软件测试

张圣林
zhangsl@nankai.edu.cn
泰达学院3区407
南开大学

课程信息

- 名称：软件测试（Software Testing）
- 学时：56（32理论+24实验）
- 课程主页
  - http://nkcs.iops.ai/courses/softwaretesting/
  - 维护最新课程文件
- 先修课程
  - 大学计算机基础、高级语言程序设计（C/C++/Java）
- 相关专业课程
  - 面向对象程序设计、软件工程、计算机网络、数据库原理

课程简介

- 成绩评定
  - 平时成绩 * 10% + 实验成绩*30%+期末考试 * 60%
    - 平时：出勤、课堂表现
    - 实验：实验报告
    - 期末：闭卷考试
- 教材
  - 软件测试教程（第2版），宫云战主编，机械工业出版社，2016
- TA
  - 申松，硕士一年级学生
  - Email: 2269429539@qq.com
  - Tel: 18222614196

课程简介

- 课程目标
  - 学习软件测试的基本概念和方法
  - 了解常见自动化软件测试工具的使用
  - 提高编程和软件开发水平
- 优秀软件测试工程师应具备的素质
  - 技术能力
  - 兴趣和自信心
  - 沟通能力
  - 团队意识
  - 耐心
  - 怀疑精神
  - 探索精神
  - 创造性

课程简介

- 内容
  - 引言 ☆☆
  - 第1章 软件测试概述 ☆☆
  - 第2章 软件缺陷 ☆☆
  - 第3章 黑盒测试 ☆☆☆☆
  - 第4章 白盒测试 ☆☆☆☆☆
  - 第5章 基于缺陷模式 的软件测试 ☆☆☆
  - 第6章 集成测试 ☆
  - 第7章 系统测试 ☆☆☆
  - 第8章 主流信息应用系统测试 ☆☆☆

为什么学习软件测试？

- 互联网公司技术类招聘
  - 研发工程师
    - 数据结构、算法导论、C/C++/Java/Python/Go、数据库、计算机网络
  - 测试工程师
    - 数据结构、C/C++/Java/Python/Go（要求不高）
    - 软件测试基本理论
  - 算法工程师
    - 数据结构、算法导论
    - 机器学习、深度学习
  - 运维工程师
    - 数据结构、 C/C++/Java/Python/Go、计算机网络、数据库

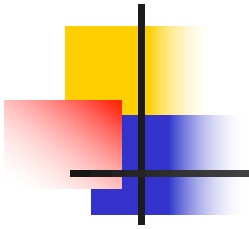
为什么学习软件测试？

- 互
  -
  -
  -
  -

算机网络

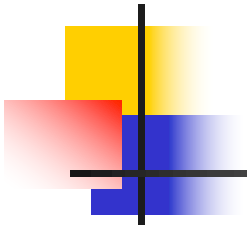
课程安排

- 第1/2周 2节理论课
- 第3-14周 2节理论课+2节实验课
- 第15周 2节理论课
- 第16周 元旦



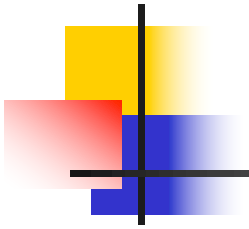
软件

- 软件
  - 计算机系统中与硬件相互依存的另一部分
  - 包括程序、数据及其文档的完整集合
  - 程序
    - 按事先设计的功能和性能要求执行的指令序列
  - 数据
    - 使程序能正常操纵信息的数据结构
  - 文档
    - 与程序开发、维护和使用有关的图文材料



软件

- 软件的分类
  - 按系统功能划分
    - 系统软件
      - Unix、Windows、Solaris
    - 支撑软件
      - 编译器，数据库管理，存储器格式化，文件系统管理，用户身份验证，驱动管理
    - 应用软件
      - 文字处理、信息管理、辅助设计、媒体播放、系统优化、杀毒、通信协作

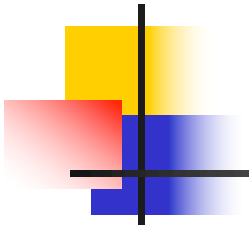


测试

- 用任何一种可能采取的方法进行的直接实际实验
  - 验证特性
  - 查找错误
- 考查人的知识、技能
- 对机械、仪器和电器等的性能和精度进行测量

软件测试

- 软件测试
  - 通过一些经济有效的方法，发现软件中存在的缺陷，从而保证软件质量。
- 软件测试 Vs. 医生看病
  - 测试人员 ←→ 医生
  - 软件 ←→ 病人



软件调试

- 软件调试/排错（Debug）
  - 利用测试结果和测试提供的信息进行全面的分析
  - 找到错误的根源和出现错误的原因
  - 纠正已发现的问题
    - 弄清了出错原因和错误根源，立即修正
    - 不确定出错原因，作出推测，再次进行测试
- 软件测试 Vs. 医生看病
  - 手段：体检、化验←→ 测试
  - 目的：分析病因，对症治疗←→ Debug

软件缺陷

- 一个真实的故事
  - 1947年9月9日，一个炎热的下午。当时计算机还是由机械式继电器和真空管驱动的庞然大物，机器有房间那么大，机房没有空调，所有窗户都敞开着。
  - Grace Hopper正领导着一个研究小组夜以继日地工作，研制一台称为“Harvard MARK II”的计算机，在对Harvard Mark II设置好17000个继电器进行编程后，突然，MARK II死机了 ……

软件缺陷

- 一个真实的故事（续）
  - 技术人员爬上去找原因，发现这台巨大的计算机内部一组继电器的触点之间有一只飞蛾，这显然是由于飞蛾受光和热的吸引，飞到了触点上，然后被高电压击死。
  - 这只小虫子被取出后，计算机又恢复正常。
  - 后来，Bug这个名词就沿用下来，用来表示计算机系统或程序中隐藏的错误、缺陷、漏洞等问题。

Photo # NH 96566-KN (Color) First Computer "Bug", 1947

9/9

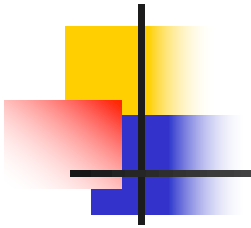
0800 Arctan started
1000 " stopped - arctan ✓ { 1.2700 9.037 847 025
9.037 846 795 correct
13" vc (032) MP - MC ~~1.9826447000~~ ~~2.13047764715~~ 4.615925059(-2)
(033) PRO 2 2.13047764715
correct 2.13067764715
Relays 6-2 in 033 failed special speed test
In relay " 11.00 test.
Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Multy Adder Test.

1545 Relay #70 Panel F
(moth) in relay.

debug

First actual case of bug being found.

1700 1630 Arctangent started.
closed down.



软件缺陷

- 软件缺陷
  - 既指程序中存在的错误，例如语法错误、拼写错误或者是一个不正确的程序语句
  - 也指可能出现在软件设计过程中，甚至在需求分析、规格说明或其他的文档中的种种错误

软件缺陷实例

- 用户为了保证自己业务的顺利完成，希望选用优质的软件。质量不佳的软件产品不仅会使开发者的维护费用和用户的使用成本大幅度增加，还可能产生其他的责任风险，造成开发方信誉下降。一些关键的应用领域（例如银行、证券交易、军事等）如果质量有问题，还可能造成灾难性的后果。
- 人们逐步认识到是软件中的缺陷导致了软件开发在成本、进度和质量上的失控。由于软件是由人来完成的，所以它不可能十全十美。虽然不可能完全杜绝软件中的错误，但是可以通过软件测试等手段使程序中的错误数量尽可能少，密度尽可能小。

软件缺陷实例

- Ariane 5 阿里亚娜5号火箭
- 波音787失控
- Therac-25辐射治疗仪
- 大型票务系统瘫痪
- 一个消息框的案例
- 许霆ATM案例
- 挪威$100,000网上银行转账
- 千年虫问题
- Intel奔腾处理器芯片缺陷
- Linux幽灵漏洞

案例1：Ariane 5号火箭案例

- 情形：1996年6月4日，阿里亚娜火箭在升空40秒后偏离飞行轨道，解体并爆炸。火箭上载有价值5亿美元的通信卫星。

- 这是一个著名的案例，几乎所有的的软件工程书上都会提到这个案例－ “一行代码错误带来了6亿美元的直接损失”。
- 让我们来看看这是一行什么样的代码？

原因分析

- Ada 语言代码

```
begin
    double d_bh; short s_bh;
    sense_horizontal_velocity(&d_bh);
     s_bh = d_bh; // OPERAND ERROR
end;
```

- 在飞行过程中，水平速度产生了一个很大的值，该值存贮在一个浮点型的变量中，在向一个短整型变量赋值的过程中，产生了溢出，该溢出导致程序异常，而该异常并没有被捕获和进行保护处理。
- 调查报告的原文如下：

during execution of a data conversion from 64-bit floating point to 16-bit signed integer value. The floating point number which was converted had a value greater than what could be represented by a 16-bit signed integer. This resulted in an Operand Error.

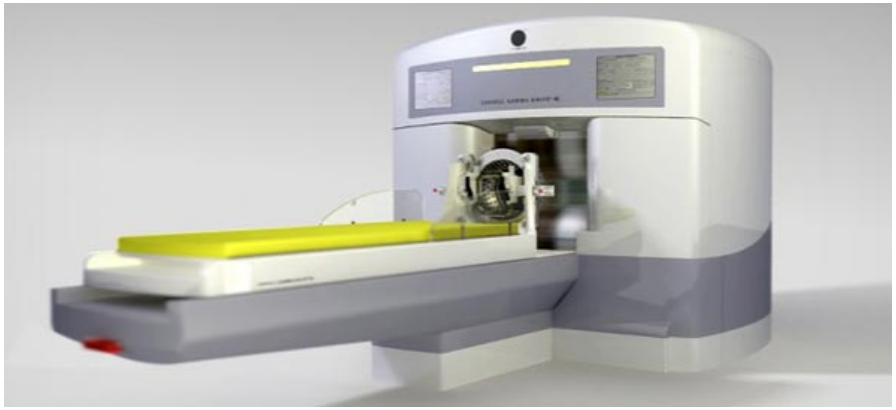
案例2：波音787失控

- 2015年5月，美国联邦航空管理局针对波音787型梦幻客机(Dreamliner)发出警告，称这架飞机系统中存在软件漏洞，可能会造成飞机在飞行途中突然切断所有电源，让飞行员失去对飞机系统的控制。
- 在实验室测试中，787型梦幻客机系统软件中存在的漏洞会每隔248天（大概8个月时间）出现一次。在经过很多天持续使用后，飞机上的主要发电机控制装置将会自行失控。如果这种情况发生在飞行途中或者起飞起降状态下，结果是非常害怕的。

案例2：波音787失控

- 幸运的是，有一种临时、最简单的方案，就是周期性关闭电源系统。根据波音提供的记录，作为常规维护工作的一部分，所有飞机都会定期关闭系统，因此现在没有“迫在眉睫”的危险。与此同时，波音正在修补此软件漏洞。

案例3：辐射治疗仪案例



- Therac-25是Atomic Energy of Canada Limited所生产的一种辐射治疗的机器。由于其软件设计时的瑕疵，致命地超过剂量设定导致在1985年六月到1987年一月之间产生多起医疗事故，5名患者死亡，多名患者严重辐射灼伤。这些事故是操作失误和软件缺陷共同造成的。
- Therac-25有两种电子束设置：低能量模式，可以直接照射病人；高能量模式，需要屏蔽一个X射线过滤镜。

原因分析

- 悲惨的是，用户界面和射线控制器之间用户界面设计存在竞争。一旦操作者选择了一种模式，机器就开始自我配置。如果操作者在8秒之内，撤销了前面的操作并选择其他不同的模式，系统的其他部分并不会接收到新的设置。因为，该机器需要8秒钟才能使磁针摇摆到位。因此，某些操作熟练、动作敏捷的操作者会不经意地增加病人的药量，而这些致命的药量造成了几个病人的死亡。
- 出错的情况是：
  - 1.操作人员首先错误选择了高能量模式（此时，机器将开始配置，而且机器配置是高级别任务，在配置完成的8秒钟内将不接受新命令）；
  - 2. 操作人员撤销前面的高能量模式选择，并选择另一种模式－低能量模式；
  - 3. 操作人员熟练到在8秒钟之内完成其它输入操作，并启动放射；
  - 4. 机器将仍按高能量方式照射病人，而不是操作人员重新输入的低能量模式。

案例4：大型票务系统瘫痪

- 2008年北京奥运会网络票务系统在开始运行不到半小时就彻底瘫痪。
  - 原因：票务系统设计的最高峰值为100万次/小时并发登录，而当天的官方网站访问量达800万次/小时，就连呼叫中心的访问量也超过380万次/小时。
- 2012年伦敦奥运会门票网上预订系统在最后一天的最后时刻因需求量太大而瘫痪。伦敦奥组委不得不临时把订票截止时间延长一个小时，改为伦敦当地时间27日凌晨1时。

案例5：一个消息框的案例

- 在某个公安局的数字录音系统上用到的Paradox数据库(Borland公司的产品)在处理超过10万条记录时总会出现莫名的错误，为了分析错误规律，程序员在程序中设置成：系统每向数据库增加一定数量的记录后，就弹出一个消息框，提醒自己去检查数据库；
- 但上线的应用中这个消息框并没有去掉…。于是，在弹出消息框后，程序就停在那里等待用户干预。而这又是一个无人值守的程序，它停止工作了，并没有人知道；
- 而在这段过程中，有人打110电话威胁要炸王府井百货；公安局长极其重视，带队来查录音，结果…..
- 再结果就是，数字录音系统被换成了国外的成熟系统，公司损失了一个100万单子。

案例6：许霆ATM案例

- 许霆（1983年出生），南下的打工小伙，2006年4月21日（周五）晚在广州商业银行的一个ATM机取款，原本卡上只有176.97元，想取100元，但多敲了一个“0”，结果ATM机真的吐了1000元出来，而且他查询后发现账上只扣了一元钱。随后，他连续取款171笔，合计17.5万元；
- 银行于4月24日（周一）进行设备例行检查时发现情况，30日报案；
- 2008年3月31日广州市中级人民法院二审判定“许霆犯盗窃罪，判处有期徒刑五年，并处罚金二万元”；（一审判的是无期，引起巨大的社会舆论）；
- 2010年7月31日，许霆因表现良好获假释。

原因分析

- 2006年4月21日17时许，运营商广州广达运通公司对涉案的自动柜员机进行系统升级。该自动柜员机在系统升级后出现异常，1000元以下(不含1000元)取款交易正常；1000元以上的取款交易，每取款1000元按1元形成交易报文向银行主机报送，即持卡人输入取款 1000元的指令，自动柜员机出钞1000元，但持卡人帐实际扣款1元。（摘自二审判决书）
- 且，许霆在170余次取款中间，还把情况告诉了他的一位郭姓工友，该位先生以农行卡同样在该柜面机上分多次取款19000元（按以上情况推测，广州商行以“本代他”的形式通知农行扣款19元）；
- 这是一个典型的软件缺陷，软件不经测试直接上线！

案例7：挪威网上银行转账案例

- 2006，某家挪威银行的网银行系统
- 一个老妇人通过网银向她女儿的账户转10万美元
- 她女儿的账号是：　　71581555022
- 老妇人输入的账号是：　71581555<u>5</u>022　　（多输入了一个数字“5”）
- 挪威标准的账号是11位长，最后1位是校验和。网银系统在两次接受到同样12位账号的情况下，自动将账号截尾，也即只使用前11位数字。 巧的是，这个截尾后的11位数字正好能通过账号的验证规则，更巧的是这个数字是一个酒鬼的账号；
- 10万美元被酒鬼挥霍掉了，老妇人要求银行赔偿，理由是：“since she typed in 12 digits, it was the responsibility of the system to give her an error message instead of just dropping all digits after the 11th.”。银行拒绝赔偿，理由是：“she hit the confirm button”；
- 在一个由五人组成的银行业消费者仲裁委员会裁定中，2：3，老妇人败诉，理由是：“She made an error and has to take responsibility.”。老妇人随后将银行告上了法庭，最终结果未知。

原因分析

- 系统对于不符合要求的输入情况自动进行了处理，未给客户以必要的提示；
- 一般情况下，误输的卡号是不符合卡号规则的，但例外情况总是会存在。

案例8：千年虫问题

- 上世纪70年代，所使用的计算机存储空间很小，这就迫使程序员在开发工资系统时尽量节省存储空间，一个简单的方法是在存储日期时，只存储2位，如1974存储为74。
- 例如：银行在计算利息时，是用现在的日期如“2000年1月1日”减去客户当时的存款日期如“1974年1月1日”，如果年利息为3%，那么每100元银行应付给客户78元的利息。如果年份存储问题没有得到纠正，其存款年数就变为－74年，客户反而应付给银行利息。
- 开发者认为在20多年内程序肯定会更新或升级，而且眼前的任务比计划遥不可及的未来更加重要。为此，全世界付出了数亿美元的代价来更换或升级类似程序以解决千年虫问题，特别是金融、保险、军事、科学、商务等领域，花费大量的人力、物力对现有的各种各样程序进行检查、修改和更新。

案例9：Intel奔腾芯片缺陷

- 在PC机的“计算器”中输入以下算式：
- （4195835／3145727）× 3145727-4195835 = 0 ？
- 带有浮点除法软件缺陷的老式Intel奔腾处理器。
- 这种情况很少出现，仅在精度要求很高的数学、科学和工程计算中才会出现。
- Intel公司为自己处理软件缺陷的行为道歉并拿出4亿多美元来支付更换芯片的费用。

案例1

Linux glibc幽灵漏洞检测及修复方案

发表于 2015-01-29 17:49 | 2982次阅读 | 来源 yunsuo.com | 0 条评论 | 作者 yunsuo.com

云锁 linux 漏洞

摘要：昨日Linux glibc库曝出高危缓冲区溢出漏洞GHOST(幽灵)，漏洞CVE编号为CVE-2015-0235，通过该漏洞，攻击者可以远程获取linux服务器的最高控制权限。

昨日Linux glibc库曝出高危缓冲区溢出漏洞GHOST(幽灵)，漏洞CVE编号为CVE-2015-0235，通过该漏洞，攻击者可以远程获取linux服务器的最高控制权限。glibc是linux系统中最底层的API，几乎其它运行库都会依赖于glibc，因此该漏洞的危害巨大，众多linux发行版本将受影响。漏洞发现者已经利用该漏洞，成功远程获取了一台邮件服务器的最高权限，并称该漏洞将会有更大的影响，Redhat在昨日发布的紧急安全通告里，也将该漏洞定义为“高危”。

案例10：Linux幽灵

- 受影响操作系统版本
  - CentOS 6 、7
  - Debian 7
  - Red Hat Enterprise Linux 6 、 7
  - Ubuntu 10.04 、12.04等众多使用glibc库2.2-2.17版本的Linux发行版本