一种可生存系统的自主管理模型

苏 岩[1] 赵国生[1] 王 健[2] 张 楠[1] 李 琳[1]
(哈尔滨师范大学现代实验中心 哈尔滨 150025)[1] (哈尔滨理工大学计算机学院 哈尔滨 150001)[2]

摘 要 提出了一种基于生存检测参数的可生存系统自主管理模型。自主管理机制通过自主检测和控制单元来实现。首先,定义了若干生存检测参数,依据累计分布函数确定了动态可变的阈值约束;然后,基于生存参数的参考基点,给出了服务连接综合评判的计算方法,并通过控制单元来实现可生存系统的自主管理;最后,通过仿真实验说明文中所提方法提高了可生存系统的服务承载能力,增强了系统的可生存能力。

关键词 可生存性,自主管理,控制单元,累计分布函数,检测参数
中图法分类号 TP302 **文献标识码** A

Model of Autonomous Management for Survivable System

SU Yan[1] ZHAO Guo-sheng[1] WANG Jian[2] ZHANG Nan[1] LI Lin[1]
(Modern Experimental Center, Harbin Normal University, Harbin 150025, China)[1]
(School of Computer Science and Technoogy, Harbin University of Science and Technology, Harbin 150001, China)[2]

Abstract This paper presented a model of autonomous management for survivable system based on the survivable detection parameters. Autonomous management mechanism is implemented by autonomous detection and control unit. Firstly, a number of survivable detection parameters were defined, and according to the cumulative distribution function, the dynamic and variable threshold constraints were determined. Then, based on the reference mark of parameters, the calculation methods of comprehensive evaluation for service connections were given, and autonomous management of survivable system was achieved through the control unit. Finally, the simulation experiment shows that the proposed method can effectively improve the service capacity of survivable system and enhance the survivability of system.

Keywords Survivability, Autonomous management, Control unit, CDF, Detection parameters

信息系统现实的安全状况是无论如何防御,入侵总会发生,无论如何检测,系统总会遭到不同程度的破坏,关键任务的失效在所难免。如何在各种安全灾难发生时尽量保证现有关键任务的持续、及时完成,已成为亟待解决的问题。可生存性(Survivability)研究就是顺应这个需求而产生的新课题。

可生存性作为下一代网络安全的核心目标,代表了网络安全发展的新方向。可生存性的思想可以总结为自适应的"容",实际上是让系统带"病"坚持工作,继续提供其关键任务,即使在系统遭受到成功入侵后,甚至系统的重要或部分组件遭到损坏或摧毁时,系统只要能在结构上合理配置资源,能在攻击下资源重组,就依然能够保证按时完成其提供的各种关键服务,并能及时修复被损坏的关键服务[1]。从网络安全的历史、现状和发展趋势来看,提高系统的可生存性,是目前应付一切攻击、入侵和破坏的最佳途径。

生存性是一个综合的概念,根据 Ellison[2] 的生存性定义,可以通过系统的防护、监视和响应 3 个方面来描述,这是一个闭环型自主响应机制。在查阅大量国内外相关文献后,发现现有研究较多关注的是生存性的定义[1,2]、模型分析[3,4]、生存性评估[5-7]、形式化建模[8-10]、生存性防护[11,12]、生存应急恢复[13,14]等方面,并对其进行了大量工作,而对可生存系统整体上的自主响应管理机制的研究仅局限于 CMU、CERT/CC、DARPA 等几个研究机构,并且目前这方面的研究还未见诸报道。因此,本文承接前期在生存性方面的研究成果[15]将着重从可生存系统的自主管理机制方面来研究可生存系统的主动、自适应性的生存增强机制。

对于可生存系统的自主管理机制,本文认为应该从"检测"+"控制"两个方面入手。对于检测机制,则应从系统内部的服务状态和外部的攻击两个角度入手。对系统内部运行状况的管理包括服务状态监控、内部完整性检查、操作系统的协议栈、完整性检测以及维护系统各部分间的信任模型等。而面向生存性的外部服务攻击管理研究主要侧重的是主动的生存应急响应,在检测到攻击时及时做出自调节和重配置等可生存性规避机制以使系统持续提供服务。对于系统内部状态的识别,现有许多研究都已有实现,而对可生存系统外部服务攻击的自主检测上的研究还未见诸报道。本文从面向生存性的外部攻击检测入手,侧重研究主动的生存应急响应。对于

到稿日期:2013-07-29 返修日期:2013-09-01 本文受国家自然科学基金(61202458),黑龙江省教育厅科学技术项目(12521146,12511099)资助。

苏 岩(1972—),男,硕士,助理研究员,主要研究方向为可信网络、自律计算,E-mail:zgswj@163.com;**赵国生**(1977—),男,博士,副研究员,主要研究方向为可生存系统、认知网络、自律计算,E-mail:zgswj@163.com(通信作者);**王 健**(1979—),女,博士,副教授,主要研究方向为认知网络、可信计算;**张 楠**(1988—),女,硕士生,主要研究方向为认知网络、可生存系统;**李 琳**(1989—),女,硕士生,主要研究方向为可生存系统。

控制机制，本文第2、3节研究了自主控制单元的工作机制，以此来实现可生存系统自主管理功能。

1 生存检测参数

可生存系统会以访问用户和网络状况中的不同连接状态参数作为参考，在本文中以下几个参数指标被用来对可生存系统的外部服务连接进行检测。

·正常运行时间和停止运行时间

正常运行时间(Uptime)这个参数代表的是一个访问用户从开始连接服务器到断开连接所经过的时间。而停止运行时间(Downtime)这个参数代表的是一个用户从断开与服务器的连接到再次与服务器连接的时间间隔。在没有攻击发生的情况下，系统可以在不同的时间段随机选择访问用户并计算他们的正常运行时间和停止运行时间，然后系统就可以获得合法用户的正常运行时间和停止运行时间的累积分布函数CDF(cumulative distribution function)。

·请求速率和下载速率

请求速率(Request rate)这个参数代表在特定的时间段内一个访问用户对服务器发出请求的次数。而下载速率(Download rate)代表在特定时间段内一个访问用户从服务器下载的次数。服务器通过在不同时间随机地统计不同用户的下载速率和请求速率，来得出一个下载速率和请求速率的累积分布函数CDF。

·访问行为

用户访问行为(Browsing behavior)主要取决于两个方面，第一个方面是网站的层次结构，第二个方面是用户的习惯。一般来说，一个网站是由很多个网页构成的，这些网页通过有层次结构的超级链接彼此联系。用户的习惯一般代表着访问用户更愿意浏览哪些网页，有哪些超级链接会被一个正常的访问用户点击。

页面分类(Page based on type)：网页可以按照逻辑和内容来划分。举个例子，一个新闻网站可以按照以下类型分类：政治、财经、体育、文化，等等。服务器可以统计出每个用户对于每种类型的网页的请求速率，不同时间不同用户搜集到的数据可以帮助服务器计算得到累积分布函数(CDF)。同样，累积分布函数(CDF)可以帮助服务器对每种类型的网页算出一个阈值，例如当攻击发生时如果一个用户访问一个类型网页的请求速率超过了该类型网页的阈值，那么它就被定义为一个攻击行为。

访问率(Page access rate)和喜好度(Page popularity)：一般来说，每个网页都有不同的访问率，一些网页经常被人浏览，而另一些网页很少被人浏览。有研究指出，一个Web网站仅有20%的内容被客户高频访问(约80%)，而且文件被访问的概率呈zipf分布[16]。这些参数对于判断服务器是否遭到攻击具有重要意义，因为一个攻击者一般并不具备一个正常访问者的上网行为，而是通过随机的方式访问网页。服务器可以在一个时间段内统计每一个网页的访问率，然后服务器可以计算出每个网页的喜好度，公式如下：$p_i^t=a_i^t/\sum_{j=1}^{N}a_j^t$，其中$P_i^t$代表网页$i$的喜好度，$t$代表访问时间间隔，$N$代表所有网页的总数量，$a_i^t$代表一个网页$i$在时间间隔$t$内的访问率。

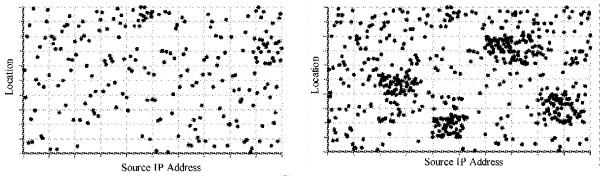
超链接点击数(Hyperlink click number)：假设一个网页有k个超链接。在正常服务情况下系统可以分析出这k个超链接被用户正常点击的比率。这样系统就可以设定一个阈值，如果当前对某个超链接的点击和阈值比较有较高的偏差，则这个用户是恶意用户的可能性就较大。

超链接深度(Hyperlink depth)：其表示用户点击有序链接页的多少。同样，系统也可以为这个属性计算其CDF。

对于用户访问行为还有一些其他的属性，比如重复页面、顺序连接页面和过期页面等，可以参考文献[17]。

·源IP地址

一般来说，合法用户的源地址(Source IP address distribution)一般都会广泛地分布在因特网上。而攻击者的源地址分布一般都会集中在一个区域内，因为攻击者会侵入一个局域网内捕获大量僵尸肉鸡，对服务器进行攻击，所以大多数僵尸肉鸡会集中在一个局域网中。从图1(a)可以看出用户均匀地分布在网络上，而图1(b)中出现了若干地址集束，说明这些IP很有可能就是攻击者机器。



(a) 合法用户的源地址分布　　(b) 攻击者的源地址分布

图1　源地址分布图

·用户到达率

正常情况下，用户到达率(Arrival rate of users)服从泊松分布。如果在一定时间间隔内用户d的到达率是λ，那么在这个时间间隔内k个用户请求连接服务器的概率是：$f(k,\lambda)=\frac{\lambda^k e^{-\lambda}}{k!}$。举个例子，如果在一秒钟内有两个用户连接系统($\lambda=2$)，那么20个用户在一秒钟内连接到系统的概率是$5.8\times10^{-14}$。在一次服务攻击中，会有大量的攻击者同时访问系统，根据以上概率，我们可以预测出连接系统的大部分用户是恶意攻击者。

2 自主管理模型

2.1 检测方法

首先我们假设一个对可生存系统的服务连接为c，那么这个连接就会有i个生存参数A_i^c，参数A_1^c可以代表请求速率，A_2^c可以代表下载速率等等，以此类推。$S(A_i^c)$就是连接c对应参数A_i的检测值，进而可以对一个连接的所有参数的数值求和：$S(c)=\sum_{i=1}^{n}S(A_i^c)$，$n$表示参数的数量。

下面我们给出一个生存参数的检测值的计算方法。这些参数包括正常运转时间、停止运转时间、下载速率和请求速率、页面喜好度、超链点击数和超链深度等。假设$y=f_i(x)$，f_i表示参数A_i的累积分布函数(CDF)，控制单元会确定一个数对(x_b,y_b)作为参考基点，$y_b=f_i(x_b)$。控制单元会通过回馈控制进程来选取一个合适的基点。假设连接c的参数A_i用x来表示，那么我们给出一个计算检测分值的公式：

$$S(A_i^c)=\begin{cases}0, & x_i^c\leqslant x_b\\ -1\times k^{div(\frac{x_i^c-x_b}{\Delta x})}\times\dfrac{x_i^c-x_b}{\Delta x}, & x_i^c>x_b\end{cases}\quad(1)$$

其中，k表示一个几何定值，例如1,2,…。$div(m/n)$表示m

和 n 取商。Δx 表示一个恒定的比例因子，如果基点的偏离量越高，那么在这个公式中所得到的值就越低。服务器可以根据攻击的频率适当地选择定值 k 和 Δx 的值。对于停止运行时间这个参数的综合评判计算则是与其他参数的计算方法相反，停止运行时间的综合评判公式如下：

$$S(A_i^c)=\begin{cases}0, & x_i^c \geqslant x_b \\ -1\times k^{div(\frac{x_b-x_i^c}{\Delta x})}\times\dfrac{x_b-x_i^c}{\Delta x}, & x_i^c < x_b\end{cases} \tag{2}$$

但是源 ip 地址分布这个参数是一个定值，如果服务器猜测这个 ip 地址是一个攻击者的 ip 地址，那么服务器就会给出一个定值，例如－1，否则定为 0。

2.2 控制单元

控制单元的引入一是为了实现可生存系统的自主管理，二是预防可生存系统的服务资源被恶意攻击耗尽，导致系统当机而无法提供正常的服务。控制单元首先为可生存系统的各种服务资源量定义 threshold1、threshold2 和 threshold3 3 种阈值划分和红色、黄色、绿色 3 种生存状态。其中，Threshold1、threshold2 和 threshold3 分别表示系统服务资源总量的 90%、80%和 60%。当系统的服务资源超过 threshold1 时，我们设定系统的资源状况为红色，当系统的服务资源处于 threshold2 和 threshold3 之间时，我们将系统资源状态设定为黄色。而当系统处于 threshold3 以下时，我们将系统的资源状态设定为绿色(正常状态)。

控制单元会定期地检查系统的各种资源状态，例如 ip 栈、cpu 状态、内存等。当系统的服务资源超过 threshold1 时，系统就会进入一种警戒模式，并且控制单元的进程就会被触发。与此同时系统不会再允许新的用户连接到系统。接下来控制单元就会给已经建立起来的服务连接计算出一个综合检测值，如果得到的数值低于阈值，就断开连接。这种工作一直持续到系统资源恢复到 threshold2 以下。当系统资源状态不是红色的时候，系统就可以离开警戒模式并且可以接受新的用户请求了。当资源状态是黄色的时候，如果有新的用户请求，那么控制单元会检查这个用户的源 ip 地址是否在黑名单中。如果在黑名单中，那么系统之前就会保存有一个停止运行时间参数的检测值；如果该 ip 地址没有在黑名单中，那么系统就会算出它的停止运行时间的检测值。然后将新连接的综合检测值和已经建立起来的连接的综合检测值作比较，如果新连接的值较高，就将以前的连接断开，接受新的连接请求。

控制单元遵循以下 3 条规则。

规则 1 检测值为 0 的服务连接不能被当掉。

规则 2 如果服务连接的检测值大于等于阈值，控制单元就给用户发送一个 puzzle 测试。如果用户完成了这个测试，连接就不被当掉，如果用户并没有完成测试，连接就被当掉。

规则 3 如果连接的检测值小于阈值，那么服务器不需要发送一个 puzzle 测试，但是连接被放入当机候选队列中。

3 自主管理的过程

控制单元的自主管理过程如下：

Step 1 控制单元为每个识别参数设定一个初始参考基点，初始基点的值可以根据以往的攻击情况和系统状态来设定。一般来说初始值的选择最好能够使得误判最小化，所以我们一般将初始化的值设为较大的值，这样最初阶段的误判为 0。

Step 2 控制单元要定期监控已经建立起来的连接，并且给每一个建立起来的连接按照这段时间的基点给出一个检测值。

Step 3 控制单元会断开一些检测值较低的连接，直到没有一个负值的连接存在，或这个服务器资源恢复到 threshold2 以下。

Step 4 如果系统的资源状态不是红色，那么服务器就会终止警戒模式。如果至少有一项资源处于黄色状态，那么控制单元仍然需要计算已建立的连接的检测值，并且等待新的用户请求。

Step 5 如果系统资源状态恢复到绿色，那么控制单元的进程就会被终止。

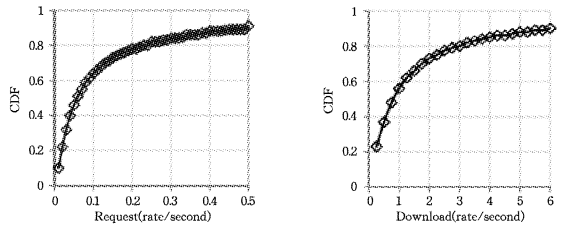
Step 6 如果至少有一项系统资源状态处于红色并且没有任何连接的检测值低于阈值，那么控制单元就重新设定一个适当的初始基点。其中除了停止运行时间和源 ip 地址两项参数外，其他的初始基点需要做如下调整：假设在一个参数的累积分布函数中，(x_b, y_b)和(x_b', y_b')分别是当前的基点和下一个基点。对于下一个基点，我们可以通过减少一个 Δy (Δy 为正量)得到 y_b'。于是我们得到 $y_b'=y_b-\Delta y, x_b'=f^{-1}(y_b-\Delta y)$。所以下一个基点坐标就是$(f^{-1}(y_b-\Delta y), y_b-\Delta y)$。对于 Δy 的取值，我们可以取为 0.05 或者 0.1。而对于停止运行时间这个参数，下一个基点的计算方法虽然与以上方法相似，但它的值却是变大的，所以停止运行时间的下一个基点的坐标是$(x_b', y_b')=(f^{-1}(y_b+\Delta y), y_b+\Delta y)$。对于源 ip 地址分布这个参数，它并不存在累积分布函数和基点，它的分数在之前已经被确定。

Step 7 算法返回到步骤 2。这个循环一直持续到步骤 5 的条件满足后终止。

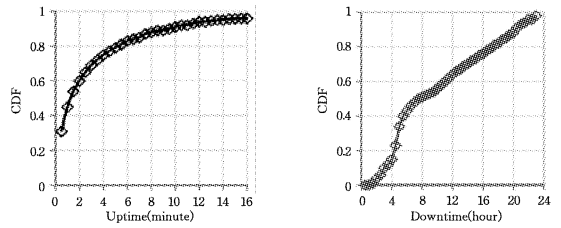
4 仿真与分析

我们基于虚拟机搭建了一个网络生存环境，该网络对外只提供一种 www 关键服务。在实验中，我们测试了两周内所有对于该 Web 系统的 http 请求。

在图 2(a)中大约 50%以上的用户的请求速率都低于 0.066，这说明这些访问用户大约每 15 秒钟会访问一次 Web 服务器。另外，大约 80%的访问用户的请求速率都小于 0.25，大约 10%的访问用户每 2 秒发送一个以上的请求。因此我们将初始基点设置为$(x_b, y_b)=(0.5, 0.9)$。在图 2(b)中大约超过 50%的访问用户的下载速率都低于 1000byte/秒，大约超过 80%的访问用户的下载速率低于 3000byte/秒，而只有 10%的访问用户的下载速率超过 6000byte/秒。所以我们选择$(x_b, y_b)=(6000, 0.9)$作为初始基点。在图 2(c)中大约 54%的访问用户在线时间都会少于 1.5 分钟，80%的访问用户在线时间都低于 5 分钟，而只有 4%的用户在线时间超过 16 分钟。所以我们选取$(10, 0.9)$作为初始基点。在图 2(d)中大约 50%的访问用户的停止运行时间都超过 8 小时，80%的访问用户的停止运行时间都超过 4 小时，而只有 3%的访问用户的停止运行时间少于 1 小时。所以我们选择$(3, 0.1)$作为初始基点。

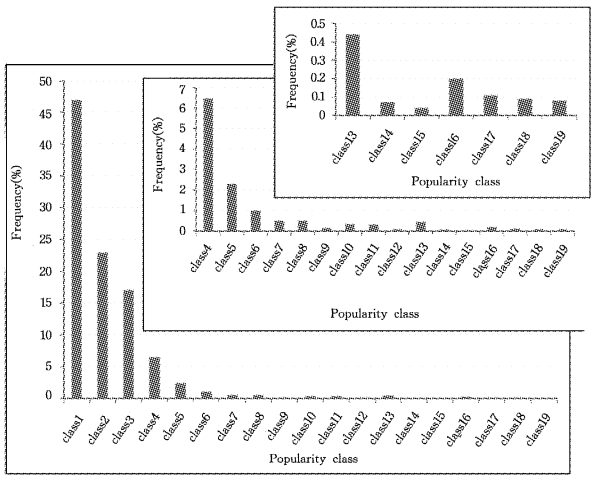


(a) 请求速率　(b)下载速率

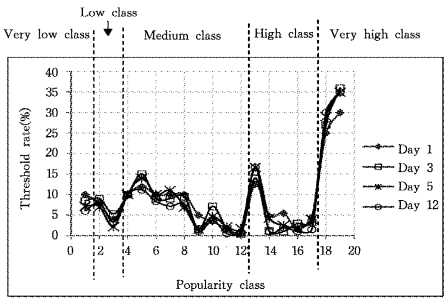


(c) 正常运行时间　(d) 停止运行时间

图 2　累积分布函数图



(a) 不同等级喜好度的分布率

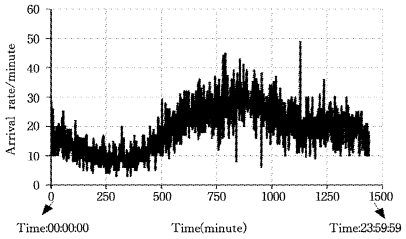


(b) 不同等级喜好度的阈值率

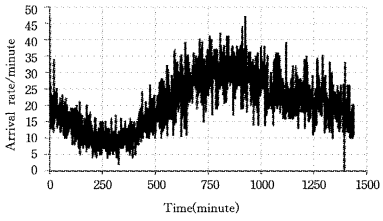
图 3

图 3(a)中给出的是 3 天中 11:00－14:00 网页喜好度的分布图。在前文中我们根据好感度把网页分为 5 个等级:超低、低、中、高、超高。为了更好地用数据分析网页的喜好度,需要将这 5 个级别更加细分。在图 3(a)中超低这个等级由 class1 组成,而低等级可细分为 class2 和 class3,中级由 class4 到 class12 组成,高级由 class13 到 class17 组成,超高等级则细分为 class18 和 class19。在图 3(a)中,若 $i>j$,那么 class i 的喜好度就要高于 class j。图 3(b)显示的是第 1、3、5 天和第 12 天基于累积分布函数 CDF 得到的不同等级喜好度的阈值率(百分比)分布图。从图中可以看出,虽然日期不同,但是每一个喜好度等级类的阈值率都大致保持在一个相似的范围值内。因此,我们可以确定一个喜好度等级类的上限阈值率和与日期无关的一个固定阈值率。

图 4 显示了第 1 天和第 12 天用户到达的分布率。从图中可以看出,到达率在一天中不同时间段是不同的。在这两天中,半夜 1:00 到凌晨 6:00 的用户到达率低于其他部分,中午 11:00 到 16:00 的用户到达率达到最大。用户平均到达率是每分钟 27 个用户,可以看出 1 分钟内少于 50 个用户连接服务器,换句话说,每秒不多于 1 个用户访问 Web 服务。



(a) 第 1 天用户到达率



(b) 第 12 天用户到达率

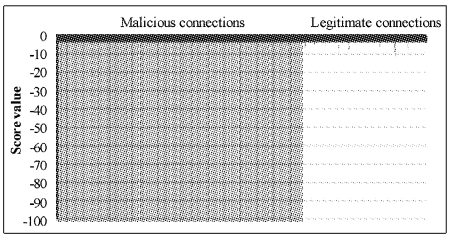
图 4

为了验证模型对提高可生存系统自主应急管理能力的有效性,我们在 EMulab 环境下建立了 Clarknet www 服务的仿真试验。时间跨度 6 天,共计产生 3559 个大小不同的可访问页面,然后在 14:00 前将它们上传到 Web 服务器。在本次试验中,我们将攻击开始时间定为 14:03,服务器带宽设定为 10^6 字节每秒。

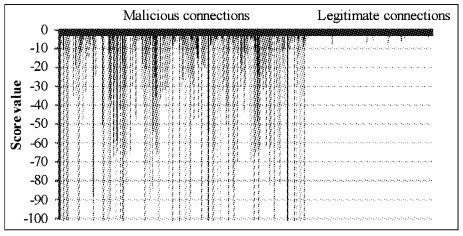
实验中采用了两种攻击方式:普通攻击和隐藏攻击。对于普通攻击,使用了 150 台主机作为攻击者的角色,采用 Netsky. Q、Trojan Sientok 和 BlueCode. Worm 这 3 种病毒程序向 Web 服务器发送 get 请求,这 3 种病毒程序的请求速率分别是 300 毫秒、250 毫秒和 130 毫秒。而对于隐藏攻击,200 台攻击主机以与合法用户相同的请求速率和下载速率向 Web 服务器发出 TCP 连接请求。

图 5(a)和(b)分别显示在普通攻击和隐藏攻击下已经和服务器建立连接的综合检测值分布图,红色条表示非法连接的得分而绿色条代表合法连接的得分。大约有7.5%的合法用户得到了一个负的检测分值,但绝对值很小(最大 11)。在普通攻击中,大部分的恶意连接都得到了一个非常低的综合检测分值,这主要是因为攻击者会用一个非常高的请求速率连接服务器,所以请求速率这个参数的检测值就非常低。在隐藏攻击中,不是所有的恶意连接都会得到负的检测分值,而且这个负的检测分值的绝对值有可能还要低于普通攻击检测分值的绝对值。

借助 Load Runner8. 0 测试工具,我们对不采用本文所提方法(试验 A)和采用本文所提方法(试验 B)进行了生存应急的实验对比,实验数据如表 1 和表 2 所列。



(a) 普通攻击的检测值分布



(b) 隐藏攻击的检测值分布

图 5　攻击发生时的检测值的分布

表 1　实验 A 的测试数据表

	隐藏攻击	普通攻击	合法用户
主机数	200	150	200
请求速率	275150	300250130	350275150
请求次数	24856	27821	23247
响应次数	13950	18037	11479
成功率	56.12%	64.83%	49.37%

表 1 的实验结果显示，平均有 64.83% 的普通攻击和 56.12% 的隐藏攻击都得到了系统服务响应，而只有平均 49.37% 的合法用户请求得到了响应，严重干扰了可生存系统的正常服务能力，高可生存系统的正常服务能力无从体现。

表 2　实验 B 的测试数据表

	隐藏攻击	普通攻击	合法用户
主机数	200	150	200
请求速率	275150	300250130	350275150
请求次数	29361	31250	34592
响应次数	1908	405	29185
成功率	6.5%	1.3%	84.3%

表 2 的实验结果显示，不仅可生存系统的服务承载能力提高了 25.3%，而且合法用户的服务完成率提高了 70.7%，普通攻击和隐藏攻击的服务响应率下降到可容忍的程度，可生存系统的自主服务能力大大高于实验初始预期，说明了本文所提方法的可行性。

结束语　可生存性作为下一代网络安全的核心目标，代表了网络安全发展的新方向。从网络安全的历史、现状和发展趋势来看，提高系统的可生存性，是目前应付一切攻击、入侵和破坏的最佳途径。本文从面向生存性的外部攻击识别入手，侧重研究主动的生存响应管理，以在检测到攻击时及时建立自调节和重配置等可生存性规避机制，使系统持续提供服务。提出了一种基于生存检测参数的可生存系统自主管理机制，实验结果显示该方法提高了可生存系统的服务承载能力，增强了系统的可生存能力。

参 考 文 献

[1] Westmark V R. A Definition for Information System Survivability[C]//Proceedings of the 37th Hawaii Internal Conference on System Sciences, Track 9. Washington, IEEE Computer Society Press, 2004: 2086-2096

[2] Ellison R J, Fisher D A, Linger R C, et al. Survivable network systems: An emerging discipline[R]. CMU/SEI, Technical Report, CMU/SEI-97-TR-013. 1997

[3] Ellison R J, Moore A P. Trustworthy refinement through intrusion-aware design (TRIAD): An Overview[C]//Proceedings of the 3rd Annual High Confidence Software and Systems Conference. Baltimore, MD, 2003: 1-10

[4] Richard C L, Howard F L, John M, et al. Life-cycle models for survivable systems[R]. CMU/SEI-2002-ESC-TR-026. Boston: Carnegie Mellon University, 2002

[5] 赵国生，王慧强，王健. 一种基于灰色关联分析的网络可生存性态势评估方法[J]. 小型微型计算机系统，2006，27(10)：1861-1864

[6] 王健，王慧强，赵国生. 基于序列蒙特卡罗的网络生存态势跟踪评估[J]. 哈尔滨工业大学学报，2008，40(5)：802-806

[7] Wang J, Wang H Q, Zhao G S. A situation assessment method for network survivability[J]. Journal of Wuhan University: Natural Sciences, 2006, 11(6): 1785-1788

[8] Wang J, Wang H Q, Zhao G S. Formal modeling and quantitative evaluation for information system survivability based on PEPA [J]. The Journal of China Universities of Posts and Telecommunications, 2008, 15(2): 88-96

[9] 王健，王慧强，赵国生. 分布式任务关键系统生存性自动分析与验证[J]. 高技术通讯，2009，19(6)：572-579

[10] Zhao G S, Wang H Q, Wang J. A novel formal analysis method of network survivability based on stochastic process algebra[J]. Tsinghua Science & Technology, 2007, 12(1): 175-179

[11] Zinky J A, Bakken D E, Schantz R. Architectural support for quality of service for CORBA objects[J]. Theory and Practice of Object Systems, 1997, 3(1): 55-73

[12] Matti A H, Richard D S, Carlos A U, et al. Survivability through customization and adaptability: The cactus approach[C]//DARPA Information Survivability Conference and Exposition (DISCEX 2000). Hilton Head, SC, USA, 2000: 294-307

[13] Xie W, Hong Y, Trivedi K. Analysis of a two-level software rejuvenation policy[J]. Reliability Engineering and System Safety, 2005, 87(1): 13-22

[14] Patterson D, Brown A, Broadwell P, et al. Recovery-oriented computing (ROC): Motivation, definition, techniques, and case studies[R]. Technical Report UCB//CSD-02-1175. UC Berkeley Computer Science, 2002

[15] 赵国生. 任务关键系统生存性应急增强技术研究[D]. 哈尔滨：哈尔滨工程大学，2009

[16] Xie Y, Yu S Z. A large-scale hidden semi-markov model for anomaly detection on user browsing behaviors[J]. Networking, IEEE/ACM Transactions on, 2009, 17(1): 54-65

[17] Beitollahi H, Deconinck G. ConnectionScore: A Statistical Technique to Resist Application-layer DDoS Attacks[R]. Tech. Rep. 01-2012-0130. Electrical Engineering Department, University of Leuven, Belgium