关的服务。在安全环境中，流程包括为保护有价值资产而实施的程序和标准。为了有效，必须定义流程，最新，一致，并遵循最佳安全实践。流程定义执行任务所涉及的角色和职责，控制，工具和特定步骤。

技术：技术包括自动化或支持业务的设施，设备，计算机硬件和软件。技术使人们能够比没有它的情况下手动执行更快，更少地完成重复性工作。事实上，如果没有合适的工具，密码实施等一些任务是不可能的。风险在于错误使用的技术可以帮助人们更快地犯错误。

这三个领域可以被认为是一个“铁三角”，它们共同构成了一个完整的 IT 解决方案。如果忽略这三个方面中的任何一个，整个 IT 交付和安全工作都会受到影响。

在评估安全控制时，审核员应从攻击者的角度审视系统，并预测如何利用人员，流程或技术来获取对有价值资产的未授权访问。组织中的管理层经常对他们认为安全的安全机制感到惊讶。确定特定安全防御是否有效且唯一的方法是从攻击者的角度测试系统。这通常被称为道德黑客攻击或渗透（笔）测试。

这是审计和测试之间的关系变得最直接的地方。审计确定了缺陷和测试重要领域。安全性测试是证明或证明安全控制实际到位并有效工作的手段。

示例场景：

一个国家的税务机构是安全审计的主体。其中一项审计结果是，犯罪分子可能会提交欺诈性的纳税申报表，并因纳税人的纳税而获得退税。此审核结果通过安全性测试得到确认，风险被评为“关键”。税务机构承认存在此类欺诈风险的可能性，但决定在下一年之前不对风险采取行动。

欺骗纳税人遵守所有规定的安全程序，可以向税务机构提出索赔，该税务机构知道税务申报过程中的缺陷。在这种情况下，税务机构将对欺诈负责。

2. 安全性测试目的，目标以及策略 -130 分钟

关键词

跨站脚本攻击，数据混淆，服务拒绝，信息保障，安全策略，安全性测试，安全漏洞，软件生命周期，测试策略

针对“安全性测试目的，目标以及策略”的学习目标

2.1 介绍

此处暂无学习目标

2.2 安全性测试的目的

AS-2.2.1 (K2) 理解一个组织为什么需要引入安全性测试，以及引入之后所带来的好处，比如减少风险以及带来更高层次的信心和信任。

2.3 组织环境

AS-2.3.1(K2) 理解项目现状，业务约束，软件开发生命周期以及其他因素是如何影响安全性测试组的任务。

2.4 安全性测试的目标

AS-2.4.1(K2) 解释为什么安全性测试的目标必须和组织的安全性策略以及其他测试目标相一致。

AS-2.4.2(K3) 对于既定的项目场景，描述基于功能，技术特性以及已知漏洞定义安全性测试目标的能力。

AS-2.4.3(K2) 理解信息保障和安全性测试之间的关系。

2.5 安全性测试目标的范围及规模

AS-2.5.1(K3) 对于既定的项目，描述定义安全性测试目标以及加强敏感数据与物理资产完整性的需求之间的联系的能力。

2.6 安全性测试的方法

AS-2.6.1(K4) 分析一个既定场景以及决定选择最合适的安全性测试方法。

AS-2.6.2 (K4) 分析一个安全性测试方法失败的场景，并确定失败的原因。

AS-2.6.3 (K3)对于既定场景，描述能够确定负责人以及将安全性测试的好处介绍给每个负责小组的能力。

2.7 改进安全性测试的实践

AS-2.7.1 (K4) 分析 KPI 以便确定哪些安全性测试需要改进，哪些不需要。

2.1 介绍

在应用专门的安全性测试技术之前，了解更广泛的安全性测试环境及其在特定组织中的角色非常重要。这种理解回答了以下问题：

- 为什么需要进行安全性测试？
- 安全性测试的目的是什么？
- 安全性测试如何适合组织？

安全性测试与两个重要领域的其他形式的功能测试不同[ISTQB®_ATTA_SYL]：

1. 选择测试输入数据的标准技术可能会遗漏重要的安全问题
2. 安全缺陷的症状与其他类型的功能测试的症状非常不同

安全性测试通过尝试破坏系统的安全策略来评估系统对威胁的脆弱性。以下是潜在威胁列表，应在安全性测试期间探讨[ISTQB®_ATTA_SYL]：

- 未经授权复制应用程序或数据
- 未经授权的访问控制（例如，执行用户没有权限的任务的能力）。用户权限，访问权限和权限是此测试的重点。该信息应该在系统规范中提供。
- 在执行预期功能时出现意外副作用的软件。例如，正确播放音频但通过将文件写入未加密的临时存储器而这样做的媒体播放器表现出副作用，这可能被软件盗版者利用。
- 插入到可由后续用户执行的网页中的代码（跨站点脚本或 XSS）。此代码可能是恶意的。
- 缓冲区溢出（缓冲区溢出）可能是由于将数据字符串输入到用户界面输入字段中而导致的，该字段长度超出代码可以正确处理的范围。缓冲区溢出漏洞表示运行恶意代码指令的机会。
- 拒绝服务，阻止用户与应用程序交互（例如，通过使用“讨厌”请求重载 Web 服务器）
- 第三方拦截，模仿和/或改变和随后转发通信（例如，信用卡交易），使得用户仍然不知道该第三方的存在（“中间人”攻击）
- 破坏用于保护敏感数据的加密代码
- 逻辑炸弹（有时称为复活节彩蛋），可能被恶意插入代码中，并且仅在某些条件下（例如，在特定日期）激活。当逻辑炸弹激活时，它们可能会执行恶意行为，例如删除文件或格式化磁盘。

安全性测试必须与所有其他开发和测试活动集成。这需要考虑组织的独特需求，任何现有的安全策略，当前的安全性测试技能集以及任何现有的测试策略。

2.2 安全性测试目的

与一般的软件测试一样，安全性测试无法保证系统或组织免受攻击。但是，安全性测试可以帮助识别风险并评估现有安全防御的有效性。还有其他活动可以补充安全性测试，例如审计和安全实践审查。

安全性测试还表明，在保护数字资产方面已经进行了尽职调查。如果发生安全漏洞，可能会导致法律诉讼。如果一家公司能够证明它采取了合理的措施来保护数字资产，例如测试漏洞，那么法院可能会提出辩护。安全性测试还可以向客户和客户保证组织采取适当措施来保护敏感信息。

2.3 组织背景

安全性通常是与其他类型的测试一起执行的一种功能测试。只有一定时间可用于测试，测试经理必须决定可以执行多少测试，包括安全性测试。安全性测试被视为专家角色并因此外包给专门从事安全性测试的组织并不罕见。安全性测试的范围最终由基于安全性的业务或组织风险驱动。当组织中的安全风险很多时，需要进行更广泛的安全性测试。

与软件测试一样，信息安全也是一种生命周期活动。必须在需求中定义安全需求，在设计中表示并在代码中实现。然后，安全性测试可以验证并验证安全实现的正确性和有效性。安全性无法有效地修补到代码中或测试到代码中。只有在使用安全编码和设计技术将安全性内置到软件中时，才能确保软件安全。

有限的时间，资源和范围以及风险级别，安全性测试技能集和生命周期方法的现实极大地影响了组织中安全性测试团队的成功。

2.4 安全性测试目标

2.4.1 如何与安全性测试目标保持一致

一旦组织的安全策略被高层认可后，安全性测试的策略就可以确定了。安全性测试策略中描述的测试目标与整个组织的安全策略相一致是非常重要的，否则的话，未授权的安全性测试就会被执行，或者授权的安全性测试未能达到预期的目的。

2.4.2 安全性测试目标的设定

我们可以把安全性测试目标比作其他功能测试目标，只是更侧重于安全性。针对每一个系统或应用的安全性功能都会有至少一个安全性测试目标。

安全性测试目标必须基于技术的特性（比方说，web，mobile，cloud，LAN）以及已知的漏洞，包括应用程序特有的及普遍的漏洞。比方说：

- 验证密码的认证符合正确的密码长度规则
- 验证所有数据输入处都能阻止 SQL 注入攻击
- 验证客户数据被正确地加密

2.4.3 信息保障与安全性测试的区别

信息保障（IA）定义为“通过确保信息和信息系统的可用性，完整性，身份验证，机密性和不可否认性来保护和保护信息和信息系统的措施。这些措施包括通过结合保护，检测和反应能力来恢复信息系统。“[NISTIR 7298]

安全性测试是“用于确定系统的安全功能按设计实现并且适用于建议的应用程序环境的过程。”[MDA1]

在比较信息保证（IA）和安全性测试这两个术语时，IA 是一个更广泛，更具包容性的术语。这种关系类似于质量保证（QA）和软件测试之间的关系。

2.5 安全性测试目标的范围及覆盖率

一般来说，敏感数据与物理资产的完整性需求越高，则安全性测试目标的覆盖率需求就越大。安全性测试的目标必须定义安全性测试的范围，如果范围过小，那么就很难有足够的自信来证明该系统足够安全；但如果范围过大，那么相关的资源很有可能在测试完成之前就被损耗了。

安全性测试目标必须描述对于验证敏感数据和物理资产的保护程度，通过安全性测试需要达到什么样的期望效果，当然该目标应该细化到特定的资产，防护的评估，风险以及安全性漏洞的识别。

2.6 安全性测试的方法

安全性测试策略定义出了一个组织在安全性测试方面的总体方向，之后就需要定义安全性测试的方法以便实现该策略。

2.6.1 安全性方法的分析

每个组织都有其独特的业务和使命，因此也就需要不同的安全性测试策略及方法来识别并减少其安全性风险。不过也有一些通用的安全性考虑适用于绝大多数组织。

我们需要在项目的水平来定义安全性测试方法，并且应该要与整个组织的测试策略相一致。每一个项目的安全性测试方法应该都各不相同，是一系列为了达到该项目的安全性测试目标而使用的技术，工具和技能的集合。

我们需要考虑以下几个因素来分析决定如何定义安全性测试的方法：

- 系统及应用程序的来源
- 之前所执行过的安全性测试
- 安全性策略
- 安全性测试的策略
- 该组织已做过的安全性风险评估
- 现有的技术环境（比方说，软件类型和版本，框架，开发语言，操作系统等）

- 测试团队所掌握的安全性测试技能
- 常见的安全性风险
- 测试组织架构
- 项目团队的组织架构
- 测试团队对于不同安全性测试工具的使用经验
- 限制（比方说，有限的资源，有限的时间，对环境没有足够的使用权限等）
- 假设（一些需要在执行安全性测试之前的假设）

不同的技术背景和不同的应用类型（比如，client/server，web，大型机）通常都需要不同的安全性测试方法和策略。比方说，软件开发的时候需要通过代码审查来检查代码内的安全性漏洞，而软件测试则需要模糊处理测试数据。基于 web 的应用程序与大型机系统有不同的漏洞，因此也就需要不同的安全性测试。

当然也有一些漏洞是普遍存在于多种技术背景的，比方说，缓存区溢出就会发生在各种应用类型：client-server，web 以及移动端 app，因为这主要取决于内存管理是怎么处理的。当然这些漏洞也会导致相同的结果：一些黑客可以轻而易举的利用这些漏洞来侵入系统并且执行一些非法操作。

不同的数据保护技术也存在于不同的技术背景中，Web 端和移动端应用程序的数据加密就跟大型机不同。他们的加密算法有可能是一致的，但是不同之处在于如果是 web 端或者移动段应用程序的话，那么相应的数据也就需要在通过 Internet 传输的阶段被保护起来。不过不管是何种技术，任何敏感数据都必须要用过加密格式来保存。曾经就发生过由于一些敏感的大型机数据未经加密就通过磁带传输给对方而导致的事故。《计算机周刊》曾报道过：Cattles Group（一家专门负责个人信贷记录恢复的机构）承认他们丢失了两份包含大约 140 万客户信息的备份磁带。

2.6.2 分析安全性测试方法的失败案例

我们必须要理解有不同等级的安全性失效。如果仅仅是一个安全性漏洞未能被发现和解决，那是不能说明整个安全性测试的方法是失败的。因为目前有很多可能存在的安全性漏洞，并且每天还会有新的漏洞被发现。不过还有其他安全性测试方法不能有效发现安全性风险的情况，这些会导致敏感数据及其他数据资产被破坏。

分析根本原因可以帮助确定一种安全性测试方法失效的原因。以下是有可能的一些因素：

- 缺乏有足够执行力的领导团队的支持去建立安全性测试
- 缺乏足够的用以实现安全性测试策略的资源（比如，缺少资金，缺少时间，缺少资源）
- 缺乏有效地执行安全性测试方法的能力（比如，缺乏必要的执行相关任务的技能）
- 缺乏组织对于安全性测试的理解和支持
- 缺乏利益相关者对于安全性测试的理解和支持
- 缺乏对安全性风险的认识
- 缺乏组织的安全性策略与总体测试方法的一致性
- 缺乏组织的安全性测试策略与总体测试方法的一致性
- 缺乏对整体系统目标的理解

- 缺乏整体系统的技术细节（导致一些错误的假设）
- 缺乏有效的安全性测试工具
- 缺乏安全性测试的技能

2.6.3 利益相关者的认同

为了使安全性测试更有效，这就必须让管理层意识到这是与业务紧密相连的。因此我们必须非常清楚的定义出安全性缺失所带来的风险以及有效的安全性测试方法所带来的好处。

不同的利益相关者将会看到不同的安全性测试所带来的好处：

- 执行管理层将业务保护视为一种福利
- 高级管理层可能会进行尽职调查
- 商业客户将会看到保护的措施以免受到欺骗
- 合规人员（针对公司内部的安全策略）可以确保组织在法律义务方面符合要求
- 监管人员（针对外部安全性法律）可能会看到遵从安全性规则所带来的好处
- 隐私人员可能会看到私人数据保密的好处，并且在保护数字资产方面已经表现出尽职调查

2.7 改进安全性测试的实践

为了改进安全性测试实践，首先需要对现有实践进行评估。应该有一种客观的方法来评估安全性测试实践。这些基于安全性测试目标的关键指标，从中可以确定关键战略要素的成功程度。

对这些实践的评估如下：

- 通过短期和长期两方面相结合
- 考虑到流程及组织架构
- 考虑到人手，工具，系统以及技术

这些关键指标至少包括以下几条：

- 安全性风险的覆盖率
- 安全性策略和实践的覆盖率
- 安全性需求的覆盖率
- 过去的安全性测试的有效性，这个可以根据安全性漏洞于何时何处被发现所决定（这些漏洞包括了发布前和发布后）。

3.安全性测试流程 - 140 分钟

关键词

合法账户收集，密码破解，社会工程，测试方法，测试计划，测试流程

安全性测试流程章节学习目标

3.1 安全性测试流程定义

AS-3.1.1 (K3) 对于某个项目，具有定义有效的安全性测试流程及其要素的能力。

3.2 安全性测试计划

AS-3.2.1 (K4) 分析安全性测试计划并对其优缺点提供反馈。

3.3 安全性测试设计

AS-3.3.1 (K3) 对于某个项目，基于特定的安全性测试方法以及已识别的功能和结构的安全风险，实现概念（抽象）安全性测试模型

AS-3.3.2 (K3) 为验证安全策略和规程创建安全性测试用例

3.4 安全性测试执行

AS-3.4.1 (K2) 理解一个有效安全性测试环境的要素和特性

AS-3.4.2 (K2) 理解在执行任何安全性测试之前计划并获得批准的重要性

3.5 安全性测试评估

AS-3.5.1 (K4) 分析安全性测试结果并确定以下几点

- 安全漏洞的性质
- 安全漏洞的程度和范围
- 安全漏洞的潜在影响
- 建议的补救方案
- 报告测试结果最合适的方法

3.6 安全性测试维护

AS-3.6.1 (K2) 鉴于技术和威胁不断演变的性质，理解维护安全性测试流程的重要性

3.1 安全性测试过程定义

像常规的软件测试一样，安全性测试也是软件生命周期中不可或缺的一项活动，未能在整个项目中进行有效的安全性测试并实施安全防御可能导致严重的安全缺陷发生，这些缺陷可能永远都无法完全解决。安全性测试流程必须与开发流程对应，以便在需要时执行适当的测试活动。

由于组织的性质，技术环境，软件开发流程以及业务风险不同，每个组织的安全性测试需求和风险将是独一无二的。因此必须在这些因素的背景中定义安全性测试流程。

3.1.1 ISTQB®安全性测试流程

表 3.1 说明了在 ISTQB®基础和高级大纲中表述的常规测试流程和 ISTQB®安全性测试流程的关系，并在流程的每个步骤举例说明对应的安全性测试任务。

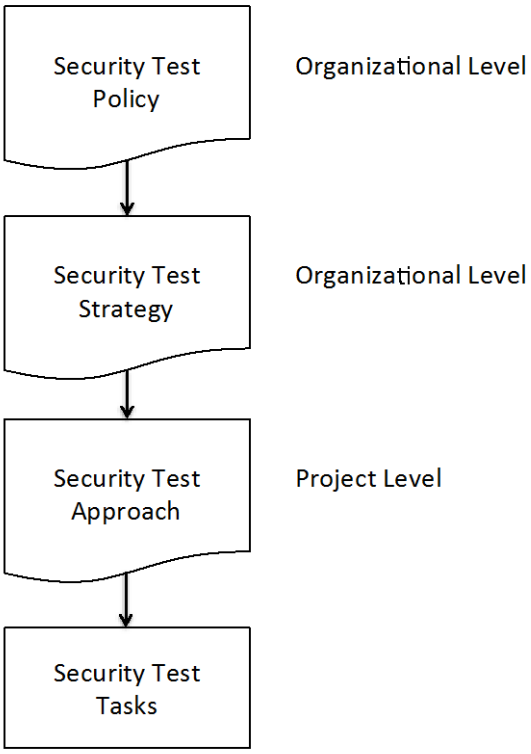
表 3.1 – ISTQB®安全性测试流程

ISTQB®测试流程	**ISTQB®安全性测试流程**	**安全性测试任务示例**
测试计划与控制	安全性测试计划与控制 - 目标是定义与安全风险相对应的合理测试范围	• 定义安全性测试目标 • 定义安全性测试范围 • 确定安全性测试所需资源 • 定义安全性测试所需成本和时间表 • 定义安全性测试指标，准入准出标准 • 监控安全性测试进度和结果 • 收集各种安全性测试活动的信息，根据需要采取措施。
测试分析与设计	安全性测试分析与设计 - 目标是基于安全评估，审计和已知漏洞的标准来源，了解具体的安全威胁和风险。	• 审阅诸如安全风险评估，安全需求和安全策略等作为安全性测试基础的条目。 • 基于如下内容定义安全性测试条件： **1.** 测试目标 **2.** 安全风险 **3.** 和已知漏洞 4. 已实施的系统及其数据的安全保护措施 **5.** 安全性测试范围

ISTQB®测试流程	ISTQB®安全性测试流程	安全性测试任务示例
		6. 安全性测试工具的适用性
测试实施与执行	安全性测试实施与执行 - 目标是将概念（抽象）测试转换为可以手动或使用工具执行的测试。并且要用多种视角（内部用户，外部用户，恶意用户等）来执行这些测试。	• 创建测试用例，测试场景，测试脚本或其他测试规格 • 根据定义的安全性测试规格执行功能安全性测试 • 基于测试人员所掌握的知识和直觉能力执行功能性安全性测试和渗透测试 • 基于系统的设计架构执行安全性测试 • 为安全性测试准备测试环境
评估测试退出准测并提交报告	评估安全性测试并提交报告 – 通常与每个单独的测试执行同步进行，评估每个测试并尽快报告新的威胁。	• 根据测试结果确定具体的安全漏洞。 • 根据安全性测试结果评估安全风险级别 • 向管理层和其他授权方报告临时和最终的安全性测试结果
测试总结	测试结束 – 目标是总结已完成的安全性测试活动，以便能够定期维护和执行测试来支持新的安全性测试需求和新的安全威胁。此外要以安全的方式存储安全性测试工具和结果，并在以后测试需要时使用。	• 确保所有计划的安全性测试都已执行 • 确定安全性测试可交付成果（报告）是否已交付 • 将测试结果，测试数据和其他敏感信息归档到安全位置 • 分析安全性测试结果以改进系统和应用程序开发的安全性

重点要理解 ISTQB®安全性测试流程本质上不一定是顺序的。安全性测试流程应与组织的软件生命周期过程保持一致。本章节所描述的安全性测试流程的一个重要概念是安全性测试活动与项目生命周期中其他活动和测试并行执行。

此外，表 3.1 中所示的安全性测试任务仅作为示例，而不是安全性测试任务规定的要求。对于一个组织，确切的安全性测试任务取决于组织采用的安全性测试策略和方法，如下图 3.1 所示。



安全性测试方针 – 组织级

安全性测试策略 – 组织级

安全性测试方法 – 项目级

安全性测试任务

图 3.1 -安全性测试计划的层级

3.1.2 安全性测试流程匹配特定的软件生命周期模型

以下类型的生命周期过程都需要关注安全性测试的问题，将安全性测试与生命周期模型相匹配是至关重要的。

瀑布式

在这类项目中，安全性测试人员应该注意以下几点：

- 安全需求和风险在项目早期定义，并记录在软件需求规格中。

- 安全需求可能在项目期间发生变化，但可能不会反映在更新的软件需求中。安全性测试可能看起来非常具体和完整，但在项目后期可能并不会覆盖完整和实时的安全需求。
- 虽然安全性测试可以随时执行，但对于瀑布模型来说这些测试通常在项目后期执行。在瀑布模型项目快结束时，可能很难解决安全性测试所发现的问题。

迭代式/增量式

增量式项目会频繁发布较小的应用程序版本。敏捷开发就是这种模型的一个例子。在这些项目中，安全性测试人员应该注意以下几点：

- 在整个项目中（通常在完成一个迭代或冲刺的背景下）暴露出的安全需求和风险，可以在需求规格，用户故事，模型，验收标准和原型这些文档或交付物中来定义。
- 安全需求和风险可能在项目期间发生变化，应在它们被识别出的迭代中加以解决。
- 安全性测试可以在整个项目中持续进行。
- 对于安全风险而言，可能无法在一个短暂的发布周期内完全缓解并对其进行验收测试，这取决于安全风险的性质。

商业成品软件

这些项目本质上通常是黑盒的，产品可能会进行客户化定制也可能不会。它们通常会包含某种程度上的一些安全漏洞，以至需要频繁的安全更新和修补程序。因为无法查看产品的源代码，所以不可能进行结构化的分析和测试。

开源软件

开源软件可以理解为商业成品软件的另一个版本，但有一个重要的区别就是源代码所有人都可以看到。这些产品也会存在安全漏洞，因此，保持最新的安全补丁是至关重要的。一旦安全漏洞被公开，该软件的特定版本（和更早版本）的用户就有受到攻击的风险。

示例– 瀑布式生命周期中的安全性测试流程

重点注意，安全性测试不必局限于项目中的一个阶段或活动。特别重要的是避免直到项目验收阶段也没有执行过安全性测试（和其他测试）的情况。在项目结束时，处理任何发现的缺陷是特别昂贵并有风险的。以下指出了如何在瀑布式生命周期的每个阶段完成适当的安全性测试任务：

- **需求阶段** - 定义和评估安全需求并作为陈述组织整体需求工作的一部分。这个阶段可以编写安全需求相关的用例。与此同时安全性测试方法需要被详细阐述。

- **分析和设计** – 通常，业务分析人员将检查初始需求声明并对其进行优化以解决分歧。随后，系统分析员或架构师将分析要求，提出满足用户需求的最佳解决方案。在这种情况下，安全将是功能需求，非功能需求，以及其他诸如可用性和有效性需求的一种。在这个阶段，安全性测试设计人员可以从结构和功能安全的角度了解架构和需要测试的内容。主要的安全性测试目标也应同时被定义。

- **详细设计**– 对用户界面和数据库进行设计。对功能规则进行改进。同时安全性测试设计变得更加详细。可能需要执行初始的基于模型的安全性测试。

- **编码/实现** – 将设计规格实现为代码。这个阶段也是测试应用程序结构的第一次好的时机，包括测试如缓冲区溢出和字段编辑可能引起的 SQL 注入等安全漏洞。从安全的角度进行静态分析和代

码审查在此阶段是非常重要的。组件测试也是验证代码是否满足设计需求的关键活动。组件之间的集成测试也可以开始，因为组件间用于集成的接口适合于小范围的装配测试。

- **系统测试** – 测试系统及其子系统。系统测试包括软件，硬件，数据，过程以及人们如何与系统交互。通常这些测试本质上是事务性的以便测试业务流程。系统测试的基础可以是需求，设计模型，用例和表达系统远景的任何其他规范。此外，可能需要执行系统集成测试以测试各种子系统如何通信和交换数据。因为涉及硬件和数据交换，此阶段的安全性测试涉及更为广泛。同时事务安全性也可以被测试，包括认证，数据存储，防火墙实现以及程序安全控制。

- **用户验收测试** – 验证系统支持真实业务流程并可跨越多个组织中的多个系统。这个阶段的目标不是找到更多缺陷，而是验证系统在现实条件下满足用户需求。用户验收测试包括确保安全需求已得到完整的实施和有效的满足。到了这个阶段，安全性测试应该很大部分已经被执行，但仍然有机会测试业务流程方面的安全场景。

- **部署** – 将完成的以及被测试过的系统部署给用户使用。部署过程有很多种方式，例如对某个组别用户的试验性部署或对所有用户的大规模部署。另一种方式是并行部署，其中旧系统和新系统在短期内同时运行。大多情况下对系统实施直接切换取决于部署给所有用户的风险以及对验收测试的信心。在系统部署期间需要考虑安全性的问题，因为所有系统组件的部署必须以不引入新的安全漏洞为前提。如果目标环境中的安全配置不正确，则可能会出现这种情况。例如，数据库访问权限在生产环境中配置不当。

- **维护** - 在部署后发现新需求或发现缺陷时，需要进行维护。此时测试会采用不同的维度，重点在于测试变更和执行回归测试。安全性测试也需要被执行，以确保在更改期间不会引入新的漏洞。维护过程的一部分是确保实时更新防火墙和其他安全防护措施。持续系统监控能够检测到需要立即解决的可疑活动。

示例 - 迭代/增量生命周期中的安全性测试过程

在过去 20 年中引入了各种方法来以较小的增量或迭代来定义软件的构建。在此示例中，软件每四周发布一个版本。开发（和测试）的基础是有明确验收标准的用户故事.

选择要构建和交付的功能是基于需要优先处理的待办事项。所选的应是那些能提供最大价值的、可实现的功能，并体现在冲刺的时间表中。安全性测试人员与产品经理协作以定义出合适和正确的安全需求。

在本示例中，为第一次迭代选择四个主要的安全功能，因为它们将会是开发许多其他功能的基础。这些功能是：

- 用户登录
- 启用 SSL
- 密码重置
- 三次登录失败后锁定账户

其中每一个功能都是以用户故事来编写的，并完善成更为详细的需求及相应的验收标准。

从安全性测试的角度来看，安全性测试人员与开发人员协作，以确保正确的策略和规则能够反映在代码中。安全性测试人员还将与开发人员在同一场所内工作，以便及时对开发的功能进行测试。

在该示例中，初始版本可以仅是登录页面和用于登录的相关功能，例如重置密码和锁定控制。在下一次迭代中，将根据利益相关者的优先级来开发其他功能。在每次迭代中，安全性测试人员进行测试，以确保相关安全控制正常工作，并且没有引入新的安全漏洞。迭代将会继续直到待办任务完成。

在这两个示例（迭代/增量和瀑布）中，安全性测试流程的步骤可以看作是确保应用程序安全不可或缺的任务。

3.2 安全性测试计划

3.2.1 安全性测试计划目标

安全性测试一般应关注两个方面：

- 验证设计的安全防护措施是否已被实施并按设计功能运行
- 验证在应用程序开发期间没有引入安全漏洞

如本大纲前文所述，所有要实施的安全防护措施应基于风险分析，这为规划项目安全性测试提供了一个起点。

在创建架构，设计和编码活动期间，可以使用质量保证和最佳实践的方式来避免许多意外发生的漏洞。测试是否引入了漏洞，可以从评估开发团队过往的实践开始。根据需要选择和采用额外的安全性测试。

3.2.2 安全性测试计划关键要素

安全性测试计划的关键要素如下所示，这些都可以通过采用项目问答的方式来确定。

- 确定安全性测试的范围
  - 什么在测试范围内以及什么不在测试范围呢？
  - 以指定的项目资源，安全风险和限定时间来说可实现什么？
- 确定谁负责执行安全性测试
  - 组织中是否有人具有适当的安全性测试技能？
  - 组织是否对外部的安全性测试有信心?
  - 对于商业软件和供应商开发的软件，哪些安全性测试由供应商负责，哪些由客户负责？
  - 安全性测试人员是否需要特定安全性测试工具的培训?
- 根据项目其他测试的调度要求，为安全性测试分配合适的时间表
  - 在进行其他测试之前，需要实施和测试哪些与安全相关的功能？（例如，访问权限和登录）
  - 安全功能何时可以被测试
  - 根据计划的资源和范围，执行安全性测试需要多少时间？
- 定义要执行的任务和每个任务所需的时间
  - 基于计划的资源和范围设计适当的安全性测试需要多少时间？
  - 评估并汇报安全性测试结果需要多少时间？
  - 执行与安全相关的回归测试需要多少时间？
  - 建立安全性测试环境需要多少时间？
- 定义安全性测试环境

- 环境的体量如何？（平台，技术，规模，所在位置）
- 是一个新的环境吗
- 需要在环境中安装哪些安全性测试工具和其他辅助测试工具？

- 为安全性测试活动取得批准和授权
  - 由谁来批准和授权安全性测试？
  - 授权需要的时间段？
  - 资金预算是否足够？

如同任何可交付项目一样，安全性测试计划应当被评审，以评估其完整性和正确性。由于安全性测试通常是技术性的，技术评审会议可能是最适当的。然而，演练和监控也是较合适的方式。

一个标准的检查列表可以辅助形成评审会议需要涵盖的内容。像任何其他评审一样，反馈应该是建设性的，而不是针对安全性测试计划的编写者。讨论测试计划的评审小组应由所有那些会受安全方面影响的领域专家组成。评审小组成员可能不一定是安全性测试人员或拥有安全专业知识的人员。例如，业务部门的经理可能知晓需要记录在安全性测试计划中的关于安全风险的信息。 IT审计员和安全管理员对安全性测试计划评审尤其有帮助，因为他们了解安全策略和安全规格。

3.3 安全性测试设计

有几种途径来启动安全性测试设计。例如：
- 基于已完成的风险分析
- 基于可用的威胁模型
- 基于安全风险的特定来源分类（参见[ISTQB®_ATTA_SYL]）

任何这些都可以形成安全性测试设计可行的基础。

根据项目的类型，重点是要确保在每个适用的开发阶段都执行安全性测试。

3.3.1 安全性测试设计

详细的安全性测试基于安全风险，安全性测试策略和其他来源如威胁模型。安全性测试本质上也可以被视为功能性和结构性的测试。例如，对电子商务网站进行安全性测试的情况下，功能性安全风险可以是SQL注入，合法账户收集和密码破解。结构性安全风险的一个例子是在缓冲区溢出的情况下允许攻击者通过内存故障获取权限。

以下是详细的安全性测试基本属性：
- 用已识别的安全风险和威胁模型确定优先级
- 能够追踪到已定义的安全需求
- 基于预期的受众来定义（开发人员，功能测试人员，安全性测试人员）
- 基于安全缺陷的属性来定义
- 可使用情况下设计为自动化测试

安全性测试设计的基本流程如下：
1. 安全性测试方法（项目级）
2. 安全性测试风险，威胁模型和需求（项目级）
3. 安全性测试设计技术（基于风险，需求和应用程序）

4. 安全性测试用例和场景

在本章节余下部分，提供了常见的安全风险和漏洞以及相关的安全性测试设计技术。由于新的安全风险和漏洞迅速不断的发生，因此建议安全性测试策划人员保持最新的安全标准和威胁列表，如第 9 章所述。

一个关键的原则是安全性测试设计过程应该能够基于任何已识别的安全风险，需求以及威胁来创建和实施测试。

功能安全控制（如事务控制）
这些测试旨在验证和确证控制措施是否就位，运行是否正常以及如何有效检测和防止未经授权的操作。

示例：在没有财务经理批准的情况下，银行出纳员不得授权超出某金额的现金提款。

功能访问控制（如登录，密码，令牌）
这些测试也许是大多数人在安全性测试方面能够立刻想到的。它们包括：

- 正确应用用户名和密码策略
- 风险程度适当的访问控制级别
- 访问控制能够抵御密码破解软件的攻击

示例：合法账户收集是识别有效用户名的过程。一旦猜到了真实有效的用户名，密码就是获得系统访问所需的剩余部分。常见的测试是验证在输入正确用户名和错误密码时，错误消息不提示哪个输入错误。

结构访问控制（如用户访问权限，加密级别，认证）
对于这些控制措施的测试是基于如何为数据访问，功能访问和隐私级别建立用户权限。结构访问控制通常由系统管理员，安全管理员或数据库管理员实施。在某些情况下，访问权限是应用程序中的配置选项。在另外的情况下，访问权限在系统基础架构级别被实施。

结构访问控制的测试为每个安全访问级别创建测试用户，并验证每个访问级别不具有该级别限制以外的访问权限。例如，为最低访问级别，经理访问级别和管理员访问级别创建用户，执行相关测试以确保具有最低访问权限的用户不能进行管理员权限的访问活动。

安全编码实践
主要是静态测试方法，以确定软件和系统开发人员在创建应用程序时是否遵循已建立的安全方法。

一个关键的原则是很多安全攻击是通过利用软件缺陷来实现的，从而导致系统以无序的方式运行。

一个简要的安全编码实践列表包括：

- 用经过验证的会话管理控制和算法来创建随机会话标识符。
- 授权决策仅由提供授权的组织所控制的可信系统对象决定（例如，授权应在服务器端运行）。
- 安全相关的信息不应出现在错误消息中。此信息可能包括系统详细信息，会话标识符和帐户信息。
- 应用程序错误应在应用程序中处理，而不是依赖于服务器配置。
- HTTP GET 请求不应包含敏感信息。
- 错误处理程序不应显示栈跟踪信息或其他调试信息。
- 所有数据输入验证失败应当被记录。
- 任何临时存储在服务器上的敏感信息应当被保护（如使用加密）。此敏感信息应在不再需要时清除。

- 应用程序不应直接向操作系统发送指令。应该使用系统内置的 API 来执行操作系统任务。
- 密码，连接字符串或其他敏感信息不应以明文形式存储在客户端计算机上（如在 Cookie 中）。并且应禁止将此类信息嵌入到非安全格式（如 Adobe Flash，编译代码和 MS Viewstate）中
- 所有敏感信息都应以加密方式传输。安全传输层协议（TLS）是一种在使用 HTTP 连接时保护传输数据的方式。对于非 HTTP 连接，应使用加密来传输敏感信息。

- 用户输入的数据不应直接传递到任何动态“include”函数中
- 所有用户输入的数据在被应用程序使用之前应进行适当的净化和校验。
- 变量应在支持类型检查的语言中强制定义。也就是说，变量应该具有定义的输入类型。例如，数字字段不应接受字母字符。此限制将以变量类型声明来定义以及在数据库中定义。对于
- 不要对常见任务使用新的非托管代码，而应使用那些纳入配置管理的已测试的，可信的和已核准的代码。
- 以最小权限运行服务（从不在 root 下运行），并且每个服务都应在操作系统上有自己的用户账户。

安全编码实践列表可以在 OWASP 安全编码实践快速参考指南[OWASP1]和十大安全编码实践[CERT1]中找到。此外，SANS 在[SANS1]上编写了前 25 个最危险软件弱点的列表。

通过执行动态测试可以确定开发人员是否遵循了诸如数据验证和错误消息传递的惯例。此外，最常见的安全漏洞之一即内存缓冲区溢出，可以用动态内存测试工具来检查。

操作系统访问
一旦获得操作系统的访问权限，攻击者就可以得到数据和网络访问的控制并植入恶意软件。执行相关测试来鉴别操作系统是否能够被植入 rootkits 和其他恶意代码。

编程语言漏洞（如 Java）
应用程序安全供应商 WhiteHat Security 的安全研究人员提出：“全面来看，在涉及安全漏洞时，编程语言之间没有显著差异“。[WhiteHat Security，2014] 2014 年 4 月，WhiteHatSecurity 发布了一个网站安全统计报告，该报告基于使用专有扫描器对 3 万个客户网站进行的漏洞评估，结果显示..NET, Java, PHP, ASP, ColdFusion 和 Perl 在编程语言相对安全性方面差异很小。这六种语言占有相对类似的漏洞平均数，并且诸如 SQL 注入和跨站脚本之类的漏洞仍然普遍存在。 [WhiteHat Security，2014]重点要认识到，安全代码像非安全代码一样可以在许多编程语言中实现，关键因素是无论使用哪种语言，应用程序是以何种方式编码的（实现的）。

软件工程研究所的 CERT 部门提供了解决语言特定安全问题的发文[CERT2]和工具[CERT3]。此外，漏洞记录数据库[CERT4]提供了有关软件漏洞的及时信息。漏洞记录包括摘要，技术细节，补救信息和受影响的厂商列表。

平台漏洞（例如，Windows，Linux，MacOS，iOS，Android）
每个计算平台都有自己特定的安全漏洞。安全性测试人员要关注是确保对那些运行受影响平台的所有设备及时进行安全更新。

外部威胁
谈到网络攻击时大多数人会想到这是外部安全威胁。那些通过利用应用程序和语言漏洞的外部威胁是能够被发现，测试和防止的。

拒绝服务（DoS）是另一种类型的外部威胁。一般来说，这些攻击基于使系统或应用程序资源超载，从而不能被合法用户正常访问。DoS 攻击的目标可能是网络的带宽，系统或应用程序连接，或者是特定的服务或功能。

分布式拒绝服务（DDoS）是一种利用其他计算机资源间接启动的 DoS 攻击。可能的技术是放大攻击或僵尸网络，僵尸网络是受攻击者控制的大量早期受损计算机。攻击者可以通过简单地发送病毒或木马来获取控制权。受感染的计算机被用作代理向攻击者指定的某个受害者（网络）发动攻击。

当使用放大或反射攻击时，攻击者使用特定协议（如 DNS 或 NTP）中的漏洞（甚至是正常的功能）。攻击者向 IP 广播地址（多个主机）发送大量源地址伪造为受害者地址的数据包，这导致广播服务向受害者地址响应该请求，并将原始应答数据包大小与主机数量相乘。当攻击者每秒发送几次这种类型的请求时，受害者突然间就会需要面对大量需要处理的响应。

示例：攻击者 A 假冒受害人 C，通常通过伪造 IP 地址的方式向系统 B 发送请求，需要其提供所有已知 DNS 记录的完整列表。随后系统 B 将完整列表发送给受害者 C，从而形成通过扩大数据量的方式向受害者 C 发起泛滥攻击。

DoS 攻击的另一种形式是资源耗尽攻击。这些类型的攻击通过消耗提供功能所需的系统资源（CPU，内存，磁盘存储等）使得正常功能不可用。

示例：SSL 协议中的一个功能是在客户端或服务器怀疑现有会话受损时选择为其生成新的密钥。生成密钥是一个非常消耗资源的过程。当攻击者每秒发送几次生成新密钥的请求时，配置不当或未受保护的系统的处境可能是仅生成新密钥而没有任何资源执行其他操作，从而导致服务停止。

最后谈谈所谓的逻辑 DoS 攻击，攻击者通过功能滥用以阻碍其他用户访问系统。

示例：应用程序使用可预测的用户名，并在三次登录失败后永久锁定账户。攻击者可以猜测用户名并锁定系统中的许多账户，导致那些用户无法访问系统（间接对帮助台发起了拒绝服务攻击）。

DDoS 有四个层级的测试。

1. 测试以确保计算机未感染已知的恶意软件
2. 测试入侵检测系统是否能够快速识别在短时间内来自单个计算机的多个请求
3. 检查可能允许攻击者滥用的功能配置（例如，SSL，网络服务器，DNS）
4. 鉴别可能允许 DoS 攻击的逻辑性缺陷

入侵是外部攻击的另一种形式。有许多方式可以完成从外部入侵系统。这些攻击是基于某人“闯入”一个系统来获取信息。下面的列表中描述某些入侵途径：

- 社会工程
- 注入攻击（SQL，恶意代码）
- 账户损害（收集，密码重置）
- 利用已知的漏洞（防火墙，操作系统，框架，应用程序）
- 恶意软件攻击
- 针对不安全配置的攻击
- 授权缺陷
- 应用程序逻辑攻击（利用应用程序缺陷进行功能滥用，特别是在基于 Web 的应用程序中- 例如，在电子商务购物应用中未按工作流执行步骤以获取折扣或积分）

拦截某人从组织内部发送到另一个组织的网络传输不会被视为入侵攻击，而是内部违规。

内部威胁
最大的威胁可能是来自内部的。以下内部攻击源需要被考虑：

- 公司间谍，受信任的员工可能会出售公司信息，包括客户信息，商业秘密，员工权限信息等
- 外包开发人员，测试人员和其他人员（如客服代表）获得的信息。有时候这些人员从外包公司离职的时会把这些信息一并带走。
- 窃取硬盘驱动器和其他物理存储设备的内部员工
- 不满的员工，通过泄露机密信息，或通过伪造报销发票来谋取钱财等行为，以企图损害公司利益。

安全性测试格式与结构
每个执行安全性测试的组织都有自己一套格式化的详细测试方法。安全性测试通常可以使用与其他类型测试相同的设计格式，唯一的区别是测试的对象和测试环境。

即使一个组织遵循 IEEE 829-2008 和 ISO 29119 [ISO / IEC / IEEE 29119-3]等标准，该标准的使用应适合组织的需要。总之，这些标准形成了对于各种测试计划文档应包含哪些内容的标准理解。在许多情况下，测试用例和测试过程（脚本）可以在提供格式化结构的测试管理工具中定义和实现。

测试用例是相对独立的测试描述形式。它们不需要顺序执行。如果需要顺序执行以实现特定的测试目标，可以在测试过程或脚本中以特定的顺序来组合测试用例。测试用例通常用于测试单一的情形，例如，在安全性测试中，测试登录功能可能包含用于验证密码格式需求被正确实施的多个测试用例。

测试实施期间，在测试过程规格中开发测试用例，组织并确定优先级。测试过程指定测试用例执行顺序。如果使用测试执行工具运行测试，则在测试脚本（自动化测试过程）中指定执行顺序。被测试的功能比较重要时需要使用测试过程。例如，测试过程对于测试“重置密码”流程很有帮助。

当需要基于经验的测试时（如探索性测试），测试条件和预期结果不需在测试之前定义，但是测试条件和实际结果应被安全性测试人员记录在测试报告中。

3.3.2 基于策略和规程的安全性测试设计

在设计测试以验证安全策略和规程时，安全策略和规程将成为测试的基础。从这个角度来看，安全性测试几乎是安全审计的一种手段。

安全策略和规程不应该是安全性测试的唯一基础，也需要从其他视角来考虑安全性测试。设计测试以验证安全策略和规程的目标是：

- 了解安全策略和规程的目的和范围
- 评估安全策略和规程的可测试性
- 创建与安全策略和规程直接相关的测试

例如，可能有一个规程声明：“所有 XYZ 公司的 IT 系统将登录尝试失败次数限制为三次。在三次登录尝试失败后，账户将被锁定一段时间。没有有效用户账户的个人将无法访问我们的 IT 系统，并且必须联系 IT 支持服务部门验证身份才能获取临时密码。

这个例子是一个完全可以测试的安全规程，测试需要以下步骤：

1. 尝试登录应用程序失败三次，第三次登录失败时应出现账户锁定消息，任何进一步登录尝试也将收到账户锁定消息。

2. 联系 IT 支持服务部门并验证身份，测试账户下的电子邮箱将会收到一个临时密码。
3. 使用临时密码登录，登录成功并具有相应的访问权限。
4. 创建符合密码策略的新密码，新密码应当被接受。
5. 登出系统
6. 新创建的密码登录，登录成功并具有相应的访问权限。

请注意，步骤 4 还提供了测试密码策略的时机。

并不是所有的安全策略都是可测试的。例如，XYZ 公司审核记录包含所有具有日期或时间戳的审核事件，并且可追溯到特定个人。厂商定制的日志能够提供足够信息以达到安全策略要求应被视为足以满足安全审计的目的。

然而可能的测试方式是定义和执行相关测试以覆盖所有审核事件。需要执行某些操作以触发一系列会被记录到审计记录中的样本事件，并且所记录信息的准确性是需要被验证的，诸如用户 ID，日期和时间戳。

3.4 安全性测试执行

3.4.1 有效安全性测试环境的关键要素和特性

尽管很多形式的测试可以使用与其他系统共享服务器和网络的测试环境，但这种情况对于安全性测试有一定的风险，因此需要通过隔离的方法来构建安全性测试环境。在测试不受信任的应用程序时尤其如此（如第三方或开源应用程序）。

某些安全性测试，如测试功能控制和会话管理时，可以在无高风险的常用集成测试环境中执行。然而，当测试不可信的未知代码时，恶意软件可能会破坏服务器和网络，建议在隔离或虚拟测试环境中进行测试。

安全性测试环境的主要属性如下：

1. 隔离–与其他系统隔离（取决于恶意软件风险级别）
2. 完整 – 整个测试环境需要在如下方面反映目标（生产）环境：
   - 被侧的系统和应用
   - 操作系统（精确的版本和配置）
   - 网络
   - 中间件
   - 硬件品牌，处理器，内存）
   - 移动设备（制造商，处理器，内存，电源管理）
   - 数据库
     访问权限
   - 浏览器和插件
     共存的应用程序
   - 数据（经过混淆处理的测试数据或生产数据）

3. 可恢复 - 根据需要重复测试，以及在系统或数据损坏时能被恢复

3.4.2 计划和批准在安全性测试中的重要性

为什么安全性测试人员在执行安全性测试之前需获得批准，有如下原因：

- 在几乎所有国家，（试图）获取系统数据和信息都是违反法律的。甚至在某些国家获取安全性测试工具都是违法的。这意味着在大多数安全性测试活动中，你将违反一项或多项法律。能够执行测试的唯一可行方式是从系统或数据所有者那里取得不追究责任声明，并获得管理层的批准。

- 安全性测试可能触发入侵检测警报，并且测试人员可能会被误认为是内部恶意员工。尤其是渗透测试这种特定案例更需要得到批准。
- 安全性测试可能导致严重的系统故障和宕机，风险应该是事先知晓的，并且部署可行的预防措施。

没有针对安全性测试的预先特定授权，测试人员可能违反安全策略和规程，这可能使测试人员遭到解聘或起诉。

安全性测试的授权表单应包含以下信息：

- 授权实体名称
- 测试人员和实体的名称
- 工作说明
- 授权周期（起始到终止）
- 其他相关详细信息，如源 IP 地址，用户账户等等
- 证明：
  - 客户负责提供被侧系统
  - 客户有权授权安全性测试
  - 客户已对所有系统和数据进行了备份
  - 客户已测试了系统按需备份与还原
  - 客户了解安全性测试相关的风险
- 测试实体是无恶意的免责条款
- 授权签订此协议的客户代表签名

样例表格可以参考[OWASP3]。

3.5 安全性测试评估

与大多数测试一样，在执行每个单独的安全性测试时同时执行评估，安全性测试评估是对安全性测试结果的评估。当识别出安全缺陷（漏洞）时，应提交一份事件报告，至少说明以下内容：

- 发现漏洞的测试人员名称
- 发现漏洞的测试环境
- 执行的测试步骤（以便于重现测试结果）
- 安全漏洞的性质
- 安全漏洞的程度和范围
- 安全漏洞的潜在影响
- 建议的补救方案

安全性测试事件报告可以使用与其他测试相同的事件管理系统提交。应为安全性测试报告分配一个特殊类别，并且需要加以保护，以防止未经授权的人员查看。这种情况可能是：

- 独立组织正在执行安全性测试，并且在查看事件限制较少的工具中提交报告
- 发现了安全漏洞但不会被立即修复

- 内部人员可能被视为利用安全漏洞的潜在威胁

IT 审计员应该能够确定是否限制对安全性测试结果的访问。

在主要的安全性测试工作结束时，例如在系统测试结束时，可以发布最终安全性测试报告。取决于漏洞解决的状态，此报告可能需要被视为机密。

3.6 安全性测试维护

在许多情况下，修改安全性测试流程可能只包括增加新型测试以响应新型威胁。但有一件事是肯定的。安全性测试对象和威胁每天都在变化，因此安全性测试流程需要设计为易于改变。

市面上不断出现新的工具以有助于安全性测试执行。安全性测试人员应该跟上这些发展趋势，评估哪些工具可以增强安全性测试的能力和灵活性。

4.贯穿整个软件生命周期的安全性测试-225 分钟

关键词

滥用用例、模糊测试

“贯穿整个软件生命周期的安全性测试”的学习目标

4.1 安全性测试在软件生命周期中的作用

AS-4.1.1 (K2)解释为何在生命周期过程中最能实现软件的安全性
AS-4.1.2 (K3)为给定的软件生命周期（例如，迭代、连续）实施适当的安全相关的活动

4.2 安全性测试在需求阶段的作用

AS-4.2.1(K4)从安全的角度分析一组给定的需求以识别它们的不足

4.3 安全性测试在设计阶段的作用。

AS-4.3.1 (K4)从安全的角度分析一份给定的设计文档以识别其不足

4.4 安全性测试在实施活动中的作用

AS-4.4.1 (K2) 了解在组件测试过程中安全性测试的作用。
AS-4.4.2 (K3) 根据给定的代码规范实施组件级别的安全性测试（概要）
AS-4.4.3(K4)对给定的组件级别的测试结果进行分析，从而从安全的角度判断代码的充分性。
AS-4.4.4 (K2) 了解在组件集成测试过程中安全性测试的作用。
AS-4.4.5(K3)根据给定的系统规范实施组件集成的安全性测试（概要）

4.5 安全性测试在系统测试和验收测试活动中的作用

AS-4.5.1(K3) 为安全性测试搭建一个端到端的测试场景，以验证一个或多个指定的安全需求以及测试一个已描述的功能过程。
AS-4.5.2 (K3)展示为一个指定的验收测试在安全性方面定义验收标准集的能力。

4.6 安全性测试在维护阶段的作用

AS-4.6.1(K3) 基于指定的场景，实施端对端的安全再测试/回归测试方法。

4.1 安全性测试在软件生命周期中的作用

安全性并非是在应用程序构建完毕后再进行测试或添补的。相反，它是在整个构建过程中通过以安全性为导向的设计和验证活动实现的。与软件测试大体一样，安全性测试也是开发生命周期中必须进行的一个过程。

4.1.1 从生命周期的角度解读安全性测试

软件的生命周期过程提供了在特定时间点执行一系列紧密衔接的特定活动的框架。例如，在对应用程序进行设计之前应该先获取用户的需求。软件生命周期的选择取决于组织的性质、项目和相似因子[IEEE 12207]。本大纲和安全性测试中的概念和技术可以应用于任何生命周期过程，无论是顺序还是迭代。

大纲的第三章描述了一个通用的样本软件生命周期的安全性测试过程。将安全性测试集成到软件生命周期中的原因将会在下文章节进行讨论。

规定生命周期中应当进行安全相关活动的时间
例如，在捕获和定义用户需求时，业务或系统分析师应提出诸如以下问题：

- 需要什么级别的安全访问？
- 是否有任何数字或实物资产需要特殊的安全防御？
- 应用程序要达到何种程度的“开放”？
- 安全风险有哪些？

另一个例子是在编码期间。此时，开发人员拥有应用安全代码实践的最佳机会以避免攻击，如 SQL 注入攻击和内存缓冲区溢出攻击。而在项目的后期阶段，要发现和解决这类型的缺陷是十分困难和昂贵的，主要是因为缺陷发现后，许多其他的软件组件也可能需要进行类似的解决和修复。

提供审查的检查点
例如，应检查安全需求或用户故事，以确保用户需求中与安全相关的方面已得到充分的调查和记录。还应检查代码变更以检测是否存在内部员工或承包商所制造的恶意代码。

提供测试的检查点
例如，在开发中，应当记录和执行组件测试以验证源代码实践得到了遵循和成功的实施。

提供项目完整的入口准则和出口准则
该实践有一个例子：所有组件，只有在其所有与安全相关的活动（开发和测试）都已成功完成时，才可以被接受到集成测试环境中。这在项目的后期阶段尤为重要，因为此时的一个安全漏洞可能会带来影响整个系统或应用程序的安全风险。

4.1.2 软件生命周期中与安全相关的活动

下列与安全相关的活动是与其他项目活动一起进行的，而非以自身的生命周期进行。

需求：根据所使用的软件生命周期，以各种方式收集和定义需求。我们应当认识到，需求可能不仅指用户和相关干系人的需求。还可能存在其它需求，例如监管需求、技术需求和业务需求等。

需求目标包括：

- 在组织内和组织外从各个角度理解和识别安全需求。例如，尽管一项业务的客户可能不属于组织内部，但他们有对其私人信息保密的需求。
- 仔细清楚地记录安全需求，这样实施和测试活动才可追溯回需求，需求也可得到验证和确认。

需求活动包括：
- 定义所有受影响的有可能提出需求的知识人员。
- 使用各种方法，包括访谈、研习会等形式来收集每个小组的安全需求。这个活动也可以在诱导其他需求时一起进行。
- 以方便检查和跟踪的方式记录需求。
- 检查需求是否准确、完整、易理解和清晰。

设计：根据需求中所表述的需要设计体系或应用程序。需求里描述了安全需要，而设计将安全需要转换为可执行的解决方案。

设计目标包括：
- 创建满足所表述的安全需求的系统设计或应用程序设计。

设计活动包括：
- 分析记录的需求
- 用最可行的方法安全地开发应用程序
- 根据软件的生命周期，使用适当的技能记录设计。例如在迭代的方法中，我们可能使用白板的方式来进行设计环节，而在其他过程中，设计可能是在模型中显现的。

实施：这通常也称为编码活动

实施目标包括：
- 将需求和设计转化成满足需求里所表述的功能需求的安全代码
- 实施其他必要的流程或技术（防火墙、记号等）以满足安全需求

实施活动包括：
- 创建满足安全需求的代码
- 执行组件测试以验证实施是否准确、有效和安全
- 执行组件检查以直观地确认实施是否准确、有效和安全

系统测试:
要注意有一些软件生命周期模型，例如迭代的交付方式，会在更短的时间内增加新的组件或提升现有组件，另外，相比其他更加注重顺序的方式，其进行系统测试也更为频繁。

系统测试目标包括：
- 执行端对端的测试，以观察各种系统组件经实施并集成到完整系统后系统（硬件、软件、数据、人员和流程）的综合功能和性能
- 进行测试，从系统角度检查安全需求是否得到实施

系统测试活动包括：
- 在与最终目标环境近似的环境中执行安全性测试，从而从先前的实施和集成活动的开发环境中实现过渡

验收测试：
这是测试的最后阶段，系统的用户或用户代表相信系统在其目标环境中能展示必要的能力。

验收测试的目标包括：
- 由用户或代表用户的代理商，按照系统所建立的安全相关的验收标准执行安全性测试。很多情况下，安全相关的验收标准主要关注功能安全的控制和过程。

验收测试活动包括：
- 将系统安装到其运营环境当中
- 根据验收标准执行安全性测试
- 根据测试结果决定是否验收

应当注意到，系统和验收测试都必须是"黑盒"测试或刺激响应测试，不考虑整体系统的内部结构或组件的情况。之前的组件和集成测试考虑和探索了组件的内部架构以及它们与系统间的互动，因此提供了补充性的评估。

维护：
在系统开始提供服务后，我们可能需要进行额外的开发工作来纠正已发布版本的系统的缺陷（修复性维护），从而适应运营环境中的其他变更（适应性维护）、或扩充或增加功能（完善性维护）。

对于系统维护来说，安全性测试主要是对缺陷纠正的变更进行测试（确认测试）以及对关键功能进行测试（回归测试），从而：
- 确保不因维护活动而将新漏洞引入系统
- 确认在变更情况下现有的安全防御仍然有效

在这种情况下，维护活动包括升级（例如，运营系统和数据库）、代码变更、数据转换以及平台迁移。

实际上，所有的维护活动都应像初次开发活动一样得到相同的投入及关注。否则，新漏洞的引入风险将会严重地损害整个运营系统的安全。

4.2 安全性测试在需求阶段的作用

一般而言，要了解以下需求方面的注意事项：

- 许多公司在撰写清晰、明确、完整、正确和可测试的基本用户需求时遇到困难。
- 在整个项目过程中，需求在很大程度上会受制于变更，因此需求维护会是一个挑战。
- 要用特殊的技巧来理解用户需求和其他需求，如合规和技术需求，这样才能够将这些初步需求形成文档，或将这些需求输入到需求管理工具。
- 需求当中会存在差距和错误。因此，要进行验证和确认。
- 需求应包含对质量属性的需求，如安全、性能、可用性等。通常这些属性会被忽略，而只关注功能性。

所面临的挑战在于，在项目的一整套需求中使安全得以理解和表达。对评价需求时，使用检查单作为指引是一种有效方法。检查单中可以包含许多检查项，涵盖各种各样的主题。关于安全相关的属性，以下列举了需求评估好的出发点：

隐私需求

- 所有用户组及其相关的数据隐私需求是否都得到了识别和记录？
- 受此需求影响的所有数据类型是否得到了识别，相关的隐私需求是否得到了定义？
- 用户访问权限是否已得到识别和定义？

合规需求（符合安全政策）

- 所有相关的安全政策是否得到了识别和记录？
- 安全政策的期望是否得到了识别和记录？

常见漏洞- 随着安全攻击的变化，这些漏洞也会随之发生变化，但在定义需求时，应将这些漏洞定义为风险。这些漏洞则会成为安全性测试的基础。

- 是否已经将所有记录下来的、已知的未来常见安全漏洞定义为已知风险？

可测性

- 对需求的描述是否足够充分，使得在描述安全检测和其他检测时能够以该需求文档为基础？
- 是否识别和澄清了模糊术语，如“数据处理必须保证安全”和“经授权的人员才有访问权限”，使之更为具体并具备可测性？

可用性- 安全和可用性之间有许多权衡之处。例如，如果网站上的用户登陆十分复杂而困难，用户就会放弃并使用其他网站。

- 需求是否反映了与指定功能相关的安全过程处于适当水平？
- 安全流程是否清晰易懂？
- 是否为那些可能在访问信息时遇到问题的合法用户他们提供了解决措施？

性能-安全和性能之间存在权衡。例如，高级别的加密可能会降低性能。

- 这些需求是否反映了与指定功能相关的安全效率处于适当水平？

4.3 安全性测试在设计阶段的作用

应确定和避免会导致安全退化的设计方式。测试相关活动有助于识别可能易受攻击的软件系统设计，并对具有可识别的强大安全属性的软件系统设计进行指导。

IEEE 计算机学会安全设计中心[IEEE1]推荐以下关键设计方法：

- 获得或给予，但不假设、信任
- 使用无法绕过或篡改的身份验证机制
- 验证后授权
- 对数据和控制指令进行严格分离，不处理从非信任源接收的控制指令
- 定义一种确保所有数据都得到明确验证的方法
- 正确使用加密
- 识别敏感数据及定义处理方法
- 始终以用户为中心
- 了解集成外部组件是如何更改你的攻击面的
- 在考虑对象和参与者的未来变化时应当灵活

4.4 安全性测试在实施活动中的作用

安全性测试，就像在其他类型的测试活动一样，都是从实施的最低级别开始，对将被组装到整个系统中的单个软件组件进行测试。对这些组件进行了静态评估后，测试提供了一个额外的评估层级，检查回应有效和无效输入的动态行为。

4.4.1 组件测试中的安全性测试

4.4.1.1 白盒/透明盒测试注意事项

涉及检查、代码走查、审计和技术评审活动整个范围的静态测试已经提及过了。

所谓的白盒和/或透明盒（结构化）测试指的是在软件设计或实施可见性的基础上所设计的测试。相反，黑盒测试（功能性和非功能性）测试并非基于对此类结构化信息的访问，而是简单的刺激-反应测试。

白盒测试可以以在模块中执行的特定控制为目标并确定其有效性。组件结构的可见性也使得可以对测试覆盖进行度量，如所执行的可执行语句的百分比，所执行的判定结果的百分比，或者遍历的逻辑路径的百分比。

结构化安全性测试可以由自动化静态分析工具和安全扫描工具执行。模糊测试是一种安全性测试技术，通过向被测组件或系统输入大量的随机数据（称为 fuzz）来发现安全漏洞。相对于黑盒模糊工具，白盒模糊测试（针对小软件块，函数，类）可能能够在更短的时间内获得有用结果。

白盒模糊测试工具能够通过检测被测代码来检测内存损坏，缓冲区溢出等。

在结构化测试期间可以识别和修复以下安全漏洞：

- 内存缓冲区溢出
- 内部员工或承包商嵌入的恶意代码
- “后门”访问（通过只有开发人员知道的未公开接口进行访问，这是为绕过正常的安全控制故意而为之）

4.4.1.2 功能性安全性测试的注意事项

任何级别的安全性测试的充分性应通过确认指定的安全要求是否得到了满足来确定。这是除了对安全要求、安全风险评估和类似文件中没有明确规定的压力的反应进行观察以外的另一种方式。在对安全漏洞进行测试时需要创造力，因为测试人员要发现软件开发人员所忽略的内容。

4.4.2 组件级别的安全性测试设计

可以在文章“安全编码实践 Top 10”[CERT1]中找到高级编码最佳实践的示例集，其中写道：

“任何组件的测试都应该评估是否违背了以下做法：

- 确认输入。
- 注意编译器警告。
- 安全策略的架构师和设计。
- 简单化。

- 默认拒绝。
- 坚持最小特权原则。
- 清理发送到其他系统的数据。
- 练习纵深防御。
- 使用有效的质量保证技术。
- 采用安全编码准则。”

参照此类最佳实践检查表进行的测试应评估是否有可能违反了上述做法，该评估以包含现实威胁建模且记录充分的风险分析为基础。换句话说，要把关注点放在攻击概率和妥协后果方面最关键的要求。

4.4.3 组件级别的安全性测试分析

充分性的一个关键指标包括对测试覆盖的评估。覆盖的各种衡量指标源于所执行测试的性质。

基于需求的测试利用系统来保证自身能够满足所规定的要求。若不考虑执行（黑盒测试），可以从以下任意方面来衡量测试覆盖：

- 所测试需求总数的百分比
- 所测试的指定用例/滥用用例的百分比
- 所测试的关键功能、场景或任务线程的百分比

数据驱动测试利用系统来保证其在输入数据的范围和组合上的行为，试图通过将数据空间划分为等价类并从每个类中选择一个代表，尽量减少所选择的测试值（期望这个类的元素在故障检测能力方面是等效的）。成对覆盖准则和 N 平方覆盖准则属于数据覆盖准则的典型形式。

基于模型的测试保证所选建模符号的覆盖。当模型使用前置符号时，准则可以包括因果覆盖和后置条件中所有选言支的覆盖。对于代数模型符号，公理的覆盖属于典型的覆盖准则。

对于基于过渡的模型（使用包含节点和弧的显式图），图覆盖准则包括节点（状态）的百分比，过渡的百分比，过渡对的百分比和周期的百分比。

结构测试基于对实际实施的可见性和分析提供保证。通过简单枚举，测试覆盖通常表示为由测试执行的应用程序中的包、类、方法、判定或可执行代码行的百分比。后者被称为语句覆盖。

环路复杂度是某个元素中不同独立路径数量的度量，且可利用具有节点（判定结点）和弧（路径）的控制流程图来实现可视化。基于控制流的准则中覆盖程度最强的是路径覆盖，其对控制流程图中入口到出口的所有路径进行测量。由于回路的原因，穷尽路径测试通常不可行，因此可从被视为关键（关键路径覆盖）的所选逻辑路径或所执行的判定结果的百分比（分支覆盖）这些方面来表示其他不太严格的准则。

4.4.4 组件集成测试中的安全检测

由于较低级别的组件被集成到子系统中并且最终集成到完整的目标系统中，所以安全漏洞的可能性就不仅仅单独考虑的每个组件中漏洞的总和。相反，由于组件之间以及组件与较大系统和组织元素之间的交互，出现新的攻击向量是有可能的。

另一方面，组件之间的一些交互可以缓解或阻止导致安全漏洞的可能序列。同样，安全性测试人员在寻找开发人员忽略的方面时需要有创造力。

集成测试可以展示系统设计的复杂性及其行为的稳定性。集成测试方法（如自上而下或自下而上）可能影响显示安全问题的时机或对额外特定安全性测试的需求。

4.4.5 组件集成级别的安全检测设计

与组件测试一样，集成测试应基于包含现实威胁建模且记录详尽的风险分析来设计。将单个组件集成在一起的时候，注意基架有可能（以桩和驱动程序的形式）要在集成期间对通过系统的不完全路径进行测试。随着越来越多的执行组件被添加到系统，这个基架也会被逐步删除，从而能够对功能性以及可能被利用的漏洞新路径进行更全面的评估。

4.5 安全性测试在系统测试和验收测试活动中的作用

4.5.1 安全性测试在系统测试中的作用

系统测试是对完全集成组件进行的第一次端到端运行。虽然系统测试通常是在开发环境中完成的，但它应该显示集成完成之前未观察到的系统涌现性。安全需求通常会与一个或多个功能性需求结合考虑。

如“在执行 x 的过程中，系统不应允许 y 的发生”。在执行功能性测试时，测试人员应探究安全约束可能会被违反的途径。

功能性需求，包括用于安全的功能性需求，通常会涉及需求。其他规范，如用例、滥用用例、过程模型和状态转换模型描述了可用于定义安全性测试端到端测试场景的过程。

4.5.2 安全性测试在验收测试中的作用

验收测试与系统测试的区别在于，验收测试是在实际操作环境中执行的，甚至是在系统可操作的实际设置中执行。这种测试使得对性能和其他行为进行合理评估得以进行，该评估以通过外部接口的交互为基础。这种测试最终还会对系统进行设置，让外部威胁来源每日在系统中寻找漏洞。

理想情况下，验收测试应确认项目初始目标是否已经实现。这是通过对测试进行设计和执行来确认验收标准得到满足来实现的。安全需求应该记录在验收标准中。

对验收标准进行定义和记录的最佳时机是在系统开发或购买之前。因此，供应方和采办方之间可以进行初步了解，即使两者都在同一个组织中。在项目期间更改或提出验收标准也很常见，因此应分析这些标准对安全性测试的影响。

在安全性测试的情境中，验收标准本质上可以是整体性的。例如，可以存在验收标准点，指定整体系统安全方面哪些内容是可以接受的。这包括应用于所有系统功能的标准，如用户验证，用户权限，加密级别，审计跟踪等。在其他情况下，可能需要特定的安全验收标准。例如，某些功能（如支付超过某一金额的付款）可能需要两个人对付款进行批准。

4.6 安全性测试在维护阶段的作用

回归测试旨在确认在进行修改后，系统中所有先前可接受的行为保持完整。在安全性测试的消极面，这种确认需要检查系统是否成功地持续抵制试图破坏已建立的安全控制的做法。加强可用性或效率很容易会在安全控制方面做出让步。

安全回归测试的重点应在于确认所有安全需求得到满足，并对在维护活动期间可能引入的新漏洞进行测试。

回归测试通常利用以测试单个函数为基础的测试用例集合来执行。然而，对于安全性测试而言，这通常不足以检测出存在安全影响的回归缺陷。端到端回归测试场景更加健全，置信度更高，可以安全地执行完整的事务。

对于这种类型的回归测试，每次对系统进行更改时，都应定义和测试一组安全性测试方案。请记住，系统变更可能会扩展到硬件，配置文件，操作系统，DBMS，网络和软件 - 以及其他任何系统组件。回归缺陷可以在以上任意组件的变更中出现。一些回归缺陷可能存在安全影响。

示例场景：
用户能够登录到网站并安全地完成购买，用户的个人信息不受影响。
用户只能执行其用户权限和特权中定义的操作。（如在薪资管理部门工作的用户可能能够添加新员工，但无法访问其银行信息。）

5. 测试安全机制 - 240 分钟

关键词

反恶意软件，身份验证，授权，非军事区，加密，防火墙，哈希，内部威胁，入侵检测系统，恶意软件，恶意软件扫描，网络区域，域欺骗，网络钓鱼，腌制，系统强化，漏洞扫描程序

测试安全机制的学习目标

5.1 系统加固

AS-5.1.1 (K2) 理解系统加固的概念以及在增强安全性方面的作用
AS-5.1.2 (K3) 演示如何测试通用系统加固机制的有效性

5.2 鉴定与授权

AS-5.2.1 (K2) 理解身份验证与授权之间的关系，以及这两种技术在安全信息系统中是如何应用的。
AS-5.2.2 (K3)
演示如何测试通用身份验证和授权机制有效性

5.3 加密

AS-5.3.1 (K2) 理解加密的概念以及如何在安全信息系统中使用加密技术
AS-5.3.2 (K3) 演示如何测试通用加密机制的有效性

5.4 防火墙和网络区域

AS-5.4.1 (K2) 理解防火墙的定义以及网络区的使用，理解他们在安全信息系统中的应用
AS-5.4.2 (K3) 演示如何测试已有防火墙和网络区域实现的有效性

5.5 入侵检测

AS-5.5.1 (K2) 理解入侵检测的概念，以及他们如何在安全信息系统中应用的。
AS-5.5.2 (K3) 演示如何测试已有的入侵检测工具实现的有效性

5.6 恶意软件扫描

AS-5.6.1 (K2) 理解恶意软件扫描工具的定义，以及他们如何在安全信息系统中应用的。
AS-5.6.2 (K3) 演示如何测试已有恶意软件扫描工具实现的有效性

5.7 数据混淆

AS-5.7.1 (K2) 理解数据混淆的概念，理解数据混淆在安全信息系统中的应用
AS-5.7.2 (K3) 演示如何测试数据混淆方法的有效性

5.8 培训

AS-5.8.1 (K2)理解安全性培训作为软件生命周期一项活动的概念，以及为何在安全信息系统中需要安全性培训
As-5.8.2 (K3) 演示如何测试安全性培训的有效性

5.1 系统加固

近年来，涌现出不同种类的安全机制，用来保障数据和物理资产的安全。每一种机制都可以使用不同的方式进行部署，有些是通过工具和架构，有些则是人工部署。对于绝大多数的应用场合来说，单独使用一种安全机制是不够保障信息的安全的。每种机制都有利弊。

安全性测试员需要理解每一行防御代码的细微差别，并据此设计出合适的测试，以验证和确认这些措施的有效性。高级别的安全性测试员则需要理解本章所描述的每一种机制的内涵及影响，从而设计出能够提供持续的安全性测试框架的测试架构。

5.1.1 理解系统加固

现代系统变得越来越复杂，因此其攻击表面就不断增长。设计脆弱性错误带来的设计错误，架构脆弱性带来的源代码缺陷，或者因为配置脆弱性带来的系统配置严格性缺失，都可能带来系统的脆弱性。

系统加固是一个一步步完成的过程，其通过部署安全策略和不同层次的保护措施，减少攻击表面。系统加固的主要目的，就是使得系统更加安全，减少安全性丧失带来的风险。

依据上下文（使用场景）的不同，加固可以在不同层次予以部署：

- 加固软件或者硬件部件；
- 加固产品或应用
- 加固系统
- 加固复杂系统

在企业级和技术级层面，还应该包括的安全保护措施包括：

- 删除不必要的软件，其中可能包含缺陷；
- 删除不必要的库和开发者使用的工具，其中可能包括缺陷
- 删除不必要的账户和登录（攻击向量）；
- 删除不必要的应用（可能包括缺陷），和网络服务（攻击向量）；
- 删除不必要的外设和硬件，例如 USB 端口，读卡器等；
- 随时针对系统打补丁，安装升级包，例如激活自动升级；
- 升级配置
- 遵循编码规则，避免架构脆弱性
- 配置远程登录服务，例如 rsyslog，一旦出现攻击，攻击者只能删除被攻陷的机器上的 log 文件，远程登录服务器上的 log 文件仍能保留。

系统加固还应使用下面这些安全机制：

- 加强身份验证和和有效的管理授权，只为特定的角色提供必需的权利；
- 加密，通讯加密和本地存储加密；
- 防火墙（个人防火墙，系统防火墙和网络应用防火墙）；预定义好的安全区（例如在沙箱中执行）；
- 入侵检测系统；
- 反恶意软件/反间谍软件
- 数据和应用的模糊化处理（例如针对逆工程的保护）

系统加固对于保护企业的敏感资产非常关键，但是安全规则必须在正确的级别予以部署，要与系统可用性做平衡。在极端情况下，可能不得不关闭这些保护措施，因为他们可能会妨碍（阻塞）公司的生产力。

5.1.2 测试系统加固机制有效性

测试系统加固机制的有效性可以使用若干种方式。测试者需要综合考虑被加固的系统或应用的特质、被保护资产的敏感度以及被标识出来的威胁。系统加固针对正确角色，使其有限制地访问系统，只开放必须的服务，监督应用升级。因此，要测试系统加固的有效性，测试者就要设计测试用例，查看加固措施是否正确工作，是否在正确的地方被部署，是否使用正确的方式被部署。同时，也要测试系统加固措施是否过于严格，对于安全风险来说，有没有必要如此严格。

有些系统加固测试可能是基于审查的，或者基于审计的；有些测试可能是基于某些特定用户群体执行特定的一些操作，或者访问特定的数据。

测试可能包括：

- 审计数据库和应用服务器的配置，确认缺省的密码是否已经被修改；
- 审计系统配置，标识出不需要的服务和网络端口；
- 验证部件、库和应用版本，确保他们不是过期的和脆弱的；

可以运行脆弱性扫描器，降低资产脆弱性测试任务的难度，这对于复杂系统尤为重要，例如多站点环境。可以使用静态分析工具来检测与编码规则的冲突，这些冲突可能会带来架构脆弱性。面向安全的分析对于检测脆弱性，非常有用。

5.2 身份验证和授权

5.2.1 身份验证和授权之间的关系

一个企业的敏感资产（例如一系列客户的银行账号，某个新产品的设计）必须得到保护，并且只能被已授权用户访问。

授权，是基于某个使用者的标识符验证通过，且拥有可以回答问题的令牌：

- Login，用户是谁？
- Password：确认真的是这个用户么？

可以使用不同的身份验证机制，防止破解授权机制的攻击，或者偷窃密码的企图。这些机制包括，检测较弱的密码，使用一次性密码（OTP），指纹，软件许可证，硬件 Token 许可证，以及类似的身份验证措施。

按照系统结构，应用上下文以及企业需要（易于配置登录/密码），身份验证机制可以包括本地验证，服务器验证，网络验证，单一登录（SSO）以及类似的机制。

授权可以用于以下目的：

- 验证某个已授权用户是否可以执行某个操作（例如用户可以登入服务器，但是不能修改数据；或者某个用户可以授权使用 FTP 服务器，但只能在特定区间使用）；
- 决定系统资源的访问级别；

未被授权的用户不能访问系统，或者只能有限制地访问系统（不能操作敏感数据），在这个原则下，身份验证和授权之间有强烈的联系。例如，在一个购物网站，未被授权的用户可以看到商品列表，但是在购买某个商品之前，用户必须先创建一个用户账号。验证通过的用户可以购买某个产品，但是不能执行管理功能。

5.2.2 测试身份验证和授权机制的有效性

攻击者的目的，或者是偷窃密码，或者绕过系统执行未被授权的行为。通常来说，他们会利用系统不同的弱点，例如编码错误（缺乏输入过滤），过时且脆弱的库，系统配置错误（保留缺省密码，缺省的权限），以及较弱的密码（例如很多人使用“123456”作为密码）

企业会制定一些必须遵守的密码规则，但是如果用户自己不能保证密码的安全，密码规则就形同虚设。另外，密码规则必须能反映出当前的密码定义的最佳实践。有些实践措施可以在 SANS 研究所的密码编制准则中找到[SANS2]

针对身份验证和授权机制的测试，可以包括：

- 暴力测试或者字典攻击，尝试发现用户密码。第一步就要尝试 “123456”，“111111”，生日，宠物的名字，等；
- 查找输入过滤的缺失，例如不用任何已知的登录/密码，通过键入 SQL 请求获取授权；
- 输入未被授权的 URL（在 FTP 账户输入../../）或者输入 URL（网站地址/admin）尝试获取敏感数据的访问权；

另举例如下：可以利用目标系统的脆弱性（该系统可能没有升级），触发未被预设的行为，这样通常会获取系统的控制权，允许特权升级；

5.3 加密

5.3.1 理解加密

为了避免泄漏敏感数据，就算是访问存储数据，或者在用户和服务器之间交换数据，都可能会用到加密机制。哈希算法和撒盐技术，是加密过程中可能使用到的方法。

加密，是使用加密算法和密钥，将明文数据编码成密文数据的过程。这样一来，只有拥有访问权限的授权用户才能够使用解密机制访问数据。只有授权用户才能够共享和知晓密码。这样做的目的，是让加密过程足够强壮，能够防范攻击者。这些攻击者可能通过恢复明文数据，而成功盗取加密数据。使用加密算法可以帮助确保保密性，完整性，敏感资产的可用性，确保不能操作这些数据。

加密协议可以保护：

- 保存在系统中的信息，例如数据库、逻辑的加密磁盘，整个加密硬盘中存储的加密之后的密码；
- 通讯过程中的信息，例如加密的电子邮件，加密的通讯协议（SSL，TLS）

首要的和最知名的加密协议包括

- 对称加密：使用共享的密钥
- 不对称加密：使用私有和公共密钥

5.3.2 测试常用加密机制的有效性

有些加密机制强度较弱，尤其是密钥很短，或者使用静态密钥。有些则是机制比较脆弱，因为他们要么没有使用最佳实践方法来实现，要么实现过程存在代码缺陷（例如缓冲区溢出）。

测试加密机制主要包括

- 测试设计的脆弱性
  - 评估在对称加密中是否使用了正确的模式
  - 验证加密使用的密钥长度不太小（例如 RSA 密钥长度如果是 2015 就比 2028 长度的密钥安全性低）；
  - 确认许可证的有效性，如果许可证是自我签署的，需要验证发出警报的能力（SSL-trip 能够避免中间人攻击）；
  - 重放攻击（例如针对 Wired Equivalent Privacy WEP 协议的攻击）
  - 针对加密协议的攻击，以验证其强度级别
- 测试架构的脆弱性
  - 代码审查（例如，验证没有使用缺省函数 random()来生成随机数（种子），随机算法是很容易被破解的）；
  - 模糊测试，查找未预知的行为
  - 时钟攻击（分析执行加密算法的时间令牌）
  - 电量分析（用于加密的硬件设备）
- 测试配置脆弱性
  - 评估加密协议配置（例如基于 TLS 配置指南，评估针对管理员的传输层安全（TLS）服务器端的配置，客户端的授权协议）
  - 服务器端的 TLS 加密顺序，检查是否存在能够降级或者重构正在使用的密钥的任何手段
- 测试老化，验证机密算法是否有可能随着时间推移而变得弱化，易于被破解。

5.4 防火墙和网络区

5.4.1 理解防火墙

按照【Chapman 2000】的解释，“防火墙是一个组件或者一套组件，能够严格将访问限制在被保护的网络和互联网之间，或者被保护的网络和其他网络之间”。防火墙实现并执行一个安全策略，该安全策略基于已授权和被禁止的通讯的定义。防火墙可以是基于主机的（软件运行在单一主机上，该主机监视应用的输入和输出），或者基于网络的（软件监视网络之间的交通）

防火墙主要的任务，是通过过滤网络上的数据流动，控制不同的、被信任的网络区之间的交通。这样一来，来自不被信任区域的恶意的交通就可以被检测到并被阻塞。
网络区是一个带有预定义的信任级别的、被标识的子网络，。

- 互联网/公共区域是不被信任的；
- 若干被称为隔离区或 DMZ 的安全区具备不同级别的可信任度；
- 一个或几个私有/内部网络可以被视为是最被信任的。

网络区是防火墙配置的一部分：使用网络区来定义不同网络之间的被授权的流动。所有被禁止的交通都会被阻塞。

通常来说，防火墙基于以下三点过滤通信：

- 源地址和目的地址以及协议（以太网或者 IP 地址，TCP/UDP 端口等）；

- 协议选项（分段，TTL 等）
- 数据大小

网络应用防火墙（WAF）基于以下两点过滤通信：

- 连接等同于用户
- 数据过滤（例如使用模式描述）

5.4.2 测试防火墙有效性

考虑到协议的数量，他们不同的选项，以及被保护网络的复杂度，很难高效率地配置防火墙。测试防火墙的有效性，需要考虑：

- 端口扫描，验证安全策略是否正确实施；
- 使用畸形网络数据包和网络模糊技术来探测未预期的行为（例如拒绝服务）；
- 分段攻击，绕过过滤特性之后，尝试在防火墙后面实施攻击

另一个测试举例如下：针对 WAF，编码并压缩数据，或者将数据模糊化，以此隐藏恶意信息并尝试实施攻击。

5.5 入侵检测

5.5.1 理解攻击检测工具

每年，攻击的数目都在增加。入侵技术进化飞快，没有系统能够 100%安全。

入侵检测系统（IDS），是一个能够监视不同级别的活动（来自网络针对应用的活动，OSI 模式的七个层级的活动），据此检测是否存在违反安全策略行为的一个系统（独立的设备或者应用程序）。一旦检测到异常行为，IDS 就会发出警报，并且分析警报之后可能的行为（例如交通阻塞，虚拟补丁）。

按照 IDS 标准，基于两种安全模型，描述了一个 IDS 的设计模型：

- 负面安全模型（基于签名的检测，或黑名单检测）：规则是：“所有没有被显式禁止的，都是允许的”。入侵检测基于已知的攻击列表或者模式；
- 正面安全模型（基于行为的检测或白名单检测），规则是：没有显式被允许的都拒绝”。入侵检测基于被保护系统的行为规格说明，例如输入字符要按照通用表达式的形式描述。如果行为与正常或系统的预期行为不符，就认为是入侵。被信任的交通可以用来生成规格说明。

IDS 与防火墙不同之处在于，防火墙从内向外监视交通，防止入侵；而 IDS 分析可疑的入侵，一旦确认就会发出警报。

5.5.2 测试入侵检测工具有效性

基于场景的检测很容易被绕过，因其只能检测已知的攻击。测试要包括下面这些规避技术:

- 字符编码或数据修改（例如，增加白色空格，行尾，等）
- IP 分段，TCP 分割
- 加密，模糊化；
- URL 编码

基于行为的检测会生成大量的虚假正向结果和虚假负向结果。虚假负向结果，是应该被报出但没有报出的警报；如果产生某个 IDS 还不能识别的新攻击，就会产生虚假负向结果；或者某个规则能够检测出某些攻击但是不能检测出其他攻击时，也会产生虚假负向结果。同样的，一定要考虑这类检测方法的精确度。如果一个攻击者从正常的行为中派生一个 IDS 行为，就可能产生一个包含着入侵行为的新的规格说明。于是这个交通就不被认为是异常的。辅助性测试需要使用恶意的交通，以便在被授权的交通中增加新的入侵规格说明。

有些输入可以定义一系列的针对 IDS 的测试，比如“Profile Protection Intrusion Detection System” [PP-IDS]，和“Web Application Firewall Evaluation Criteria”“[WAFEC]。

5.6 恶意攻击软件扫描

5.6.1 理解恶意攻击软件扫描工具

恶意代码可能会感染服务器和终端用户的计算机，为其发明者提供预期特权和目标敏感数据。可以使用不同的方式，比如带有恶意附件的电子邮件，虚假 URL，客户端代码执行等方式，将恶意代码放置在目标处。

反恶意攻击应用程序是用来分析，检测并删除从不同源头收到的恶意代码的软件，反恶意软件应用还可以检测：恶意软件，网络钓鱼和网络嫁接。

反恶意软件使用的主要检测特性，是基于签名的策略。其原则是在数据库中查找某些带有已知模式的数据，这些数据可能是可疑代码。然而，新的恶意软件，或者其签名没有在数据库中存在的恶意软件，就没法被检测到，因此就能感染受害者。在反恶意软件中也可以嵌入探索性机制，能够标识出已知恶意模式的轻微变形，帮助对抗这种问题。

5.6.2 测试恶意软件扫描工具的有效性

恶意软件和后门的开发者会使用不同的技术来保护他们的代码，对抗逆工程技术，或者被反恶意软件检测到。这些技术包括：

- 利用恶意软件使用的一些系统库函数，比如可以使用 FindWindow 关闭反恶意软件应用程序；
- 字符串混淆技术，使得反恶意软件不能理解恶意代码行为（例如使用加密）。例如：将 Java 脚本存储在 PDF 文档中；或者使用压缩技术，如针对可执行程序的 Ultimate Packer；
- 动态加载函数和库（例如限制针对恶意代码的分析）；
- 自动升级应用程序（例如 Skype 木马）

恶意软件还可以使用其他硬件资源，比如图形处理单元 GPU，来解压缩恶意代码，将其存储于内存，让处理器运行。这种情况下，无法在其运行之前分析恶意软件。

从功能测试的角度讲，类似 Eicar[EICAR]的工具（反恶意软件的测试文件）可以检测反恶意软件的有效性，无需开发出真正的恶意代码。

在部署一个新的反恶意应用程序或升级现有的反恶意软件应用程序的时候，应该先在有代表性的平台进行测试，然后再推广到整个企业。有过这样的案例，反恶意软件错误地将合法的操作系统文件识别为恶意软件，将其隔离，于是关闭了整个企业的计算能力。

5.7 数据模糊

5.7.1 理解数据模糊化

模糊（有时也称为数据面具），是一种使得数据和源代码对于人类不可理解的机制。

这种技术主要用于保护敏感数据，免受如下所示的攻击：

- 拷贝：绕过许可证保护机制；
- 逆工程技术：理解代码后，利用代码的脆弱性

数据模糊也可以用来允许公司雇员（支持团队，功能测试人员等）与非敏感数据工作，而将敏感数据与明文显式分离。有些人将数据模糊定义为“数据匿名”，因其将个人数据匿名化。

模糊技术也可以用来保护源代码免受简单的拷贝-粘贴操作（例如，保护一个新的革新性算法），也可以避免被逆工程理解后重用这些代码。

有时候开发者需要优化其代码，使其效率更高。这可能也造成模糊的源代码（例如将代码的某些部分使用汇编语言编程）。有些网页应用级别的攻击就带有脚本入侵。为了实施入侵，攻击者需要了解网页的结构和 HTML 页面。模糊技术可以帮助保护敏感的和紧要的 HTML 页面，例如连接和管理页面。

可以使用 base64 编码，XORing，随机重命名函数，方法重载，tab-return-space 删除，Shuffling 等技术实现模糊技术。加密也是常见的模糊技术，但是它的问题是，加密数据对于拥有合法密钥的人来说，仍然是可见的。

注意：数据模糊技术经常被攻击者使用，用来隐藏其恶意代码和攻击。

5.7.2 测试数据模糊方法的有效性

必须严格配置管理模糊化之后的数据和用于模糊化的密钥，才能保证所使用密钥的正确版本。否则，数据就不能在使用逆模糊技术之后，保证其可用性。

因为在某些测试中可能包括一些私有数据，所以可以出于测试的目的，将数据模糊化，使得用于系统测试环境中的数据是匿名的。敏感数据，例如在健康信息系统中的用户信息，绝对不能泄露给测试人员。测试应该包括：

- 暴力破解或字典式攻击，试图从已模糊化的数据中获取明文数据。

验证代码模糊化的测试包括：

- 字节代码 Java 的逆工程技术（例如使用 JavaDecompiler 重新生成源代码），或者使用.Net 程序（例如抽取带有.NET Reflector 的 Net 源代码）；
- 暴力破解攻击，有些模糊化机制是比较脆弱的（例如使用 unXOR [Chopitea]）。
-

理论上，面对反模糊，代码没法保护自己，因为总是可以使用调试器。虽然有工具，其目的就是保护代码免受反编译，但在保护代码所体现的私有信息方面，仍然存在风险和限制。

5.8 培训

5.8.1 安全培训的重要性

在整体安全图片中，人是最薄弱的环节。因此，就需要持续不断的培训提醒人们遵循已有的安全策略的重要性以及为什么需要这些策略。这类培训要贯穿软件生命周期，一旦增加了新的安全策略或者出现了新的威胁，就应该升级培训。培训应该覆盖社会工程的攻击和内部威胁等方面。

5.8.2 如何测试安全培训的有效性

例如，安全性培训程序，强调使用强度较高的用户密码保证保密的重要性。
测试应该包括

- 尝试在电话中，让社工伪装成技术支持人员，让用户泄露他们的密码；
- 在工位附近查找带有密码的即时贴，尤其是键盘下面；
- 运行密码审计工具，标识出脆弱的密码。这种方法的风险，在于使用这种工具的时候，测试人员可能会看到明文密码。

另举例如下：开发人员没有在数据输入域设置一个域级编辑器，防止用户输入一个 SQL 指令。这类错误，安全性测试工程师就可以键入一个 SQL 命令，看到用户数据库中的内容。这表明开发人员需要额外的关于安全编码实践的培训。同时，也要检查其他开发人员的编程实践能力，常看类似上文所示的编码习惯是不是广为存在。如果是，那么就需要一个通用而广泛的改进培训。

第三个例子，测试人员可以尝试进入未被授权的办公室，随意浏览打开的文档。

6. 安全性测试中的人为因素 - 105 分钟

关键词

攻击者、僵尸网络、计算机取证、黑客、侦察、脚本小子

安全性测试中的人为因素：学习目标

6.1 了解攻击者

AS-6.1.1 (K2) 解释人类行为如何会导致安全风险以及影响安全性测试的有效性
AS-6.1.2 (K3) 在给定场景中，展示攻击者可用来发现特定目标的关键信息的方法以及可采取的环境防护措施
AS-6.1.3 (K2) 解释针对计算机系统的攻击的常见动机和来源
AS-6.1.4 (K4) 分析攻击场景（已发生和发现的攻击）并识别可能的来源和攻击

6.2 社会工程

AS-6.2.1 (K2) 解释安全措施如何受到社会工程学的影响

6.3 安全意识

AS-6.3.1 (K2) 了解安全意识对于整个组织的重要性
AS-6.3.2 (K3) 对特定的测试结果采取适当措施来提升安全意识

6.1 了解攻击者

在信息安全领域，人类既是最大的威胁也是防护中的最弱点。

安全攻击须由具有多种技能和各种动机的人实施，人是大多数安全攻击得以实现的最重要因素。要防御这些攻击，光了解并实施安全技术是不够的，同样重要的是了解恶意攻击者的心态、动机和方法并在防御时关注人类自身的弱点。

6.1.1 人类行为对安全风险的影响

任何攻击的关键都在于信息收集（侦察）阶段，攻击者尝试寻找和收集关于目标的信息，所有关于一个组织、其使用中的系统和其他的公开信息以及存放在互联网上的信息都有可能在不知不觉中被发现并被用于攻击，这不是“如果”的问题而是“何时”的问题。除了组织正式发布的信息，员工还会在社交网络上发布有关公司的信息，这种在数量和内容上持续变化的信息往往向攻击者提供着关键信息。

攻击者在攻击系统时不会理会安全策略和预定义过程，他们根据收集的信息决定自己的策略，他们会为每个实施选择性搜索和访问已知 IP 地址的攻击而不断更新自己的知识库。

一家公司在指定其安全政策时，往往基于当前已知的处境和事实。有时，这可能不包括所有公开可访问的信息。即使包括了，这些信息也可能变化，在创建时依然有效的安全性测试在发布的信息发生变化时可能无法提供足够的覆盖率。

6.1.2 了解攻击者的心态

在侦察或踩点活动中，攻击者会尝试各种被动和/或主动手段来寻找关于目标的各种信息。大多数面向公网的 IT 设备都会在网络上留下痕迹，这些痕迹必然会被找到。谷歌（包括谷歌地球和谷歌街景）或其他搜索引擎、Shodan [Web-5]、Facebook、LinkedIn 和其他社交网络是搜寻目标信息的首要来源，IP 地址、网页、电话号码、姓名和电子邮件地址结构、操作系统、应用程序都可以向攻击者提供有用的信息。

谷歌搜索引擎可能被用来搜寻目标的特定信息，Google Hacking Database [Web-4]中可以找到数以百计的查询。Shodan [Web-5]是另一个用于查找特定信息的工具，例如：在某个区域内哪些公司正在运行有缺陷版本的 Apache。

大多数信息无需实际连接到目标系统即可找到，其他一些工具包括：

- Whois [Web-13]
- Ripe (European IP Networks)数据库 [Web-12]
- DNS searches [Web-25]

攻击者也可使用实际接触系统的主动侦察技术，这类工具可以检测主机、开放的端口、操作系统和应用系统，这类方法和工具包括：

- Pinging - Fping [Web-15], Hping [Web-19]
- TCP/UDP 扫描 – Nmap [Web-20], Zenmap [Web-21]
- 操作系统检测 – Nmap [Web-20], Xprobe2 [Web-22]
- 服务指纹识别 (Nmap 可以确定在发现的打开端口上运行的服务的类型和版本，这是通过比较服务的指纹与 Nmap 的指纹数据库来实现的。)

由于在大多数国家黑进一个系统是被法律禁止的，黑客会试图消灭所有的黑客行为的证据，其余毁灭证据的原因有：延长停留时间、在将来继续使用此系统并利用被黑的系统或系统组成的网络（僵尸网络）来攻击其他系统。为此，攻击者可以部署类似 NetCat [Web-14]的工具或使用类似 IP Tracer [Web-7]的网站，以及隧道技术和更改日志文件。

其他用于隐藏证据的方法和工具还包括隐藏工具[Web-16]、rootkits 和文件流，这些提到的工具大部分都可以通过互联网找到，通过下载 Kali Linux [Web-17]的最新版和在 OWASP 网站 [OWASP1]上搜索即可获得其中许多工具。

6.1.3 针对计算机系统的攻击的常见动机和来源

许多对信息系统的攻击和破坏都来自于组织内部，恶意的系统用户（内部的黑客和内部人员威胁）会试图作为经授权的网络用户来损害系统，大多数时候其动机都是报复，但是最近的趋势显示经济间谍或盗窃在增加。

外部黑客的攻击仅是一小部分，对信息的好奇依然是黑进信息系统的主要动机之一，获得一些重要企业或组织的某些信息并知道其他人并不掌握则是另一个动机（声望）。其他动机包括名声、挑战自我、无聊和报复，后者被认为是最危险的（最主要的动机）。

攻击者通常可以以动机和能力分类，在攻击者领域的低端是“脚本小子”，他们只是执行其他人创造的攻击方法，而在领域的高端是专业组织和个人（政府、黑客主义）。黑客主义主要是基于政治目的而攻击系统，但也有出于经济目的和人口制度背景的。

动机可以从纯粹玩乐到因任何原因（如政治、意识形态、经济、战争、商业、恐怖主义）而击溃一个系统或组织。

黑客能力的分布从具有一些系统和网络知识的使用简单家庭电脑的个人到经过专业训练和教育的能使用实验室、代理网络和其他技术设备的专业人士。对潜在的攻击者有一个映像可以帮助组织实施必要的防护并为安全性测试策略提供指导。

6.1.4 了解攻击场景和动机

安全事件被定义为与安全相关的系统事件，在事件中系统的安全策略被违反或破坏。 [RFC2828]

找出发生了什么和谁应对安全相关事件负责是计算机取证学的一个目标，其着重于寻找攻击的数字证据。

证据恢复过程有三个阶段：

1. 获取和验证
2. 分析
3. 报告

6.1.4.1 获取和验证

组织应有事件管理流程来确保证据被收集和存储后将系统恢复到其原有状态（受攻击前），该流程开始于系统管理员收到 IDS 或其他监控手段的告警。其他典型安全事件的症状有：

- 可疑的日志条目
- 无法解释的用户账户
- 被修改的文件/文件夹

- 不正常的服务
- 不正常的系统行为
- 未成功的登录尝试

收到告警后，执行如下流程：

1. 制作被调查系统的快照或副本以收集所需的所有证据。
2. 在验证证据后（是真实和完整的副本），创建一个副本并将其存储在安全的位置。
3. 分析证据。
4. 取证流程完成后，清除（根除）造成事件的原因
5. 将系统置成正常状态（恢复）。

在这些步骤中，使用补丁或安装新的软件来修复漏洞，在报告结果时，应该描述所遵循的流程以及期间使用的工具。

6.1.4.2 分析

在黑客尝试攻击后，可以通过检查系统日志文件和活动的网络连接来寻找攻击的来源，给所有日志文件创建副本并抓取进程状态信息是很重要的。在主动攻击过程中，在将攻击者赶出门外之前收集系统信息可能是有意义的。

任何通过互联网的攻击都可以追溯到发起的 IP 地址，无论其是使用电子邮件或是互联网连接。这只是时间、金钱、精力和对成本作评估的问题。大多数攻击者使用代理、代理链、Tor 网络 [Web-9]或其他免费的匿名手段来掩盖他们的真实 IP 地址，攻击者使用的代理越多，追踪起始地址所需的时间也越多。代理的物理所在地的当地法律也可能阻碍调查。

使用 Netstat（Windows） [Web-10]、Tracert [Web-11]和 IP Tracer 网站 [Web-7]等工具可以发现攻击者并追踪起始 IP 地址，Netstat 可以显示一台机器的所有网络连接、端口和运行的服务，该工具可用于搜索任何奇怪的或未知的 IP 地址或端口号。注意：在微软 Windows 操作系统上也有一个 tracert 程序（在 Linux 和 OS/X 中是 traceroute），但上述基于 Web 的服务与这些操作系统内置程序互相独立。

在含病毒的电子邮件的头部可以找到发送邮件的 ISP 的 IP 地址。然而，对于大多数基于 Web 的电子邮件（Gmail, Yahoo mail, Outlook.com）来说，这是供应商的 IP 地址，为了找到真正的 IP 地址必须查看 X-Originating-IP 的值，使用 Whois 数据库可以找到 ISP 组织的详细联系方式以进一步调查。应当注意的是，电子邮件可以源自私人服务器和开放的中继服务器，在这种情况下，识别电子邮件的实际来源可能会非常困难。

用僵尸网络发起的攻击调查起来很困难，攻击者无需与僵尸网络服务器或僵尸网络客户端建立连接，因此追踪他们非常困难或几乎不可能，在这种情况下，调查僵尸客户端可以找到服务器，但是必须能访问服务器才能调查到攻击的真正来源，这些服务器的所有者未必意识到他们的服务器已经成为了僵尸网络的一部分。

6.1.4.3 报告

安全漏洞的报告请参见第七章所述。

6.2 社会工程

我们可以实施所有能想得到的技术防御来保护数字资产免受外来的侵害，但最终，员工（用户和管理员）需要访问这些资产来完成他们的工作，他们可能需要使用身份验证来从其桌面、笔记本、智能手机、平板电脑或其他设备来访问系统，如果 IT 经理家中的工作区域的安全可被轻易损害，则任何对办公室和办公设备的物理安全防护都变得无意义了。

人类和人类的行为是对安全最大的威胁，如果人们草率对待敏感信息，就容易在安全场所留下太多痕迹并在公共场所过度扩散这些信息（口头的和电子的）。

社会工程是利用人类的习惯行为作为攻击向量的艺术，作为社会人，人们愿意信任和帮助陌生人，这就创造了一个可被攻击的弱点，通过操纵、影响和说服有用的人，攻击者可以尝试收集访问和授权的详细情况以及其他敏感信息。

这种利用可以通过直接的人际互动，也可以通过计算机/网络设备。

直接的人际互动由人亲自完成，包括：

- 尾随或搭车（没有得到认证的人跟随员工进入受限区域）。
- 窃听（在他人不知情的情况下倾听别人的私人谈话）。
- 肩窥（在他人不知情的情况下从他人肩后看电脑上的工作或书面工作）。
- 使用电话（例如扮演经理或技术支持人员来从不生疑的用户那里获取密码）。

基于计算机的社会工程包括：

- 发送感染了恶意软件的电子邮件。
- 使用聊天或即时消息程序。任何匿名的人都可以使用聊天或即时消息程序与世界上任何地方的另一个人聊天而无需知道他的真实身份，此外，即时消息程序传输的数据可被轻易地嗅探。
- 使用弹出窗口。例如，网络连接丢失的消息窗口可能出现在用户的电脑屏幕上，此时用户被提示重新输入用户名和密码，则由入侵者预装的程序可以将此信息发送到远程站点。
- 发送垃圾邮件。垃圾邮件中充斥着欺诈性的优惠和链接，点击这些链接会被恶意软件感染从而暴露整个网络。
- 说服人们访问受感染（被操纵）的网站。这些钓鱼行为可以被广泛地发送，也可以是高度个性化的（鱼叉式钓鱼）。

对社会工程没有单一的防护方式，可以实施防护来控制损害，但主要的防护是组织中每个级别人员的安全教育和意识水平。

6.3 安全意识

6.3.1 安全意识的重要性

如本文其他章节所述，威胁模型在不断变化。网络在发展，新应用在出现，新接口投入使用，因此新的漏洞也在被引入和发现。

除了这些技术方面，还有人为因素。曾被发现但未成为问题的风险很容易被遗忘，相应的防护措施也会被撤除，这为黑客行为和社会工程攻击提供了更多的机会，因此需要定期的安全意识培训以使安全管理员和全体员工警惕并了解威胁模型的变化，安全意识培训可以关注不同的用户群体：开发人员、运营人员、管理层和普通工作人员。

6.3.2 提高安全意识

保持“安全意识”的心态很重要，除了关于公司安全防护的一般信息，培训应包含在安全性测试或实际事件中发现的真实案例研究，以这些案例为基础更容易讨论应在组织中实施什么防护和变更。

作为这一章节的概述，在安全意识培训中应包含以下问题的答案：

- 他们（我们）是怎么做的？
- 业务后果是什么？
- 调查和处理事件的成本是多少？
- 修复问题的成本是多少？
- 本应如何避免事件发生？
- 将要实施哪些变更？

7. 安全性测试评估和报告 - 70 分钟

关键词

验收标准（acceptance criteria），攻击向量（attack vector），仪表盘（dashboard），出口准则（exit criteria）

安全性测试评估和报告的学习目标

7.1 安全性测试评估

AS-7.1.1 (K2) 理解随着项目范围和目标的变化，安全性的期望和验收标准也需要相应变化

7.2 安全性测试报告

AS-7.2.1 (K2) 理解保证安全性测试结果保密和安全的重要性

AS-7.2.2 (K2) 理解需要创建合适的控制和数据采集机制，从而为安全性测试状态报告提供及时、准确、精确的源数据（例如，安全性测试仪表盘）

AS-7.2.3 (K4) 可以通过分析给定的中期安全性测试状态报告，从而决定准确性、可理解性和干系人适宜度的级别

7.1 安全性测试评估

需要对安全性测试结果进行度量并且对与安全性期望、出口准则和/或验收标准相关的状态进行评估，从而决定测试是否完成。

很难在一个项目开始时就知道所有的安全风险。另外，干系人和用户的关于所需安全性水平的期望有时会发生改变。例如，当干系人认识到一个新的威胁时，他们可能改变最初的想法，要求达到更高层次的安全性。因此，应该在整个项目过程中持续进行安全性风险评估并且将安全性风险评估结果纳入安全性测试计划和执行。

7.2 安全性测试报告

7.2.1 安全性测试结果的保密性

通常来说测试人员在测试结束后会比大部分开发人员或设计人员对测试对象了解更多。当进行了充分的测试后，测试人员会发现系统中最重要的强项和弱项。在安全性测试中也同样如此。

当进行了安全性测试之后，测试人员会发现隐藏的安全漏洞和脆弱点。如果将这些漏洞告知直接干系人以外的人将可能会带来负面影响。一个普遍的好的实践是：信息应该只提供给那些需要知道的人，特别是对于安全性测试结果而言，在分享这类信息时应保持谨慎。

7.2.2 为报告安全性测试状态创建合适的控制和数据采集机制

安全性漏洞所造成的影响通常被认为比其他"普通"缺陷造成的影响更敏感。因此测试人员在报告缺陷性质和隐含风险时需要更加准确和精确。在大部分项目中，安全性缺陷的严重程度会比同类功能性缺陷更高。

这一切意味着管理层更关注安全性缺陷、安全性风险和可能的解决方案。测试人员必须在安全性缺陷报告中仔细评估已发现问题的影响和测试结果的精确性，应该以明确定义并且及时的方式提供安全性缺陷报告。此外，与管理层讨论确认他们希望在"何时(when)以何种方式(how)"获得安全性缺陷报告是一种很好的做法。

7.2.3 分析中期安全性测试状态报告

安全性测试报告可以在安全性测试过程中生成也可以仅在安全性测试结束时生成（例如，在系统安全性测试结束时生成或者在安全性测试（作为验收测试的一部分）结束时生成）。我们鼓励早期的安全性测试报告，因为它可以给予我们更多的时间来弥补安全性漏洞。如果安全性测试过程遵照本大纲中所描述的过程，测试团队可以发现漏洞并且将在所有测试活动中所发现的漏洞文档化。

安全性测试报告结构应该包含以下部分：

1. 报告标识符
2. 概述
    a. 执行概述
    b. 关键发现

3. 差异
    a. 所遵循的测试过程
    b. 与计划的测试过程相比的任何差异
    c. 所使用的方法和工具（配置，方针）
4. 综合评估
    a. 基于测试计划中指定的标准评估测试覆盖率
    b. 对于所有未测试项目或功能进行说明
5. 结果总结
    a. 安全性测试结果总结
    b. 所有已解决的安全性漏洞及其解决方案列表
    c. 所有未解决的安全性漏洞列表
6. 评估
    a. 基于出口准则对于已经观察到的测试结果和测试状态的评估
    b. 已识别（已分类）的风险以及未解决的安全性漏洞的影响
7. 活动总结
8. 批准

安全性测试报告的有效性依赖于以下几点：

- 报告的时间
- 报告的内容
- 报告的接受者
- 报告内容的调整与接受者对于信息的需要是否匹配

有时需要多份报告来满足各种干系人的需要。例如，给高层管理层的报告内容和给系统架构师的报告内容是不一样的。

8. 安全性测试工具 - 55 分钟

关键字

无

安全性测试工具的学习目标

8.1 安全性测试工具的类型和目的

AS-8.1.1（K2）解释静态和动态分析工具在安全性测试中的作用

8.2 工具选择

AS-8.2.1（K4）分析和记录安全性测试需要通过一个或多个工具来解决
AS-8.2.2（K2）理解开源工具的问题
AS-8.2.3（K2）了解是否需要评估供应商频繁更新工具的能力，以便及时了解安全威胁

8.1 安全性测试工具的类型和目的

黑客社区设计的漏洞攻击推动了安全性测试工具的发展，以抵御这些威胁。即使从早期的黑客活动（如密码破解），简单的工具也是由那些使用它们的人发明，创建和改进的。证明有效的工具在黑客社区中得到了共享，并得到了进一步的改进和增强。起初这些工具是为了专门的任务和环境而开发的。几乎所有用户都有技术背景，可用性不是问题。最终，一些黑客工具成为信息安全管理员和测试人员使用的合法安全性测试工具的基础。

例如，“John the Ripper”是一个早期的开源密码破解工具，最初被黑客用来猜测（破解）密码并获得对 Unix 网络或应用程序的访问权限。今天，这个工具已经过改进，用于合法目的来检测弱 Unix 密码。 [网络 26]

随着主要的测试和软件开发工具供应商和专业工具供应商开始开发安全性测试工具，其中许多工具实现了更广泛的功能和可用性。但是，这种广泛的功能导致了更复杂的工具配置和实施问题。

在早期安全工具出现的同时，Nessus，Metasploit 等框架的第一个版本作为开源工具开发，提供改进和扩展的功能，在某些情况下，它也是一个易于学习的 GUI。

今天，可用安全性测试工具的数量非常庞大。几乎所有的环境或任务都可以找到专用的测试工具，无论是开源还是授权。所有这些工具面临的挑战是，其中大部分都是部署非标准化测试的智能系统。这些系统的所有开发人员或多或少都会同意如何测试安全防御措施或测试漏洞。但是，这些工具可能会使用不同的测试数据，不同的测试实现和对结果的不同解释。

安全性测试工具可用于自动评估安全防护。安全性测试工具也可用于检测已知类型的漏洞。由于认识到相同类型的安全防御或漏洞可以以不同的方式实施，所以选择和使用安全性测试工具对安全性测试人员来说是一个挑战，因为这些工具在发现漏洞和验证防御措施方面存在差异。

Web 应用程序安全联盟[Web-18]和 OWASP [OWASP1]网站提供了分类工具列表。渗透测试框架 Backtrack [Web-23]（或 Kali Linux [Web-17]）提供了其他分类安全性测试工具的方法。

与开源工具的数量相比，商业安全工具的数量相当有限。
在本课程开发时（2016 年），我们只能找到有限数量的资源，对可靠且值得信赖的开源安全工具进行或多或少的全面概述。一个安全工具列表可以在 https://sectools.org 找到[Web-24]。预计高级安全性测试人员会维护他或她自己的可用工具列表，并随着工具市场的变化而更新列表。

静态和动态分析工具都可用于安全性测试。静态测试的优势在于它可以在开发生命周期的很早阶段执行。静态分析工具可用于大多数软件语言，并且通常具有报告安全方面的能力。

与其他测试类型相比，安全性测试环境中的动态和静态测试工具之间的区别有时会有点混淆。静态测试的定义与在测试中的系统或对象未处于操作模式时执行测试活动有关。安全动态测试工具探测系统而不是被测试应用程序并不罕见。从这个角度来看，这些动态测试工具被用作一种静态测试工具。例如，动态安全性测试工具可以执行数据库的静态扫描。当然，如果整个系统被认为是测试对象，那么这些工具确实是动态测试工具。

8.2 工具选择

8.2.1 分析和记录安全性测试需求

其中，以下的文件可以构成安全性测试的测试依据：

- 组织的安全策略
- 组织的测试策略
- 实际系统/项目的威胁和风险分析结果
- 要求和其他系统规格
- 系统架构和设计
- 安全（测试）策略
- 正在测试的系统或应用程序
- 已知的安全威胁，漏洞利用和漏洞
- 用户配置文件

所有这些和更多可以提供有关威胁和可能被利用的漏洞的信息。要求和设计文件应说明数据或信息如何受到保护。这将导致概述：

- 要测试的接口（包括 GUI）
- 要验证的协议和标准
- 促进使用安全编码实践的网络编码指南
- 要验证的系统组件配置（硬化）

需要确定安全性测试是开发活动还是维护/操作活动。所有这些信息都将导致安全性测试工具集的需求。

8.2.2 开源工具的问题

有关开源工具可能遇到的问题的完整讨论，请参阅[ISTQB®_ATM_SYL]。

如前所述，许多安全性测试工具都来源于开源领域。这些工具是分布式的，可以在各种许可下使用，这些许可都允许免费使用和修改源代码。并非所有的公司或项目可以考虑在开发过程中使用开源工具。基于监管合规性问题，组织可能只能被迫使用商业或其他认证工具。

这些许可证下的工具有许多优点和缺点。在许多情况下，开源工具可以免费获得，但组织可能需要具备技术能力来支持和特定配置。如果缺乏这种能力，那么从软件的开发人员那里可能会产生成本。管理和用户手册（如果有的话）大多数都是针对特定（技术）受众编写的，而且通常不会描述或覆盖该工具中的所有功能。像YouTube这样的媒体渠道最近是有关使用这些工具的信息的额外来源。

在为任何开源工具设置ROI计算时要考虑的方面包括：

- 工具的范围有限（大多数情况下不提供其他功能或其他功能）
- 学习管理，配置和使用该工具的时间
- 在生命周期中投入用户论坛和团队的时间
- 更新和升级所需的时间（以及升级的内部策略）
- 该工具的未来方向（某些工具可能会消失或投入商业用途）
- 工具支持社区的响应水平

对于大多数企业或项目来说，安全性测试工具所需的许可证数量仅限于一个或少数几个。只有大公司会考虑更多的许可证。许可证的数量将主要基于该工具提供的功能区域的总和（例如，Web应用程序，

Web服务，代码分析等）以及假定的频率，这些服务的使用时间以及安全性测试人员的数量使用该工具。

8.2.3 评估工具供应商的能力

如果某个工具是从供应商处购买的，那么该供应商应该提供许多服务，以使安全性测试服务能够启动并发展到所需的内部支持级别。

以下属性可用于评估供应商的能力：

- 提供的许可证类型（固定/桌面/浮动/令牌）
- 许可证可扩展性选项（每个功能区域，许可证数量）
- 帮助台/支持设施（支持小时数）
- 论坛/用户社区
- 更新频率
- 管理和用户手册
- 支持和维护合同

9.标准和行业趋势 - 40 分钟

关键字

基于共识的标准

学习标准和行业趋势的目标

9.1 了解安全性测试标准

9.1.1（K2）了解使用安全性测试标准的好处以及在哪里可以找到它们
9.1.2（K2）了解标准在监管与合同情况下的适用性差异

9.2 应用安全标准

9.2.1（K2）理解任何标准中强制性（规范性）和可选（信息性）条款之间的区别

9.3 行业趋势

9.3.1（K2）了解在哪里学习信息安全行业趋势

9.1 了解安全性测试标准

各种标准提供了对专业共识或监管义务的可见性。一个基于共识的标准代表了一个知识丰富的专家组的意见，并可供供应商和客户之间的合同协议（全部或部分）自愿使用。所谓标准的类型较少，这些标准来自更多非正式或自我认定的群体，可能是供应商特定的。

在受监管的行业（包括医疗，金融，交通和能源部门），政府机构可能需要遵守其自己的规定或对其他自愿性标准的解释。

9.1.1 使用安全性测试标准的好处

一般而言，标准为执行任务提供指导和一致性。通常情况下，标准由主题专家根据有效实践的共识制定。

以下是使用安全性测试标准的好处：

- 他们为安全性测试定义了一个框架，消除了从空白页开始的需要。
- 它们描述了有效的防御措施以及如何测试最常见的安全攻击。
- 标准可以定制以满足项目或组织的需求。
- 安全性测试的尽职调查可以通过遵循公认的安全性测试标准来证明。

9.1.2 标准在监管与契约情况下的适用性

在受监管的活动中，所有各方都需要意识到他们遵守强制标准的义务，因为不这样做可能会延迟或阻止对正在开发的产品的批准，并且在极端情况下会导致财务或刑事制裁。

在合同情况下，标准为谈判项目和产品要求的协议提供了合理和方便的基础; 他们提供了一个起点，而不是以无始无终的各方。基于共识的标准允许传达和采用最佳实践或适应特定情况。

除非由监管机构或者不可谈判的合同单方面强制执行，否则标准可以作为谈判达成的协议的基本框架，或者自行实施自己的工作。如果根据索赔或协议授予合同以遵守特定标准，则该实体有义务严格遵守这些标准并记录任何偏差。

9.1.3 安全标准的选择

当然，并非所有的安全标准都适用于所有情况。研究组织的系统，应用程序，敏感数字资产，风险级别和合规要求的最合适标准是组织的责任。了解许多标准可能需要量身定制以满足组织的特定要求也很重要。

关于通用安全标准的清单可以在第 10 章中找到。

9.2 应用安全标准

请注意任何标准中语言的确切使用情况：该词应标识符合标准的强制性要求，而词语应该并可能指示声明与标准一致的可选任务。一种典型的误用是混淆了这种区别，要么需要一个可选项目，要么将强制项目视为可选项目。

组织或项目特定的情况可能会偏离严格意义上的使用标准。对标准内容的遗漏，修改或补充的理由需要记录并由各方达成一致。

9.3 行业趋势

9.3.1 在哪里学习信息安全行业趋势

通用和特定于行业的新闻服务（出版物，网站，电子邮件分发）和活动（会议，贸易展览，专业社团会议）都提供关于新的或日益增长的关注的信息和讨论。属于重点专业的社会或实践社区可能会提供及时和有针对性的更新。随着新攻击发展的速度，电子警报可以提供最直接的可操作响应。

定期公布最常见或最具破坏性的漏洞可以确定广泛的趋势，但只有一个
应特别关注与行业，应用领域或与之相关的产品更具体的问题。这些问题更有可能通过专业出版物和新闻服务或技术会议和专业活动进行传播。

9.3.2 评估改进的安全性测试实践

随着现有技术的新技术或新用途的引入，对技术的滥用或利用往往存在一个机会窗口，直到其风险和限制得到更好的理解。

例如，考虑具有位置感知服务的移动设备。为了换取方便或其他煽动，个人似乎愿意允许每分钟跟踪他们的动作和活动。

犯罪，黑客行为主义，经济和政治代理人正在涌现更大范围的动机和更多的资源。勒索和保护计划已经从物理威胁转移到数字领域。

由意识形态驱动的个人大型特设网络可能会在很短的时间内针对他们的愤怒目标发出指示。企业间谍活动通常资金充足，动力十足。寻求经济和军事优势的民族国家资源特别充足，可能认为自己不受制裁或报复。

由于威胁一直在变化和发展，安全性测试人员必须随时准备应对下一个威胁。对行业的了解，密切跟踪安全趋势以及获取最合适的工具，为组织提供最好的防御。

10. 参考文献

ISTQB®文档

[ISTQB®_FL_SYL] ISTQB® Foundation Syllabus, 2011

[ISTQB®_ATM_SYL] ISTQB® Advanced Test Manager Syllabus, 2012

[ISTQB®_ATTA_SYL] ISTQB® Advanced Technical Test Analyst Syllabus, 2012

标准

[ISO/IEC/IEEE 29119-3] - Software and systems engineering -- Software testing -- Part 3: Test documentation

[IEEE 12207] - ISO/IEC/IEEE Standard for Systems and Software Engineering - Software Life Cycle Processes

COBIT - http://www.isaca.org

ISO27001 – Information Security Management - http://www.iso.org/iso/home/standards/management-standards/iso27001.htm

PCI – Payment Card Industry Standard - https://www.pcisecuritystandards.org/

书籍

[Chapman, 2000] Chapman, Cooper, Zwicky, Building Internet Firewalls, O'Reilly & Associates, 2000.

[Jackson, 2010] Jackson, Christopher; Network Security Auditing, 2010.

文章

[ComputerWeekly] http://www.computerweekly.com/news/2240113549/Cattles-lost-backup-tapes-highlight-risk-of-unencrypted-data-storage

[Northcutt, 2014] Northcutt, Stephen; Security Controls, SANS Institute.

[Washington Post, 2007] http://www.washingtonpost.com/wp-dyn/content/article/2007/05/04/AR2007050402152.html

指南

[Bittau] Cryptographic protection of TCP Streams (tcpcrypt)
https://tools.ietf.org/html/draft-bittau-tcp-crypt-04

[CERT1] Top 10 Secure Coding Practices
https://www.securecoding.cert.org/confluence/display/seccode/Top+10+Secure+Coding+Practices

[CERT2] http://www.cert.org/secure-coding/publications/index.cfm

[CERT3] http://www.cert.org/secure-coding/tools/index.cfm

[IEEE1] Avoiding the Top 10 Security Flaws
http://cybersecurity.ieee.org/center-for-secure-design/avoiding-the-top-10-security-flaws.html

[MDA1] MDA Glossary, DoD Missile Defense Agency, www.mda.mil

[NIST 800-30] NIST Special Publication 800-30, Rev 1, *Guide for Conducting Risk Assessments* (2012)

[NISTIR 7298] Glossary of Key Information - Security Terms, Revision 2 (2013)

[OWASP1] OWASP Secure Coding Practices Quick Reference Guide
https://www.owasp.org/index.php/OWASP_Secure_Coding_Practices_-_Quick_Reference_Guide

[OWASP2] OWASP Risk Rating Methodology
https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

[OWASP3] OWASP Sample Authorization Form
https://www.owasp.org/index.php?title=Authorization_form

[PP-IDS] US Government Protection Profile Intrustion Detection System for basic robustness environments, version 1.7, 25 July 2007.

[SANS1] 25 Most Dangerous Software Errors – http://www.sans.org

[SANS2] Password Construction Guidelines - https://www.sans.org/security-resources/policies/general/pdf/password-construction-guidelines

[WAFEC] Web Application Firewall Evaluation Criteria, wasc-wafec-v1.0.pdf, 2006.

报告

[WhiteHat Security, 2014] https://www.whitehatsec.com

网页

[CERT4] Vulnerability Notes Database - http://www.kb.cert.org/vuls/

[Chopitea] tomchop.me/2012/12/yo-dawg-i-heard-you-like-xoring/

[EICAR] www.eicar.org

[RFC2828] Internet Security Glossary - http://www.rfc-archive.org/getrfc.php?rfc=2828

[Web-1] Top 20 Critical Security Controls - http://sans.org

[Web-2] National Vulnerability Database - https://web.nvd.nist.gov/view/ncp/repository

[Web-3] Website Security Statistics Report - https://www.whitehatsec.com/resource/stats.html

[Web-4] The Google Hacking Database – http://hackersforcharity.org/ghdb

[Web-5] Shodan - shodanhq.com

[Web-6] NetCat - http://sectools.org/tool/netcat/

[Web-7] IP Tracer - http://www.ip-adress.com/ip_tracer

[Web-8] Computer Forensics, Cybercrime and Steganography Resources – http://www.forensics.nl

[Web-9] Tor Project - https://www.torproject.org/

[Web-10] Netstat - https://technet.microsoft.com/en-us/library/Bb490947.aspx

[Web-11] Tracert – http://www.tracert.com

[Web-12] RIPE Scan - https://www.ripe.net

[Web-13] Whois - https://www.whois.net/

[Web-14] NetCat – http://netcat.sourceforge.net/

[Web-15] Fping – fping.org

[Web-16] Hidetools – http://hidetools.com/

[Web-17] Kali Linux – https://www.kali.org/

[Web-18] Web Application Security Consortium – http://www.webappsec.org/

[Web-19] Hping - http://www.hping.org/

[Web-20] Nmap - https://nmap.org/

[Web-21] Zenmap - https://nmap.org/zenmap/

[Web-22] Xprobe2 - http://null-byte.wonderhowto.com/how-to/hack-like-pro-conduct-os-fingerprinting-with-xprobe2-0148439/

[Web-23] BackTrack - http://www.backtrack-linux.org/

[Web-24] Top 125 Network Security Tools - at https://sectools.org

[Web-25] DNS Lookup - https://who.is/dns/

[Web-26] John the Ripper - http://www.openwall.com/john/