基于消息变异的Web服务脆弱性测试系统的设计与实现

陈加梅 陈锦富 詹永照 王环环 李 青
(江苏大学计算机科学与通信工程学院 镇江 212013)

摘 要 研制自动化的Web服务脆弱性测试工具对基于Web服务的软件工程有重大影响,并能提高软件的安全性和可靠性,是当前软件行业一个有意义的研究课题。针对广泛使用的Web服务,设计和实现了一个测试Web服务脆弱性的原型系统WSVTS(Web Service Vulnerability Testing System)。根据SOAP消息参数的个数和类型,实现了两种基于SOAP消息变异的Web服务脆弱性测试方法,分别是最坏差异输入变异方法(The Worst-input Mutation Approach)和杂乱数据变异方法(Fuzz Data-input Mutation Approach)。测试系统融合这两种测试方法,实现了两种测试用例生成算法,分别是最远邻测试用例生成算法TCFN(Test Cases generation based on Farthest Neighbor)和杂乱数据输入变异算法FDMA(Fuzz Data-input Mutation Algorithm),然后,将算法产生的测试用例作用于SOAP请求消息,从客户端观察应答消息,来分析Web服务的脆弱性。

关键词 Web服务,SOAP消息,脆弱性测试,测试用例,变异算子,原型系统
中图法分类号 TP311.5 **文献标识码** A

Design and Implementation of Web Service Vulnerability Testing System Based on SOAP Messages Mutation

CHEN Jia-mei CHEN Jin-fu ZHAN Yong-zhao WANG Huan-huan LI Qing
(School of Computer Science & Telecommunication Engineering, Jiangsu University, Zhenjiang 212013, China)

Abstract The automatic tool of testing Web service vulnerability brings great effect on Web service-based software engineering, and they can effectively ensure the security and reliability of Web service-based software. According to Web service which is used widely, a prototype system WSVTS(Web Service Vulnerability Testing System) was designed and implemented. Two mutation approaches of testing Web service vulnerability based on the input domain of SOAP message, namely the worst-input mutation approach and fuzz data-input mutation approach, were implemented. Based on the two approaches, two test cases generation algorithms which are Test Cases generation based on Farthest Neighbor (TCFN) and Fuzz Data-input Mutation Algorithm (FDMA) were also implemented. Then, the test cases generated by the algorithms were executed in the SOAP requesting message. The vulnerability of the Web services can be detected by the response message of the client.

Keywords Web service, SOAP messages, Vulnerability testing, Test cases, Mutation operators, Prototype system

软件测试的研究应从"理论+技术+辅助工具+管理"4个侧面展开,且它们之间互相依存,缺一不可。除了探索测试基本理论、发明新型测试技术、优化测试过程管理外,研制高效、便捷和可视化的辅助测试工具(平台)尤为重要[1]。

WS(Web Service)服务因具有互联互通、高度的可集成性受到广泛的应用,但WS的脆弱性制约了其发展,WS中存在着安全漏洞,利用这些漏洞会危害到WS的安全策略,导致重要信息丢失或出错,因此对WS进行脆弱性测试是保障WS的有效手段。针对WS的测试主要有自动化测试和手工测试两种形式,自动化测试与人工测试相比:执行速度更快,大大提高了测试效率;具有一致性和更高的可靠性,每次测试脚本在运行时执行同样的操作,减少了人为的错误;可以重复使用测试脚本,避免人力的浪费;替代人工测试的困难,对于人工不可为的大量测试数据的测试,自动化测试可以高效、准确完成。通过以上的比较,可以看出自动化工具对WS脆弱性测试是至关重要的。目前针对SOAP消息进行测试的自动化工具主要有SOAPUI和SMAT-WS[2]。

SOAPUI是由标准的Java Swing开发的一个GUI自动化测试工具,它可以说是JUnit测试框架的扩展和衍生[3]。它通过SOAP/HTTP来检查、调用、实现Web服务间的负载、符合性、功能测试,可以作为一个桌面应用软件,也可以使用插件集成到netbeans 、Eclipse、maven2. X和intellij中使用[4]。SOAPUI具有优点,也有不足之处,SOAPUI使用的Web服务客户端是自己的,它创建测试消息的过程和基于

到稿日期:2012-09-11 返修日期:2013-01-03 本文受国家自然科学基金项目(61202110,61063013),教育部博士点专项基金项目(20103227120005),江苏省自然科学基金项目(BK2012284)资助。

陈加梅(1987—),女,硕士生,主要研究方向为软件测试;陈锦富(1978—),男,博士,副教授,主要研究方向为软件测试、信息安全,E-mail:jinfuchen@ujs.edu.cn;詹永照(1962—),男,教授,博士生导师,主要研究方向为计算机图形学、人机交互、模式识别等。

JAX或者其他的Web服务客户端的工作过程不同，SOAPUI不从WSDL中产生Java类，并且它不处理Java对象的序列化和反序列化，这将导致客户端并不像真正的Web服务消费者那样去调用Web服务[5]。

SMAT-WS是一个基于Java的测试工具，测试者可以利用它分析SOAP消息并检查消息的合法性。它还提供了一系列的变异算子，支持数据库存储实验结果，为用户收集结果并形成报告，但SMAT-WS没有提出测试用例的生成算法。

为弥补以上两个工具的不足，我们设计实现了一个工具：WSVTS。WSVTS是基于最坏差异输入变异和杂乱数据输入变异两种变异方法与两种测试用例生成算法TCFN和FDMA设计实现的一个Web服务脆弱性的原型测试系统。与其他SOAP消息测试工具相比，WSVTS针对SOAP消息的特点对变异算子进行扩展，使得变异操作更加全面，根据SOAP消息的参数个数和类型调用相应的测试用例生成算法来自动产生测试用例，在发现故障方面更有针对性。

1 WSVTS系统总体框架

WSVTS是在Windows平台上用Visual C#开发的针对SOAP消息变异的测试系统原型，系统测试框架主要包括4部分，分别是SOAP消息生成器模块、SOAP消息变异模块、测试用例生成器模块和安全分析模块。其整体的功能如下：

(1)扫描Web服务描述文件(WSDL文件)，并通过动态解析接口的WSDL文件自动生成SOAP消息；

(2)对SOAP消息的参数个数和参数类型等信息进行变异；

(3)调用相应的测试用例生成算法来产生测试用例；

(4)对Web服务进行脆弱性测试，由安全分析模块对Web服务进行安全性分析，并给出测试报告。

Web服务脆弱性测试系统总体架构如图1所示。

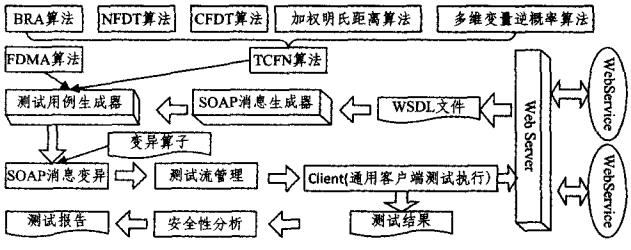

图1 WSVTS系统总体架构

1.1 SOAP消息生成器模块

SOAP消息生成器模块的输入是被测Web服务的WSDL文件，包括所有请求和响应消息的数据类型、传输协议、访问Web服务的地址等信息，输出为访问Web Service的普通SOAP消息。实现步骤是：首先使用.Net Framework命名空间System.Web.Service.Description里面的相关类对WSDL文件进行解析。此命名空间里的每个类与WSDL规范里的每个元素是对应的，并且类的层次结构与WSDL规范对应的XML文件结构也是相互对应的，解析出来的所有信息都保存在Service Description类中；然后采用.NET反射技术和动态编译技术调用代理类中的方法解析出SOAP消息，通过CompilerResults的CompileAssembly()得到代理类所在的程序集Assembly；再通过System.Reflection命名空间的Assembly类的GetType()方法得到与服务名相同的类的Type对象，再调用Type的GetMethods()方法可以读取SOAP消息包含的所有方法，使用GetParameters()方法可以读取SOAP消息的参数名称；最后将取出来的方法和参数名称暂时存放在System.Collections命名空间的Hashtable类的哈希表中，用GetType()方法得到参数的类型。

1.2 测试用例生成器模块

将引发SOAP消息失效的输入点的分布域划分成两种情况，分别是连续型和离散型区域。其中，离散型主要是点状失效域；连续型主要是块状和带状失效域。在二维输入域中相应的失效区域类型[6]如图2所示。

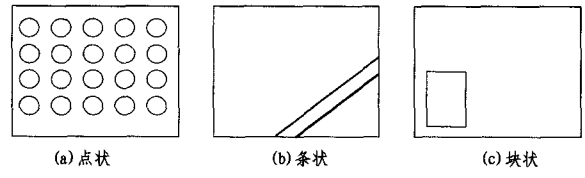

图2 SOAP消息输入域类型

当失效区域为非点状区域(连续型)时，让测试用例均匀分布在输入域中能更快地引发Web服务的安全性问题。基于该思想，系统使用了最坏差异输入变异方法对Web服务进行脆弱性测试。相应地，当失效区域为点状区域(离散型)时，系统使用了杂乱数据输入变异方法。

1.2.1 测试用生成算法

生成最远邻测试用例算法——TCFN(Test Cases generation based on Farthest Neighbor)算法。TCFN算法的主要思想是当某测试用例不能引发程序失效时，其周围的测试用例能引发程序失效的概率也比较低[7]。

首先根据参数个数的不同选择相应的测试用例生成算法，然后在每个子区域中随机生成符合要求的测试用例，从中挑选出离所有待测试用例距离最大的作为下一个测试用例对SOAP消息进行变异测试。TCFN算法包含6个子算法：BRA(Bit Reverse Algorithm)算法、ResStr算法、NFDT算法、CFDT算法、加权明氏距离算法和多维变量逆概率分布算法，后两个算法适用于参数为3个的情况。TCFN算法步骤：输入WSDL地址并解析出SOAP消息，提取SOAP消息参数的个数，如果含有一个参数，若参数的类型为数值型，则调用按位翻转BRA算法，采用PFB变异算子对其进行按位翻转操作来产生下一个最远邻的测试用例；若为字符串类型，则可调用字符串翻转算法ResStr并且结合使用CIV算子来实现。如果SOAP消息含有两个参数，则根据输入域取值的类型来求下一个最远邻测试用例，如果为rec类型则调用NFDT算法，如果为cir类型则调用CFDT算法，如果为cur类型则调用Derivation方法；如果SOAP消息含有多个参数，则根据多维变量逆概率输入分布函数算法或加权明氏距离[8]进行计算。

系统使用了杂乱数据输入变异方法，它主要是针对参数的输入域为离散型的情况来展开的。受Fuzz测试方法的启发，系统使用了自动分类生成合法测试用例和非法测试用例的思想，根据不同的待测试对象，采用不同的随机函数生成测试用例[9]。在进行杂乱数据输入变异测试时，从两个角度进行开展：一是针对无效等价类，在完全不遵循SOAP请求消息输入长度、格式等限制的前提下设计杂乱的测试用例；二是针对有效等价类，在符合基本约束的前提下设计无意义的输

入。系统使用了 FDMA(Fuzz Data-input Mutation Algorithm)算法,算法步骤:由 WSDL 地址解析得到 SOAP 消息并提取 SOAP 消息参数后,每个参数根据不同的参数类型采用相应的变异算子进行变异来产生离散集,参数类型主要有数值型、字符串类型、对象型。若参数类型为对象型,给对象的每个域赋随机值,并结合变异算子 DNS 对每个域做增加删除节点处理;若为数值型数据,调用平均分布随机数算法 ADFA(Average Distribution Function Algorithm)生成一个随机数,采用算子 IIV 生成边界值测试用例;若参数为字符串型,将随机字符串利用变异算子 FVS 和算子 LSV 生成格式化字符串和超长字符串,将这些参数的测试用例合并,生成最终的测试用例。

1.2.2 测试用例生成器模块

测试用例生成器模块给用户提供了友好的接口,测试者根据 SOAP 消息生成器解析出的 SOAP 消息的参数个数来调用不同的测试用例生成算法。当参数个数为 1 时,第一个测试用例根据参数类型随机手动输入,然后调用相应的测试用例算法来产生其它的测试用例。其中 BRA 算法适用于参数类型为数值型的数据;ResStr 算法用于参数类型为字符串类型的数据。同理,当参数个数为 2 时,首先判断参数的类型,然后调用 NFDT(the Next Farthest Distance Test case)或 CFDT(Circle Farthest Distance Test case)算法。CFDT 算法主要是根据 T. Y. Chen 等人[10]提出的思想设计一个排除区域半径,在这个半径之外寻找一个相对较远的距离来产生测试用例,最后将产生的测试用例保存到测试用例文件中。其中,NFDT 算法用 soap 类来实现,这个类中包含 5 个方法,依次为 distance()、sort()、wenjian()、random()和 research(),其分别实现了距离计算、排序、文件操作、随机数生成和改进的查找算法 5 个功能,同时它们几个又互相协作,共同实现了测试用例的生成过程。CFDT 算法用 circle 类来实现,它包括 distance()和 random()两个方法,其分别实现了距离计算和随机测试用例生成功能。当参数个数为多个时,根据前面提到的加权明氏距离算法[8]或多维逆概率分布算法产生测试用例,在此没有进行具体的实验。在针对离散型数据进行边界值输入测试的过程中,主要通过 FDMA 算法调用相应 ADFA 算法、变异算子或组合算法来产生测试用例。

1.3 SOAP 消息变异模块

利用 SOAP 消息生成器模块产生的参数个数和类型、测试用例生成器模块产生的测试用例,再结合系统使用的针对不同类型的变异算子,就可以对 SOAP 消息进行变异测试。在进行消息变异时,首先根据 Web 服务的 URL 地址寻找对应的服务,解析出 SOAP 消息,根据消息的类型,用生成的测试用例进行测试,将测试结果保存在测试用例文件中;然后选择相应的变异算子将 SOAP 消息的参数值进行变异,重新进行测试,将产生的结果保存在另一个测试用例文件中,这样就实现了针对一个 Web 服务的测试过程。完成整个过程使用了命名 System. Web. Services 里面的类,主要有 WebClient、HttpWebRequest、HttpWebResponse、ServiceDescription,还有 System. IO 命名空间里的 StreamReader 类。其中 WebClient 用来实现向 URI 发送数据和从 URI 接收数据。HttpWebRequest 类和 HttpWebResponse 类用于发送和接收 HTTP 数据。HttpWebRequet 类是通过 Create()方法来创建的,把 HTTP 响应的数据流(stream)绑定到一个 StreamReader 对象,然后通过 ReadToEnd()方法把整个 HTTP 响应作为一个字符串返回。

1.4 安全分析模块

安全分析模块的主要功能是首先根据 Web 服务的安全需求说明书进行安全性分析,然后产生测试报告。此模块分为两个部分,一是安全异常,一是发现错误数。对于一般的 Web 服务而言,安全异常主要分为两种情况:(1)在异常日志中明确标出的异常方法和代码;(2)运行状态违反了 Web 服务安全需求说明书的约束条件但没有触发异常。根据前后两次对同一个 Web 服务测试进行比较的结果,分析该 Web 服务是否存在脆弱性问题。安全异常部分实现的方法是如果同一个测试用例变异前后返回的消息不相同,就说明此 Web 服务存在脆弱性问题;然后采用微软的. NET 异常捕获技术来捕获出错信息;最后根据相关异常信息来识别 Web 服务是否存在脆弱性。"发现错误数"选项卡记录了每次测试时生成的测试用例的总数以及执行变异测试后成功的测试用例数。实现的方法是先调用测试用例生成算法产生测试用例;然后遍历存放测试用例的文件,计算出生成的无重复的测试用例总数;最后执行测试,得到成功的测试用例的个数。

2 WSVTS 系统数据流

测试开始时先导入 WSDL 文档,根据 WSDL 文档先创建测试工程,再启动测试用例生成模块,调用测试用例生成算法来生成符合规范的测试用例,然后根据需要可以对测试用例进行编辑、修改来增加或删除其长度。在创建一个具体的测试工程以后,如果需要修改测试用例和测试流,可以分别对测试用例编辑模块、测试流编辑模块进行相应的操作来实现对测试用例和测试流的编辑。当打开测试用例编辑菜单时,界面上就会显示测试用例中的各个元素属性的类型以及相应的实例值,根据界面显示的类型说明,尽可能地输入可能产生错入的实例值(边界值),点击界面上的"Save as"按钮将其存储为新的测试用例;也可以从测试用例生成模块选择测试用例,然后对其进行格式化字符串操作、插入字符串分隔符或字符翻译等操作,再点击界面上的"Save as"按钮将其存储为新的测试用例。为了能够批量执行测试用例,设计了测试流编辑模块,按照测试用例执行的逻辑顺序,一步一步选择下一个测试用例,同时根据测试用例的输入输出依赖关系,填写上一个测试用例的输出结果和下一个测试用例的输入参数之间的依赖关系,并将其存为一个文件,就形成了一个测试流。系统对应的数据流图如图 3 所示。

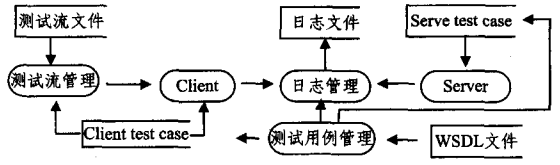

图 3 数据流图

3 系统实现及测试环境

3.1 系统实现

原型系统 WSVTS 界面采用了微软的窗口浮动技术,将

界面分成几个模块，每块根据需要添加相应的控制条来实现。窗口浮动技术可以实现将一个窗体里的控件拖动到任意的位置。它主要包括 DockPanel 类、DockWindow 类、DockPane 类和 DockContent 类 4 部分。其中，Dockpanel 是 Docking 的源头，它继承自 System. Windows. Forms. Panel 类，当 DockPanel 填充整个 MainForm 后，其他所有的从 DockContent 继承来的 Form 调用 Show() 函数时，DockPanel 类就可以自动管理这个附加的窗体。

设计本系统界面的基本思路是首先创建一个多文档窗体，将其 IsMdiContainer 属性设置为 true，并设置 DockPanel 的 DocumentStyle：DockPanel. DocumentStyle = DocumentStyle. DockingMdi，然后采用窗口停靠技术实现界面的不同位置的停靠。

设计的基本步骤如下：

(1)首先加载 WeifenLuo. WinFormsUI. Docking. dll 动态链接库。

(2)在主窗体中添加 DockPanel 控件，并将此属性 IsMdiContaine 设置为 true，同时设置 dockpanel 的属性 documentstyle 为 DocumentStyle. DockingMdi。

(3)加载 Dockpanel 的配置文件。Dockpanel suite 是基于配置文件的，其配置文件 Dockpanel. config 可以放置到指定的位置。

(4)浮动窗口定位。当进行子窗口加载时，要把它放在 DockPanel 中的某一个位置，其中 DockStyle 共有 Bottom、Fill、Left、None、Right、Top 6 个属性。

(5)浮动窗体需要继承自 DockContent，为了保证在关闭某一浮动窗体之后再打开时能够在原位置显示，将窗体的 HideOnClose 属性设置为 true。

3.2 测试环境

开发工具采用 Visual Studio. NET 2008 平台和 C# 语言以及 Visual studio C++ 6.0。WSVTS 可测试已创建的任意 Web 服务，用户可以根据 SOAP 消息参数的个数和类型调用相应的测试用例生成算法，产生的测试用例保存在本地硬盘。用户可根据不同需求，选择不同的变异算子对 SOAP 消息进行变异，变异后结果保存在本地磁盘。完整的测试过程如图 4 所示。

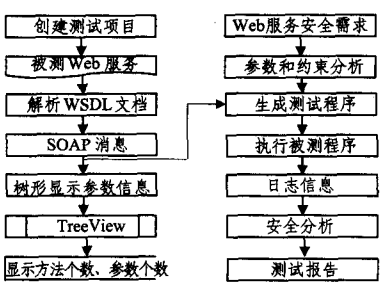

图 4 WSVTS 的测试过程

WSVTS 首先创建一个测试项目，选择待测 Web 服务，根据其 URL 地址获取 Web 服务的接口信息；通过解析 WSDL 协议得到相对应的 SOAP 消息格式，分析消息的参数个数和类型，调用相应的测试用例生成算法，然后将生成的测试作用于 SOAP 消息，观察客户端接收的响应消息，并进行安全性分析，最后得出测试报告。图 5 是 WSVTS 正常操作下的系统快照。

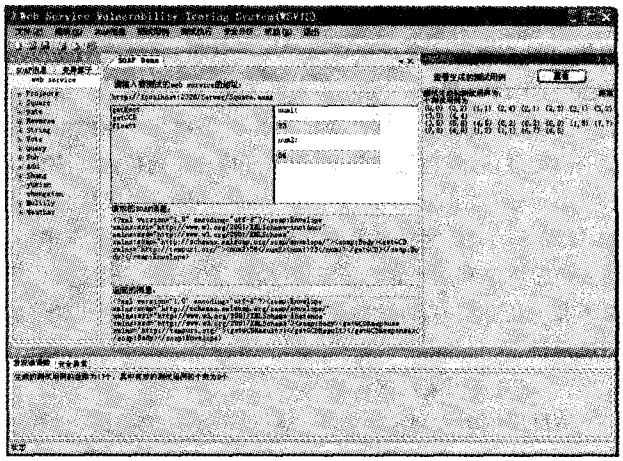

图 5 WSVTS 系统快照

4 系统测试结果分析

WSVTS 针对 18 个 Web 服务，借鉴我们在文献[11]中对构件的安全性研究的方法，再结合 Web 服务的特点，针对 SOAP 输入参数的类型设计了 15 种有效的变异算子，表 1 详细地描述了这些变异算子。

表 1 Web 服务脆弱性变异算子

编号	算子	描述	实例
01	SVB	将节点 n 的值用空格代替	改变 soap 参数值 n 为‘’
02	SVN	将节点 n 的值置空	使节点 n 为 NULL
03	IPO	插入参数运算符	对参数插入绝对值符、取反、倒置或平方等
04	DNS	删除节点	从 SOAP 消息中删除节点及其子节点
05	FVS	格式化字符串值	“%n%n……(256 个字符)”，“%s%s……(1024 个字符)”等
06	IIV	整型非常规值	0，+/−(1，28−1，28，28+1，216−1，216，216+1，232−1，232，232+1，264−1，264)等
07	FIV	浮点型非常规值	0，1，−1，+/−(最大浮点数+/−1)，+/−(最小浮点数+/−1)，5E-324，1.7E+308，pi，e 等
08	CIV	字符型非常规值	Null，‘a’，‘z’，‘A’，‘Z’，等特殊字符：‘{’，‘(’，‘[’，‘\n’，‘\0’，‘\s’，‘\d’等
09	EOV	交换节点值的顺序	在 SOAP 节点中交换节点值的顺序
10	EON	交换 SOAP 参数顺序	在 SOAP 节点中交换几个参数的顺序
11	RSV	随机非常规字符串值	如转义字符组成的串“\e\n\r\d\x\x”，“\xff\xfe\x00\x01\x42\xb5\nnnn\h9cc…”
12	LSV	超长字符串值	使用 N 位字符串函数 Generate String(int n)生成如：“AAA……(256 个字符)”，“AAA……(1024 个字符)”，“AAA……(15000 个字符)”等
13	UVF	URL 及文件路径字符串值	http://… ddddddeeeeerrttttt；“C://system32//Notepad. EON”，“H:\ABC\killvirus. EON”，“D:\AA. EONEON”
14	SSI	SQL 字符串注入	“a or 1=1”，“delete”，“drop table users”，“sql attempt5−−”等
15	PFB	SOAP 参数翻转	使用函数 flip，flipBit 翻转指定 SOAP 参数或指定位

考虑到变异算子的有效性问题，用公式 $OE=EF/TC$ 定义变异算子的效率，其中 EF 表示发现的错误数；TC 表示变

(下转第 186 页)

[4] Sarma A D, Theobald M, Widom J. Exploiting lineage for confidence computation in uncertain and probabilistic databases[A]// Proceedings of the 24th IEEE International Conference on Data Engineering[C]. Washington, DC: IEEE Compu ter Society Cancun, 2008: 1023-1032

[5] 王永利,钱江波,等. 一种 REID 数据不确定性的自适应度量算法[J]. 电子学报,2011,39(3):579-584

[6] Christopher Re, Letchner J, Balazinksa M, et al. Event queries on correlated probabilistic streams[A]// Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data[C]. New York, NY: ACM, 2008: 715-728

[7] Fang Zheng, Tong Guo-feng, Xu Xin-he. Particle swarm optimized particle filter[J]. Control and Decision, 2007, 22(3): 273-277

[8] Shawn R J, Minos G, Michael J F. Adaptive cleaning for RFID data streams[A]// Proceeding s of the 32nd International Conference on Very Large Data Bases (VLD B06) [C]. Seoul: VLDB Endowment, 2006: 167-174

[9] Gordon N J, Salmond D J, Smith A F M. Novel approach to nonlinear/ non gaussian bayesian state estimation[J]. IEE Proceedings F In Radar and Signal Processing, 2002, 140(2): 107- 113

[10] Wu Z B, Chen Y H. The maximizing deviation method for group multiple attribute decision making under linguistic environment [J]. Fuzzy Sets and Systems, 2007, 158(14): 1608-1617

(上接第 146 页)

异算子产生的测试用例的总数。图 6 给出各变异算子的有效率,可以看出 FVS 和 IIV 变异算子比其它的变异算子发现错误的数量多,说明它发现错误的效率比较高,对大多数的 Web 服务而言,这两种变异算子是通用的;其次是 SVB 变异算子,它也是针对大部分的数据来产生测试用例,这种变异算子在现实中也是非常实用的;有效性最不理想的是 SSI 算子,因为它的适用范围比较窄,只适用于某个特定范围。系统使用的变异算子大部分可以较好地发现错误,只是某种变异算子针对某种参数类型有更好的效用。从实验结果可以看出,变异算子总体的有效率约为 24%,验证了测试用例生成算法的可行性和有效性。

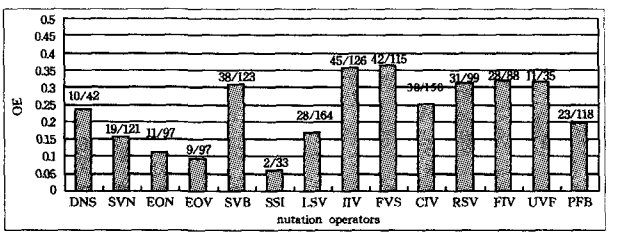

图 6 各变异算子的有效率

结束语 在基于 Web Service 的软件的开发和维护活动中,测试 Web Service 的脆弱性是至关重要的一环。研制针对 WS 脆弱性的自动化、半自动化的测试工具是 Web Service 亟待解决的课题。目前针对脆弱性的测试工具不多,主要有 SOAPUI 和 SMAT-WS,与这两类工具相比,WSVTS 测试工具原型具有如下优点:

(1)不同于 SOAPUI 的数据手工输入,WSVTS 的测试数据通过算法自动生成,WSVTS 基于 SOAP 消息参数的个数和类型实现了两种测试用例生成算法 TCFN 和 FDMA,所生成的测试用例具有很高的故障检测能力。

(2)实现了 15 种针对 SOAP 消息的变异算子,并通过实验验证了变异算子的有效率。

(3)工具的自动化程度较高,只需要少量的人工参与。

(4)具备较全面的功能模块,包括 SOAP 消息生成器模块、测试用例生成器模块、SOAP 消息变异模块、测试流管理模块、测试执行模块及安全分析模块等,几个模块协同工作完成测试过程。

(5)测试工程以项目形式管理,有利于测试的组织和管理,可以重现测试活动,得到的测试报告可以进行对比和分析。

参 考 文 献

[1] 陈锦富,卢炎生,谢晓东,等. 一个组件安全自动化测试平台的设计与实现[J]. 计算机科学,2008,35(12):229-233

[2] Lourival F, de Almeida J, Vergilio S R. Exploring Perturbation Based Testing for Web Services[C]// ICWS 2006. IEEE Computer Society, Los Alamitos, 2006: 717-726

[3] The Eviware SOAPUI 官方网站[EB/OL]. http://www.SOAPUI.org/2007

[4] Sourceforge Org[EB/OL]. http://sourceforge.net/forum/

[5] 罗作民,朱燕,程明. Web 服务测试工具 SOAPUI 及其分析[J]. 计算机应用和软件,2010,27(5):155-157

[6] Chen T Y, Eddy G, et al. Adaptive Random Testing Through Dynamic Partitioning[C]// Proceedings of the Fourth International Conference on Quality Software. 2004: 79-86

[7] Chen T Y, Leung H, Mak I K. Adaptive Random Testing[J]. LNCS, 2004, 3321: 320-329

[8] 李博涵,郝忠孝. 反向最远邻的有效过滤和查询算法[J]. 小型微型计算机系统,2009,30(10):1948-1951

[9] Kim H C, Choi Y H, Lee D H. Efficient File Fuzz Testing Using Automated Analysis of Binary File Format[J]. Journal of Systems Architecture, 2011, 57(3): 259-268

[10] Chan K P, Chen T Y, Towey D. Normalized Restricted Random Testing [C]// Springer-Verlag 2003, 2655: 368-381

[11] 陈锦富,卢炎生,谢晓东. 一种采用接口错误注入的构件安全性测试方法[J]. 小型微型计算机系统,2010,31(6):1090-1096