基于信任容错的 Web 服务可靠性增强方法研究

杨墨[1]，王丽娜[1,2]

（1. 武汉大学 计算机学院，湖北 武汉 430072；2. 武汉大学 空天信息安全与可信计算教育部重点实验室，湖北 武汉 430072）

摘 要：提出了一种基于信任容错的 Web 服务可靠性增强方法，采用选举协议发现服务的隐式错误；设计了服务信任轮询检测机制；建立的信任机制度量提供者的可靠性；针对提供者的领域特点和请求者的可靠性需求设计信任感知的容错服务个数计算方法和冗余服务选择算法；实验验证该方法对恶意攻击具有较好抵御能力。

关键词：容错；信任模型；选举协议；Web 服务

中图分类号：TP302.8 **文献标识码**：B **文章编号**：1000-436X(2010)09-0131-08

Research of Web service reliability enhancement method based on trust fault tolerant

YANG Mo[1], WANG Li-na[1,2]

(1. School of Computer, Wuhan University, Wuhan 430072, China;

2. Key Laboratory of Aerospace Information Security and Trust Computing, Ministry of Education, Wuhan University, Wuhan 430072, China;)

Abstract: A Web service reliability enhancement method based on trust fault tolerant was proposed. Non-evident errors are detected by using voting protocol. A service trust polling detection mechanism was designed. Trust mechanism was built to measure the reliability of the providers. A fault-tolerant-number calculating formula was deduced according to the characteristic of the required domain and the non-functional requirement and a fault tolerant services selection algorithm was proposed. Experiments show that this method has well resistance to malicious attacks.

Key words: fault tolerant; trust model; voting protocol; Web service

1 引言

随着 Internet 的应用，系统表现为由多个软件服务组成的动态协作网络，开始呈现出一种柔性可演化、连续反应式、多目标适应的新形态。软件实体以基本（原子）服务的形式存在，并通过协同机制进行跨网络的互连、互通、协作和联盟，进而构成更复杂的组合服务[1,2]。在服务交互协同中，请求服务的主体（以下简称请求者）和提供服务的主体(以下简称提供者)[3]之间并不存在确定的信任关系。因此，如何确保应用系统的可靠性成为一个需要解决的问题。

另一方面，网络中大量存在功能相同或相似而平台和实现相异的服务。换而言之，冗余和多样性是现有服务的固有属性[4,5]。据此，学者们提出了各种容错的算法，试图利用冗余避免单个服务错误导

收稿日期：2010-05-13；**修回日期**：2010-07-23

基金项目：国家自然基金可信软件重大研究计划项目（90718006）；国家高技术研究发展计划（“863”计划）基金资助项目（2009AA01Z442，2008AA01Z404）；国家重点基础研究发展计划（“973”计划)基金资助项目（2007CB310801）

Foundation Items: The National Natural Science Foundation of China (Major Research Plan of Trust Software) (90718006); The National High Technology Research and Development Program of China (863 Program) (2009AA01Z442, 2008AA01Z404); The National Basic Research Program of China (973 Program) (2007CB310801)

致的整个系统的失效和故障[6,7]。

FAWS 提供了 Web 平台失效检测及日志等容错机制，采用主副服务的方法实现用户透明的服务访问机制，但没有机制保证服务的可用性[8]。

Transparent Fault-Tolerant Web Service 修改操作系统及服务器内核建立客户透明的服务访问机制，但该方法需要修改操作系统，通用性较差[9]。

FTWeb 服务容错平台通过扩展基本的 Web 服务运行环境实现容错。但它没有针对服务可用性的特点组合服务，失效检测通过超时机制实现[10]。

FT-SOAP 通过扩展 WSDL，引入服务组，采用被动复制模式实现容错功能。但没有考虑到服务实现的自治性，失效检测也是通过超时机制实现[11]。

服务网络是一个动态变化、相互联系却又难以精确预测的复杂信息环境[12]。以上的研究很少考虑到网络的动态特点，失效检测机制简单，没有根据领域特点对容错实施动态调节。

本文针对以上特点，提出一种基于信任容错的服务可靠性增强方法，利用选举协议(voting protocol)[13]设计服务失效检测，提出了服务信任轮询检测机制；建立信任机制度量提供者的可靠性；依据提供者的领域特点和请求者的可靠性需求设计信任感知的容错服务个数计算方法和冗余服务选择算法；实验验证本文的方法比单纯使用信任或容错更能抵御网络中存在的 3 类恶意攻击[14]。

本文的组织结构为：第 2 节简要描述基于信任容错的服务可靠性增强框架。第 3 节设计了服务失效检测机制。第 4 节阐述了容错服务选择算法。第 5 节做出了实验设计和结果分析。最后是结束语。

2 基于信任容错的服务可靠性增强框架

图 1 是基于信任容错的服务可靠性增强框架。其包含 4 个角色（提供者、请求者、服务选择模块和信任管理模块）和 2 个主要流程（容错服务选择流程——附加轮询和独立轮询流程）。

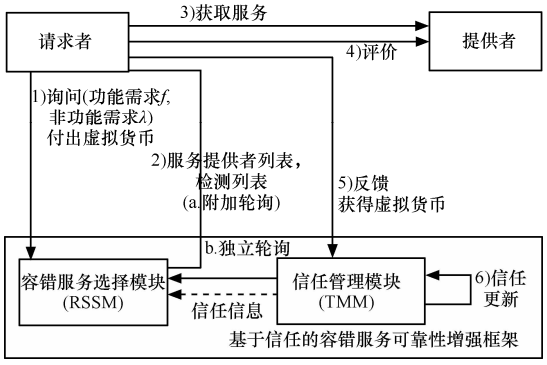


图 1 基于信任的容错服务可靠性增强框架

服务选择流程如下。

step1 当请求者需要使用某种服务时，询问容错服务选择模块 RSSM，给出其功能需求 f 和希望达到的可靠性指标（置信度 λ）。

step2 RSSM 根据 f 查找到相应的领域，综合考虑置信度 λ 和领域信任状况推选出适量的提供者和待测者，将列表返回给请求者。

step3 请求者连接各提供者，获取服务。

step4 在服务交互进行时或完成后，请求者根据选举协议获得服务的最终结果，并检测各个提供者的可靠性。

step5 请求者将检测结果作为反馈发还给信任管理模块 TMM。

step6 信任管理模块更新提供者的信任信息。

请求者需要在请求阶段付出一部分虚拟货币，并在反馈时获得略多的虚拟货币。具体虚拟货币的实现机制在本文中不做过多描述。

本方法构架了轮询机制以实现服务可靠性信息的更新，分为附加轮询和独立轮询。附加轮询整合到容错服务获取过程中，而独立轮询为信任模型主动发起的一次服务请求，用于刷新提供者的信任状态。轮询的具体算法见第 3 节。

本文的方法具有以下功能特点。

顾及所有领域和提供者。通过轮询机制能保证冷僻和低信任的提供者可靠性信息的新鲜性。

面向领域，用户偏好敏感的信任机制。当请求者对某一特定领域的查询增加时，该领域下所有提供者的可靠性信息的准确度也会相应的提高。

面向领域，需求驱动的选择算法。不同于已往的容错算法在设计阶段固定容错个数的策略，本方法在服务选择时，综合考虑请求者的非功能需求和领域当前的信任状态，能够较为准确的计算出完成当前服务任务所需容错服务的个数。

3 服务检测机制

3.1 非功能性需求

在 Web 环境中，请求者的非功能属性，一般包括系统的性能、可靠性、可维护性、可扩充性和对技术和对业务的适应性等。其中，提供者的可靠性最为重要。

可靠性：指在给定的时间内以及规定的环境条

件下，系统能完成所要求功能的概率。

而不可靠主要是由错误引起的。服务主要包含 2 个类型的错误：1) 显式的错误，并引发异常信息，解决方式为延时发送请求或者更改可替代的提供者；2) 隐式的错误，提供者返回错误的数据，不会立刻引发程序异常信息，但会导致程序产生不可预期的行为和结果。

3.2 基于选举协议的服务检测

基于选举协议的服务检测用于发现提供者的隐式错误。请求者从多个提供者处获取结果，只要它获得的正确结果足够多，则仍可以发现有效的服务结果。基于选举协议的检测算法如下。

step1 请求者和同一领域中的若干个提供者交互。

step2 请求者在一段时间 t 内获得的提供者有效返回数据集为 D。$D=\{d_1,d_2,\cdots,d_l\}, l\in N^+, l\leqslant rn$ 其中 rn 为容错提供者个数。当满足超过 $rn/2$ 的返回数据值相同或相似时请求者可以获得正确的服务任务结果 $result$。ε 为允许最大误差。

$\forall d_i,d_j\in D_a$,

if ($diff(d_i(value)-d_j(value))<\varepsilon$ and $|D_a|>\dfrac{rn}{2}$),

then $result=\text{avg}(D_a)$

step3 逐一比较最终结果 $result$ 和各提供者的结果 $value$，按照相识程度记为不同程度的正确。

step4 请求者将标记值作为反馈返回给信任管理模块。

3.3 轮询检测机制

本文使用信任度表示提供者的可靠性。而系统中实体的信任属性具有易失性，长时间不进行检测，其值可能不准确。

本文借鉴内存的轮询思路，提出了一套基于轮询机制的服务失效检测机制，并将实体信任的检测过程和对其的使用过程分离开来。假设在一个周期内，实体的信任程度不会有很大程度的改变。对信任程度高的实体的申请相当于 CPU 对内存的访问，访问过程可以作为刷新过程。而信任度低和长期得不到访问信任度的不确定性高的节点，类似于内存电容漏电导致的数据消失，需要轮询过程更新其信任状况。

本文设定 3 种信任状态，在服务和检测中分配 2 种不同的职能。

未知状态：处于未知状态的提供者（未知者）的信任度不确定或很低，因此需要对其进行检测。未知者只具有待测者职能。

可信状态：处于可信状态的提供者（可信者）必须长期保持很高的信任度。可信者只具有检测者职能。

混合状态：信任度大于未知者小于可信者的提供者处于混合状态（混合者）。混合者同时具有待测者和检测者职能。

检测者：检测者在轮询检测中提供服务，结果用于产生服务任务最终结果和检测待测者。

待测者：待测者在轮询检测中提供服务，但是其结果并不被用于产生服务任务最终结果，而是作为评判其信任状态的依据。

附加轮询算法如下。

step1 服务选择模块将 stn 个信任情况最久未更新的待测者加入检测列表返回给请求者。stn 为附加轮询待测者个数，其值根据领域的特点和请求频率按下面公式得到：

$$stn=\min\left(MaxStn,\left\lceil\frac{hn+tn}{Time_iFreq_i}\right\rceil\right)$$

其中，$Time_i$ 表示领域 i 中，提供者允许未被使用的最长时间。$Freq_i$ 表示领域 i 的查询频率。hn 和 tn 为领域中混合者和未知者个数。$MaxStn$ 为每次服务选择允许的最大轮询提供者个数

step2 请求者和所有提供者交互。利用基于选举协议的服务检测或得评测结果，并反馈给信任管理模块 TMM。

step3 信任模型使用检测结果更新待测者信任度。

对于长时间没有任何请求的冷僻的领域，信任管理模块需要发起一次独立轮询。算法如下。

step1 信任管理模块保持一个计数器。当 $TimeD$（领域允许未使用的最长时间）内没有任何对该领域的查询时，由 TMM 发出一个独立的轮询发起信号给服务选择模块，要求产生一个置信度为领域缺省值的服务选择过程。$TimeD$ 由以下公式获得

$$TimeD=\frac{MaxStnTime_i}{hn+tn}$$

step2 服务选择模块用虚拟货币招募请求者进行一次服务交互过程，待测者个数为 $MaxStn$。

step3 请求者和所有提供者交互。利用基于选举协议的服务检测或得评测结果，并反馈给信任管理模块 TMM。

step4 信任模型使用检测结果更新待测者信任度。

3.4 恶意攻击

服务检测机制用于发现和排除网络中的恶意提供者。

网络实体分为 2 类：善意实体和恶意实体。善意实体大部分时间能提供正确的服务结果。恶意实体提供虚假或恶意的服务,并且设法诋毁正常实体。本文假设客户的反馈大部分是正确的，讨论提供者的恶意行为。

独立恶意攻击：此类恶意提供者彼此没有联系，在与请求者的交互中提供虚假的结果或在结果中插入恶意代码。发生故障而提供错误结果的提供者具有以上的特点，同样视为恶意提供者。

复杂策略攻击：此类恶意提供者知晓一部分系统的信任机制信息，施展一套复杂的策略伪装为善意提供者：交替提供正确服务和施展恶意攻击，通过控制两者的比例使得自身的信任度在请求者可以接受的范围内波动，从而继续产生恶意行为。

协同恶意攻击：此类恶意攻击者可以看做独立恶意攻击者间形成了协同作弊的团体，通过策略上的合作，试图极力夸大同一团体中的同伙，并试图诋毁其他提供者。

以上 3 类恶意攻击具有代表性。事实上，恶意提供者可以实施同时实施以上多种恶意行为。

4 信任感知的容错服务选择

现有的容错算法一般在系统设计中设定容错服务个数为一个固定值。信任感知的容错服务选择算法针对动态变化的 Web 网络环境,考虑领域当前的信任状态，能够计算出保证请求者非功能需求的最小的容错数量和容错提供者列表。

4.1 置信度

可靠性置信度 λ（简称置信度）是请求者对服务任务能达到的可靠性的量化值的期望。

置信度的等级同服务领域相关。表 1 为服务可靠性和置信度等级的对应关系。容错服务选择模块中保持了一个服务类型和置信度取值的对应列表。表 2 为各个服务置信度的缺省设定，如果请求者没有提出置信度要求，则在容错个数计算时使用缺省值。而缺省值会随着请求者的置信度请求而动态改变。

表 1 服务可靠性和置信度等级的对应关系

赋值	等级	定义
[0.99，1)	高	可靠性非常重要。错误对业务造成的影响是破坏性的，不可补救
[0.95,0.99)	较高	错误对业务的影响极大，较难补救
[0.9,0.95)	中等	错误对业务的造成影响，但是可以补救
[0.8,0.9)	较低	错误对业务的影响很小，容易补救
(0,0.8)	低	错误对业务的影响可以忽略

表 2 服务置信度缺省值

服务类型	置信度	赋值
翻译	低	0.75
天气	较低	0.85
时间/日期	中等	0.9
压缩	高	0.97
存储	高	0.98
加密	极高	0.995
…	…	

4.2 容错提供者个数计算

为了在选举协议中请求者能获得正确的服务任务结果并能对检测各个提供者的正确性，提供者的服务结果必须有一半以上为正确。即

$$D_a \geqslant \max\left(\left\lceil \frac{rn+1}{2}, 2 \right\rceil\right)$$

rn 个服务提供者平均信任度为 $\overline{P}$，假设有 i 个提供者供应了正确服务。则 rn 中 i 个提供者提供正确服务的概率为 $C_{rn}^{i}\overline{P}^{i}(1-\overline{P})^{rn-i}$。则请求者能够根据多数原则从 rn 个服务提供者中达到服务任务的概率为

$$Af(rn,\overline{P}) = \sum_{i=\max\left(\left\lceil \frac{rn+1}{2} \right\rceil, 2\right)}^{rn} C_{rn}^{i}\overline{P}^{i}(1-\overline{P})^{rn-i}$$

这个取值需要大于等于此次服务选择请求的置信度，并且 rn 应该设置得尽可能的小。因此 rn 的取值应该满足下面的公式：

$$Af(rn,\overline{P}) \geqslant \lambda \text{ 并且 } Af(rn-1,\overline{P}) < \lambda$$

4.3 信任感知容错服务选择算法

以往的容错方法对服务的选取具有盲目性。而信任感知的容错服务选择方法根据领域的信任特点，优先从可信者中选取提供者。在可信者数目不足的情况下，仍然可以通过使用混合者达请求者的置信度要求。已下是信任感知容错服务选择算法。

step1 尝试在可信者中选择提供者。由于 rn 数量不会太大，使用穷举法获得容错数 rn，使其满足式(1)。

$$\begin{cases} Af(rn,\overline{P_a}) \geqslant \lambda \\ Af(rn-1,\overline{P_a}) < \lambda \end{cases} \tag{1}$$

其中，$\overline{P_a}$ 为可信者的平均信任度。

step2　如果容错数小于可信者个数 $rn \leqslant an$，即可信者足够满足用户非功能需求。则随机选取 rn 个可信者加入选择列表，转到 step 5。

step3　可信者不够，需要降低标准，容忍一部分错误，同时使用可信者和混合者提供服务。使用穷举法获得所需混合者数量 shn，使其满足式(2)

$$\begin{cases} \sum\limits_{i=\max\left(\left\lceil\frac{an+shn+1}{2}\right\rceil,2\right)}^{an+shn}\ \sum\limits_{j=0,j\leqslant i-an}^{shn} C_{shn}^{j}C_{an}^{i-j}\overline{P_h}^{j}(1-\overline{P_h})^{shn-i}\overline{P_a}^{i-j}(1-\overline{P_a})^{shn+j-i} \geqslant \lambda \\ \sum\limits_{i=\max\left(\left\lceil\frac{an+shn}{2}\right\rceil,2\right)}^{an+shn-1}\ \sum\limits_{j=0,j\leqslant i-an}^{shn-1} C_{shn-1}^{j}C_{an}^{i-j}\overline{P_{shn-1}}^{j}(1-\overline{P_{shn-1}})^{shn-1-i}\overline{P_a}^{i-j}(1-\overline{P_a})^{shn+j-1-i} \geqslant \lambda \\ \overline{P_{shn}} = \dfrac{\sum\limits_{i=1}^{shn} PHt_i}{shn} \end{cases} \tag{2}$$

其中，an 和 hn 分别为可信者和混合者的个数。PHt_i 为信任度第 i 高的混合者。式(2)和式(1)的原理类似，这里就不做具体推导。

step4　将全部可信者和信任度最高的 shn 个混合者加入选择列表。

step5　将选择列表返回给请求者。

5　实验及结果分析

本节测试 3 种方法对恶意攻击的抵抗能力。第 1 种方法是基于 Beta 信任模型的方法，阈值设定为 0.5。并假设该能够使用某种其他机制能立即无误的获知服务结果的正确性。第 2 种方法是容错方法。第 3 种方法是基于信任容错的方法。实验检测指标为给定置信度的容错数和服务成功率。

5.1　独立恶意攻击

实验 1　原子实验。

实验 1 为原子实验，仿真一个领域的 200 个提供者的运行过程。实验设定如表 3 所示。

表 3　独立恶意攻击原子实验的参数设置

提供者数	请求次数	置信度	善意提供者		恶意提供者	
			比率	正确率	比率	正确率
200	3 000	[0.95,0.99]	50%	[80%,90%]	50%	10%

图 2~图 4 表明的是经过 3 000 次服务请求后，3 种方法的结果。其中横坐标是提供者序号，1~100 号为恶意提供者。纵坐标表示的是各提供者的正确率、信任度和负载。

图 2 为容错方法对于独立恶意攻击的抵御情况。为了和基于信任容错的可靠性增强方法比较，在实验中容错数设为 6。大部分提供者的负载在 80~110 随机分布。由于没有机制辨认恶意提供者，提供者服务的准确率和负载没有任何相关性，服务可靠性没有得到很好的保障。

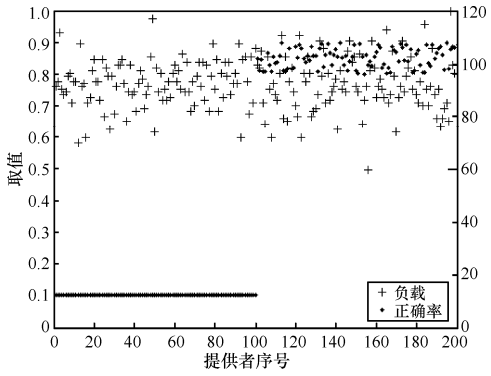


图 2　容错方法负载和正确率的关系

从图 3 中可以看到，基于信任的方法可以较好检测出恶意提供者。几乎所有的恶意提供者信任度

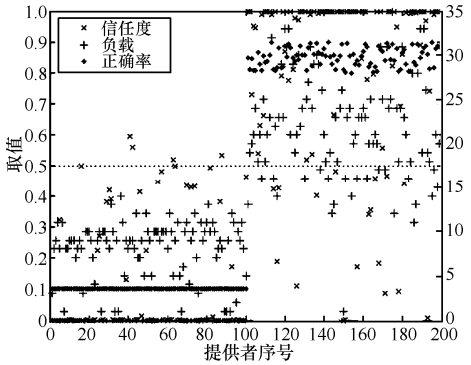


图 3　基于信任的方法负载和成功率同正确率的关系

为 0，而剩余的大部分不到阈值 0.5；而大部分善意提供者被认定为可信。大部分恶意提供者的负载大于 8；而善意提供者的负载在 10~35 之间。基于信任的方法可以在一定程度上识别善意和恶意提供者。其缺点为恶意提供者也有很高的负载，这意味着该方法对恶意攻击抵御较差。

图 4 是使用基于信任感知容错的方法，提供者的信任度和负载同正确服务比率具有相同的趋势。几乎所有恶意提供者的信任度都为 0，其负载几乎为 0；而善意提供者的信任度一般高于阈值 0.3，并且负载比较大。其中负载的最高值是 300。而理论上如果所有善意提供者的具有相同的访问量，其负载为 180，两者差距不大。

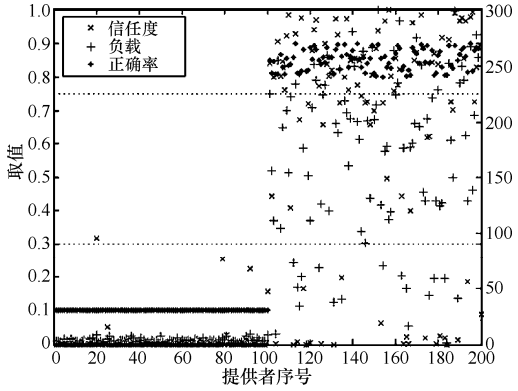


图 4 基于信任容错的方法负载和成功率同正确率的关系

实验 2 恶意节点率对容错数以及成功率的影响。

本组实验中，恶意提供者的比率为[0,90%]，每增加 10%进行 100 次原子实验，计算平均的容错数和服务任务成功率。

表 4 显示的是容错方法和基于信任容错的方法所需的容错数。前者是由理论计算获得。当恶意提供者比率增加时基于信任容错的方法的平均容错数增速平缓，说明其对于善意和恶意的环境有平稳的性能。而容错方法在无恶意提供者的环境中都需要 5 个提供者。而且其容错数增速很快，当领域有超过 30%的恶意提供者时，已不具有实现性。

表 4 容错数和恶意提供者比率的关系

恶意提供者比率	基于信任容错的方法	容错方法
0	4.26	5
0.1	4.59	7
0.2	4.88	17
0.3	5.19	43
0.4	5.38	100+
0.5	5.86	100+
0.6	6.14	100+
0.7	6.48	100+
0.8	6.61	100+
0.9	6.76	100+

图 5 显示的是使用 3 种方法的提供者比率和成功率的关系。当恶意提供者比率增长时，容错和基于信任的方法成功率下降的幅度较大，即该方法不能有效地防御独立恶意攻击。基于信任容错的方法在所有情况下都能达到高于 0.96 的成功率。

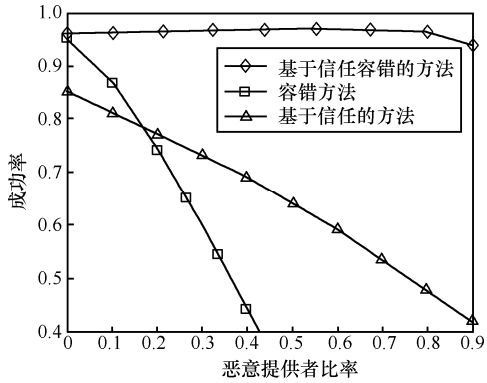


图 5 恶意攻击提供者比率和成功率的关系

通过实验 2，可以看到基于信任容错的方法在错误容忍和环境的适应方面都比基于信任的和容错方法要好。

实验 3 置信度对容错数及成功率的影响。

本组实验研究特定的恶意提供者比率下，基于信任容错方法的容错数、置信度和成功率的关系。本组实验中，恶意提供者比率设置为 0.7、0.8 和 0.9。而置信度为[0.9,0.99]，每增加 0.01，运行 100 次原子实验，并计算容错数和成功率的平均值。

从图 6 看到，随着置信度的增加，容错数缓慢增加到大约 8.3，说明该方法能够根据请求者需求动态调配资源，并且开销在系统能够接受的范围之内。从图 7 可以看到，在 3 种恶意提供者比率下，超过 96%的任务达到了置信度要求。

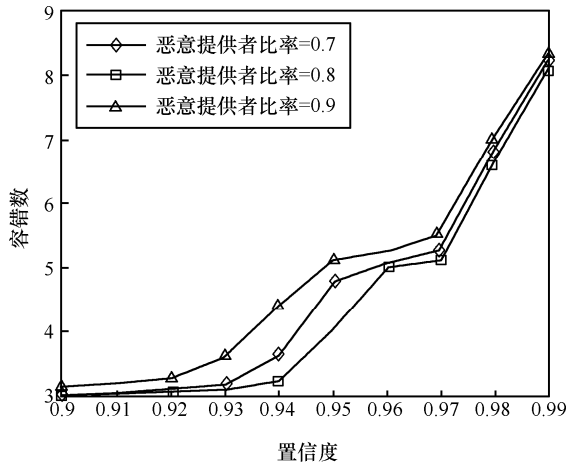


图 6 容错数和置信度的关系

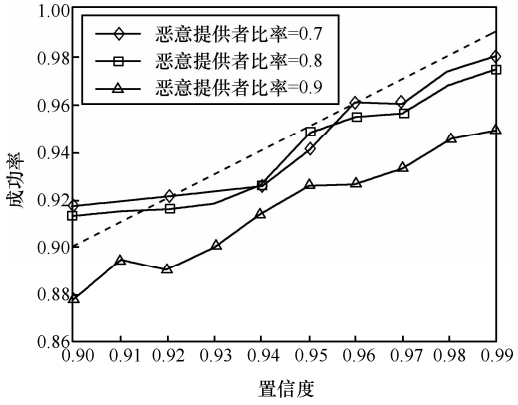


图 7　成功率和置信度的关系

5.2　复杂策略攻击

本组实验比较 3 种方法对于复杂策略攻击的抵抗能力。恶意提供者行为策略设定为交替提供 *CN* 次正确服务和 *MN* 次恶意服务。恶意提供者比率为[0,90%]，每 10%运行 100 次原子实验。分别测试 3 种策略：(CN,MN) =(2, 1), (1, 1), (1, 2)。完成后计算平均成功率。

从图 8 中可以看到，不论是善意还是恶意的环境，基于信任容错的方法都有很好的抵抗能力。相对而言，容错方法适用于恶意提供者较少的环境，而基于信任的方法适用于恶意提供者较多和恶意提供者产生恶意服务较多的环境。

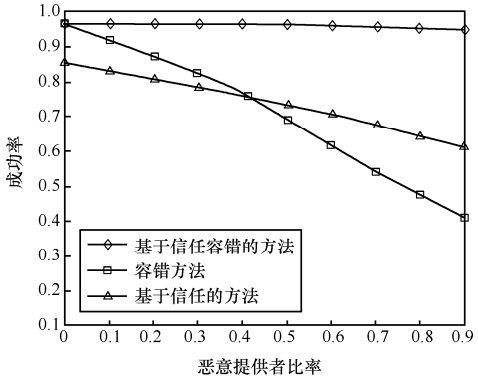
5.3　合谋攻击

在本组实验测试了 3 种方法对合谋攻击的抵抗能力。实验中采取对而谋者最有利的设定：即所有的恶意提供者都属于一个合谋的集团。恶意提供者比率为[0.4,0.5]。恶意提供者比率每增加 0.01 运行 120 次原子实验后，计算平均成功率。

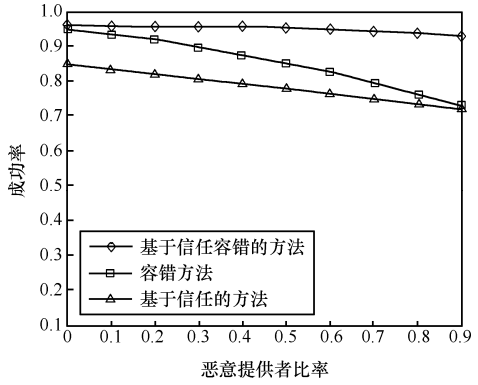
从图 9 可以看到，随着恶意节点率的增加，其他 2 种方法的成功率缓慢的下降。其原因是基于信任的方法没有考虑到提供者之间的联系，所以不受合谋攻击的影响。而基于信任容错的方法在恶意节点率在 0.4~0.46 都具有高于 0.85 的成功率。

表 5 为单次实验的服务成功率的部分数据。

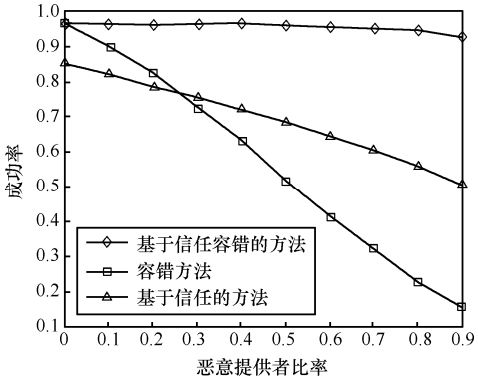
可以看到，成功率的取值处于 2 个极端——大于 0.96 或接近于 0.02。当恶意提供者比率增大时，取 0.02 的次数也在缓慢增大。其原因是当恶意节点较少时，系统在初始阶段取善意节点的概率较大，而选举协议中的多数获胜原则可以将恶意提供在清除出可以状态，所以成功率很大。当恶意节点增加时，共谋的恶意节点可能会处于可用或混合状态，同样由于选举协议的多数获胜原则，恶意提供者可以提升



(a) 攻击策略为(2,1)



(b) 攻击策略为(1,1)



(c) 攻击策略为(1,2)

图 8　复杂策略攻击下成功率和恶意提供者比率的关系

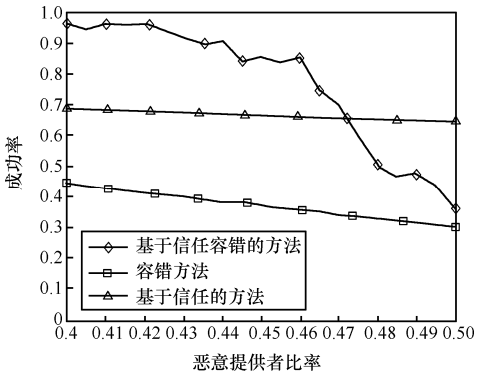


图 9　合谋攻击下成功率对比

表 5　　单次实验的成功率

4	4.2	4.4	4.6	4.8	5
0.959	0.967	0.960	0.960	0.962	0.967
0.963	0.959	0.955	0.031	0.032	0.958
0.964	0.970	0.956	0.965	0.965	0.969
0.957	0.957	0.965	0.966	0.029	0.963
0.959	0.962	0.967	0.030	0.964	0.032
0.962	0.960	0.962	0.956	0.968	0.970
0.963	0.960	0.967	0.029	0.962	0.029
0.956	0.961	0.957	0.959	0.966	0.966
0.964	0.958	0.962	0.962	0.963	0.961
0.963	0.960	0.025	0.964	0.975	0.036
0.962	0.951	0.965	0.959	0.027	0.031
0.963	0.958	0.956	0.965	0.029	0.968
0.964	0.964	0.959	0.959	0.965	0.032
0.958	0.958	0.952	0.964	0.028	0.034
…					

共谋者的信任度，并贬低善意提供者的信任度，从而使得共谋恶意提供者占据信任度较高的位置。由于算法的自驱动机制，使得之后的服务提供模块会更可能选择恶意的提供者，造成成功率很低。

从实验可以看到，在较善意的环境中，基于信任容错的方法对共谋攻击的抵抗能力最好。

6　结束语

本文提出了一种基于信任容错的服务可靠性增强方法，利用选举协议设计服务失效轮询检测机制；建立信任机制度量提供者的可靠性；依据提供者的领域特点和请求者的可靠性需求设计信任感知的容错服务冗余数计算方法和冗余服务选择算法，具有面向领域，用户偏好敏感的特点。实验验证本方法比容错和基于信任的方法更能抵御独立节点攻击、复杂策略攻击和合谋攻击。

在今后的工作中，将试图提升高置信度下服务的成功率，并且考虑来自请求者的攻击以及完善和设计虚拟货币机制保障有效的反馈。

参考文献：

[1] 怀进鹏.对未来网络化软件技术的几点认识[R].中国计算机大会 CNCC07 特邀报告, 2007.
HUAI J P. Understandings about Future Networked Software Techniques[R]. China National Computer Conference 2007, 2007.

[2] 王远, 吕建, 徐锋等. 一种适用于网构软件的信任度量与演化模型[J]. 软件学报,2006,17(4): 682-690.
WANG Y, LV J, XU F, *et al*. A trust measurement and evolution model for internetware[J]. Journal of Software, 2006, 17(4): 682-690.

[3] ANATOLIY G, VYACHESLAV K, ALEXANDER R, *et al*. Using inherent service redundancy and diversity to ensure Web services dependability[A]. Workshop on Methods, Models and Tools for Fault Tolerance[C]. 2007. 324-341.

[4] CHANDRA S, CHEN P M, WHITHER G. Recovery from application faults? a fault study using open-source software[A]. International Conference on Dependable Systems and Networks[C]. New York, USA, 2000. 97-106.

[5] GORBENKO A, KHARCHENKO V, POPOV P, *et al*. Dependable composite Web services with components upgraded online[A]. International Conference on Dependable Systems and Networks[C]. 2004.92-121.

[6] DESWARTE Y, KANOUN K, LAPRIE J. Diversity against accidental and deliberate faults computer security[A]. Computer Security, Dependability and Assurance: from Needs to Solutions[C]. 1998. 171- 181.

[7] 刘玲霞，武兆雪，钱渊等. Web 服务容错技术研究[J]. 计算机科学, 2009,36(1):24-28.
LIU L X, WU Z X, QIAN Y, *et al*. Fault-tolerant Web services[J]. Computer Science, 2009, 36(1):24-28.

[8] JAYASINGHE D. FAWS - a client transparent fault tolerance system for SOAP-based Web services[EB/OL]. http://www-128.ibm.com/developerworks/webservices/library/ws-faws/,2005.

[9] AGHDAIE N, TAMIR Y. Implementation and evaluation of transparent fault-tolerant Web service with kernel-level support[A]. The IEEE International Conference on Computer Communications and Networks[C]. Miami ,Florida , 2002. 63-68.

[10] SANTOS G T, LUNG L C, MONTEZ C. FTWeb: a fault tolerant infrastructure for Web services[A]. The 2005 Ninth IEEE International EDOC Enterprise Computing Conference (EDOC'05)[C]. Enscheda, the Netherlands, 2005.95-105.

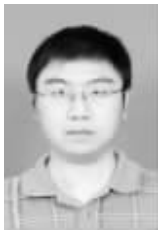
[11] DERON L, FANG C L, CHEN C, *et al*. Fault-tolerant web service[A]. The 10th Asia-Pacific Software Engineering Conference[C]. Taipei, Taiwan, China, 2003. 310-319.

[12] SUNGKEUN P, LING L, CALTON P, *et al*. Resilient trust management for Web service integration[A]. IEEE International Conference on Web Services[C]. Atlanta, USA, 2005. 11-15.

[13] XIN Y Z, YI N C, XIAO Y B. On testing and evaluating service-oriented software[J]. Computer Volume, 2008, 41(8): 40-46.

[14] ZAKI M, ATHMAN B. Reputation bootstrapping for trust establishment among Web services[J]. IEEE Internet Computing, 2009, 13(1): 40-47.

作者简介：



杨墨（1982-），男，湖北武汉人，武汉大学博士生，主要研究方向为可信计算、信任管理。

王丽娜（1964-），女，辽宁营口人，武汉大学教授、博士生导师，主要研究方向为网络安全、信息隐藏、可信计算。