# web 服务器拟态防御原理验证系统测试与分析

张 铮[1], 马博林[1], 邬江兴[2]

[1] 数字工程与先进计算国家重点实验室 郑州 中国 450001
[2] 国家数字交换系统工程技术研究中心 郑州 中国 450002

**摘要** web 服务器拟态防御原理验证系统是基于拟态防御原理的新型 web 安全防御系统, 利用异构性、冗余性、动态性等特性阻断或扰乱网络攻击, 以达成系统安全风险可控的要求。针对传统的测试方法实施于 web 服务器拟态防御原理验证系统中存在不足、不适应复杂安全功能测试以及难以实现准确度量等问题, 本文提出了适用于拟态防御架构的 web 服务器测试方法, 基于让步规则改进了灰盒测试, 还丰富了漏洞和后门利用复杂度的含义。并以此为基础设计适于该系统的测试方案、测试原则和测试方法, 在性能、兼容性、功能实现、HTTP 协议一致性, 安全性这些方面进行了全面的测试和分析。

**关键词** 拟态防御原理; 灰盒测试; 利用复杂度; 测试原则; 测试用例; 测试分析; 测试
**中图法分类号** TP393 DOI 号 10.19363/j.cnki.cn10-1380/tn.2017.01.002

# The Test and Analysis of Prototype of Mimic Defense in Web Servers

ZHANG Zheng[1], MA Bolin[1], WU Jiangxing[2]

[1] State key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China
[2] National Digital Switching System Engineering & Technological R&D Center, Zhengzhou 450002, China

**Abstract** Prototype of mimic defense in web servers is a new type of web security defense system based on mimic security defense theory, which makes use of heterogeneity, redundancy, dynamic and other characteristics to block or disrupt the network attacks, in order to achieve the requirement of controlling system security risk. The traditional web services testing methods are inadequate and do not meet the complex security testing requirements and have difficulty in accurate measurement. This paper presents a web services testing method which is applicable to mimic defense architecture, improve gray-box testing method based on concession rule and enriches the meaning of exploiting complexity of vulnerability and back door. Based on this, this paper puts forward the test projects, test principles and test methods for the newly system. It covers comprehensive test and analysis on aspects of performance, compatibility, function, HTTP protocol conformance, security.

**Key words** Micmic defense theory; Gray-Box testing method; exploiting complexity; test principle; test case; test analysis; test

## 1 概述

拟态安全防御[1,2]是针对网络空间攻击成本和防御成本的严重不对称性, 以及当前条件下我国信息领域核心技术与产业基础严重滞后国家安全需求的严峻性所提出的一种安全策略思路, 目标是利用异构性、多样性改变系统相似性、单一性, 利用动态性、随机性改变系统静态性、确定性, 利用非相似余度空间交叉研判未知、未明威胁, 阻断或扰乱网络攻击, 以达成系统安全风险可控的要求。

web 服务器拟态防御原理验证系统[3]是依据拟态安全防御的基本思想, 通过对异构资源差异最大化、web 服务器请求分发表决、非相似 web 虚拟机池调度清洗等关键技术的研究而研制的。作为首创的拟态防御 web 服务器, 在各关键技术、功能模块、异构体之间的相互配合、相互影响等测试项目方面以及 web 服务器整体安全测试方面缺乏相应的测试手段和实验数据来验证目前的原理验证系统是否达

**通讯作者:** 张铮, 博士, 副教授, Email: ponyzhang@126.com。
本课题得到国家重点研发计划(2016YFB0800104), 上海市科学技术委员会科研计划项目(14DZ1105300)和国家自然科学基金(61572520)资助。
收稿日期: 2016-09-13; 修改日期: 2016-12-03; 定稿日期: 2016-12-20

到拟态防御原理应用于 web 服务器的预期效果, 是否满足 web 服务器运行的安全性、可靠性和稳定性要求。传统的测试方法在实施于 web 服务器拟态防御原理验证系统中存在以下几点不足:

(1) 传统的安全性测试方法多集中于对 web 应用的测试, 作为 web 应用的载体, 传统 web 服务器本身不具有安全防护特性, 也不存在 web 服务器本身安全性测试, 多数为针对服务器中入侵检测系统、防火墙等安全组件的测试方法;

(2) web 服务器拟态防御原理系统中采用请求分发和响应表决的设计思路, 引入异构、冗余、动态等特性, 异构性为拟态防御 web 服务器提供了输出表决的基础; 冗余性则提升了整个系统的可靠性; 动态性通过感知错误异常, 清洗还原异常, 因此黑盒测试和白盒测试均不能精准地测试拟态架构中每一环节的安全防御效果;

(3) 在安全性渗透测试中, 只从直观的安全防御效果来评判 web 服务器拟态防御原理验证系统的安全防御能力是不充分、不全面、不科学的, 缺少具有形式化描述的属性值来衡量安全性, 例如当拟态防御 web 服务器成功抵御了 SQL 注入攻击, 攻击者继续渗透攻击获取 SQL 指令异构信息后, 攻击者是否具有攻击成功的可能, 攻击成功难度有多大, 传统的安全性测试是无法衡量的。

因此, 有必要依据拟态防御原理, 基于 web 服务器拟态防御原理样机搭建相应的测试环境, 通过科学合理的测试方法开展测试工作, 以验证和指导拟态防御 web 服务器的研发工作。

## 2 适用于拟态防御架构的 web 服务器测试方法

针对以上问题, 本文提出一种适用于拟态防御架构的 web 服务器安全性测试方法。作为创新型的防御架构, 拟态防御技术为 web 服务器带来了异构、动态、冗余、随机等特性, 在进行安全性测试前, 首先要对整个系统进行全面的基础性能测试工作, 验证拟态架构的引入不会影响 web 服务器的正常运行和用户的正常使用, 因此测试工作主要分为基础性能测试和安全性测试, 测试框架如图 1 所示。

该方法在传统的安全测试基础上, 增加基础性能测试环节, 通过增量式对比测试方法, 逐级测试各具有拟态特性模块的基础性能, 其中包括性能测试、HTTP 通信协议一致性测试、拟态防御原理功能测试、兼容性测试。

在安全性测试中, 本文提出了一种基于让步的灰盒测试方法, 充分验证拟态防御架构的防御效果, 达到安全性测试的目的。灰盒测试介于白盒测试和黑盒测试之间, 不同于白盒测试需要详细、完整地关注被测系统的内部结构, 灰盒测试更关注的是输出对于输入的正确性, 通过一些表征性的现象、事件、标志来判断内部的运行状态, 测试方法范围没有黑盒测试广, 但是覆盖的精度更高。基于让步的灰盒测试中, 被测系统开发人员配合测试人员, 在不破坏拟态防御机制的前提下, 在协助测试阶段, 逐步开放所需条件或者设置所需环境, 使得渗透测试更深入系统内部, 全面覆盖各个模块。例如, 在 web 应用攻击测试中的上传漏洞测试案例中, 部署在 web 服务器的应用中不存在上传漏洞, 为了验证 web 服务器抵御上传漏洞威胁, 系统开发人员按照测试人员要求, 向 web 服务器的目标位置拷贝测试人员提供的恶意脚本; 而后测试人员尝试利用该恶意脚本, 在无法提取系统权限, 不能利用该脚本的情况下, 系统开发人员再次配合测试人员提升用户权限, 赋予恶意脚本可执行权限; 最终, 通过系统开发人员的不断让步配合, “上传”的恶意脚本具有了可触发性, 从而为全面验证 web 服务器抵御攻击者利用上传漏洞攻击目标服务器的防护效果提供了基础。

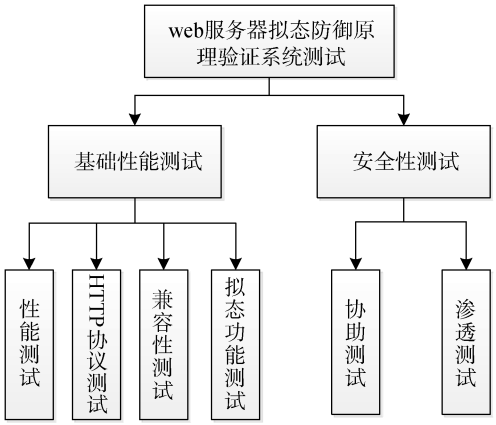


**图 1 web 服务器拟态防御原理验证系统测试框架**

在安全测试中的安全性评估环节, 本文重新定义了漏洞和后门的利用复杂度, 将利用复杂度分为攻击链的复杂度和攻击场景的可再现度两个因素, 并利用这两项度量值评判目标服务器的安全性, 具体内容见本文 8.3 节。

## 3 测试方案设计

### 3.1 测试对象

web 服务器拟态防御原理验证系统依据非相似余度拟态安全原理, 构建功能等价的、多样化的、动

态化的异构虚拟 web 服务器池, 采用响应多余度表决、动态执行体调度、执行体清洗还原、数据库指令异构化等技术, 阻断攻击链, 增大漏洞或后门的利用难度, 保证 web 服务的可用性和安全性。

web 服务器拟态防御原理验证系统是由信息碎片随机化传输模块(information scattering and randomly routing module, ISRR)、请求分发均衡模块(request dispatching and balancing module, RDB)、响应多余度表决器(dissimilar redundant responses voter, DRRV)、非相似 web 虚拟机池(dissimilar virtual web server pool, DVSP)、动态执行体调度器(dynamicly executing scheduler, DES)、web 服务物理主机(physical web server, PWS)、中心调度器(primary scheduler, PS)、数据库指令异构化模块(database instruction labelling module, DIL)组成。web 服务器拟态防御原理验证系统架构如图 2 所示, 其中作为 web 服务主要载体的非相似 web 虚拟机池的操作系统和软件配置如表 1 所示。

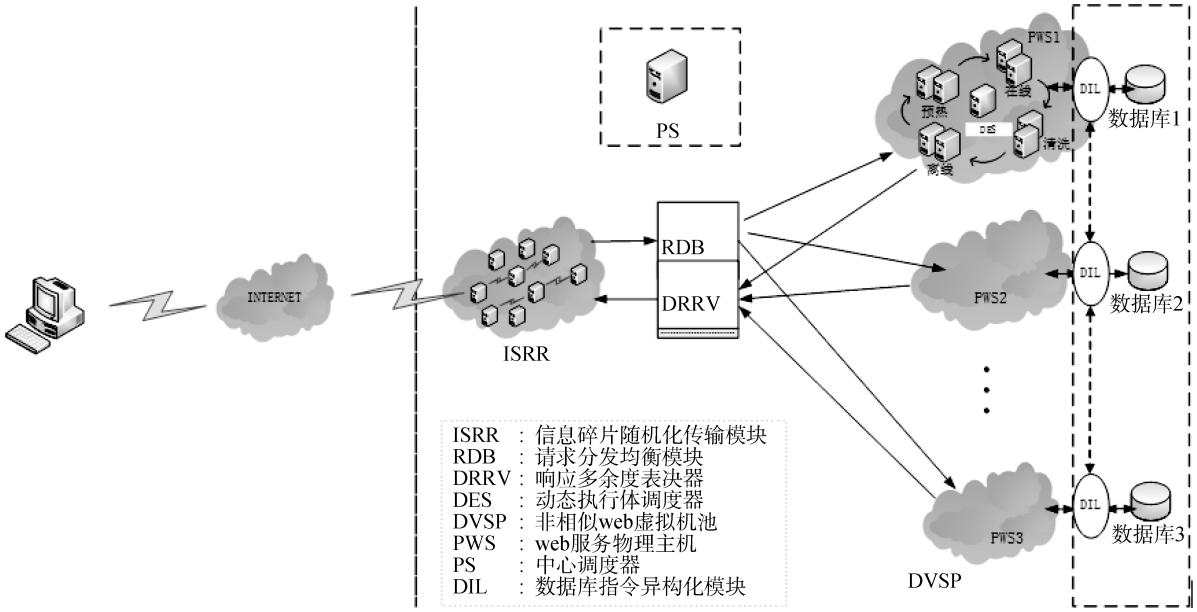


**图 2　web 服务器拟态防御原理验证系统架构**

**表 1　非相似 web 虚拟机池操作系统和软件配置**

| 名称 | 虚拟机服务器系统 | web 服务器软件 |
|---|---|---|
| 非相似虚拟机池 1 | Centos7 | Apache/2.2.15 |
| | Centos7 | Apache/2.2.15 |
| | Centos7 | Nginx/1.9.11 |
| | Centos7 | Nginx /1.9.11 |
| 非相似虚拟机池 2 | RedHat | Apache/2.4.7 |
| | Ubuntu SMP | Apache/2.4.10 |
| | Ubuntu SMP | Nginx /1.9.11 |
| | Ubuntu SMP | Apache/2.4.10 |
| | Ubuntu SMP | Nginx /1.9.11 |
| | RedHat | Nginx /1.9.11 |
| 非相似虚拟机池 3 | WinServer2003 | IIS6.0 |
| | WinServer2008 | Apache2.0 |
| | WinServer2008 | Lighttpd |
| | WinServer2008 | IIS7.0 |
| | WinServer2003 | Lighttpd |

## 3.2　测试方案

为确保测试工作的科学严谨、公开公正, 测试结果具有公信力, 具有代表性, 整个测试项目组将由来自权威科研机构、知名互联网厂商和一流高等院校的人员组成, 测试人员按照以下测试过程开展每项测试样例的测试工作:

(1) 制定测试计划

测试组长与开发人员进行沟通和交流, 针对测试目的, 从用户、维护人员、恶意攻击者等不同角度进行考虑, 使用合适的测试方法和制定科学的测试计划, 并选取合适人员组建测试小组。

(2) 设计测试用例

测试人员在熟悉测试对象后, 设计科学的测试用例, 其中包括针对测试需求选取合适的测试工具, 编写有效的测试脚本等, 为测试执行阶段准备好必要条件。

(3) 构造测试环境

因为 web 服务器拟态防御原理验证系统中存在较多的异构环境, 包括多种操作系统、服务器应用软件、数据库、语言等, 这就要求测试人员准确定位本

测试案例的所需环境, 保证测试的顺利展开。

(4) 测试执行

由测试组长检查上述工作的完成情况, 并决定是否开展测试工作。为确保测试的准确性, 每个测试用例执行次数大于等于 2 次。

(5) 测试问题提交

将测试中出现的问题或者不确定结果, 提交给测试组长, 由测试组长进行分析确定解决方案。

(6) 测试记录

测试人员将测试过程和测试结果记录存档, 存档要求符合实际情况, 客观准确, 不夹杂任何主观定论。

### 3.3 测试方法

web 服务系统测试的方法和技术是多种多样的。对于系统测试方法,可以从不同的角度加以分类[4]如图 3 所示:

(1) 从测试是否针对系统的内部结构和具体实现算法的角度来看,可分为白盒测试和黑盒测试;

(2) 从测试目标和质量特性上,可分为性能测试、安全性测试、功能测试、兼容性测试、可靠性测试等;

(3) 从测试阶段或层次上,可分为单元测试 、整体测试、集成测试和验收测试。

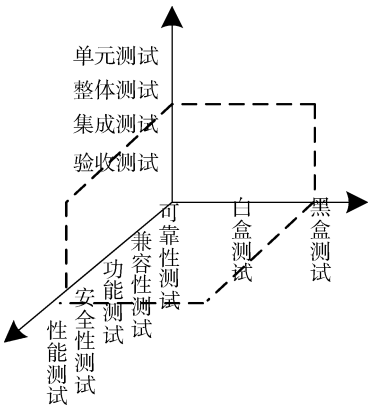


**图 3 测试方法分类三维空间图**

本文将从测试目标和质量特性上开展性能测试、安全性测试、功能测试、兼容性测试和 HTTP 通信协议测试, 根据具体的测试工作制定相应的测试原则并使用合适的测试方法。

### 3.4 设备配置

为避免 web 服务器硬件差异给测试结果带来误差, 本文测试中采用统一的 web 服务器硬件配置, 如表 2 所示。

**表 2 web 服务器硬件配置表**

| 类型 | 配置信息 |
|---|---|
| CPU 型号 | 英特尔 Xeon(R) CPU E5-2620 v3 @ 2.40GHz (1224 线程) |
| 主板 | GIGABYTE MD30-RS0 |
| 显卡 | ASPEED,16MB |
| 磁盘驱动器 | LSI MR9260-8i SCSI Disk Device |
| 内存 | LSI MegaRAID SAS 9260-8i (32G) |
| 网卡 1 | Broadcom BCM57810 NetXtreme II 10 Gige*2 |
| 网卡 2 | Intel(R) I210 Gigabit Network Connection*2 |

## 4 性能测试实施

web 服务是在 HTTP 协议的基础之上实现用户跟服务器之间的信息交换[5]。用户访问 web 服务器可以分为以下几个过程:

(1) 用户的客户端和 web 服务器端使用 TCP 协议建立连接;

(2) 客户端通过 HTTP Get/Post 向 web 服务器端发送请求;

(3) 服务器端响应客户端的请求,发送客户端需要的网页文件;

(4) 客户端收到服务器端发送来的文件后,就会发送一个确定报文给服务器端;

(5) 服务器端收到客户端发送来的确定报文后就会关闭本次的 TCP 连接,然后结束。

评价 web 服务器性能的指标[6]包括四个主要方面:最大 TCP 并发连接数(Max Concurrent TCP Connection Capacity, MCTCC)、吞吐量(Throughput)、平均响应时间(Average Response Time, RTT)、每秒事务处理次数(Transactions Per Second, TPS)。

(1) 最大 TCP 并发连接数[7]

IETF 2647 中指出, 该指标为穿过网关的主机之间或主机与网关之间能同时建立的最大 TCP 连接数。

(2) 吞吐量[8]

该指标主要体现网络设备的数据包转发能力, 通常表现为网络设备在不丢包前提下的数据转发能力, 其度量单位为字节, 单位一般使用 kbps。

(3) 平均响应时间

响应时间为服务器端对客户端的请求作出响应所需要的时间, 该指标为服务器端在运行稳定后平均完成一个用户端请求的时长。

(4) 每秒事务处理次数

一个事务是指客户端向服务器端发送请求, 然后服务器端作出响应的过程。该指标体现服务器每秒事务处理的次数, 确定服务器的事务负载能力。

### 4.1　性能测试目标

web 服务器性能测试作为 web 服务器测试中的基础部分, 其测试目标在整个性能测试过程中具有重要的指导作用, 只有明确了测试目标, 测试结果才有意义。其目标可以分为以下两种级别:

(1) 性能指标验证目标。实施性能测试来验证受测 web 服务器是否可以达到指定的系统性能指标, 其中包括最大 TCP 并发连接数、吞吐量、平均响应时间、每秒事务处理次数等。

(2) 性能指标对比目标。实施性能测试来对比基准 web 服务器与受测 web 服务器的性能指标, 验证受测 web 服务器的性能指标变化。本文根据这种目标实施性能测试工作, 验证基准 web 服务器在实施拟态防御架构改造为 web 服务器拟态防御原理验证系统后较基准 web 服务器的性能指标变化, 其中包括最大 TCP 并发连接数、吞吐量、平均响应时间、每秒事务处理次数等。

### 4.2　基准虚拟 web 服务器性能测试

web 服务器拟态防御原理验证系统中组网多使用虚拟机服务器搭建, 故本节的测试案例为了对比虚拟机 web 服务器与 web 服务器拟态原理系统的性能指标设计产生, 测试内容与测试结果如表 3 所示。

对比本节测试案例 1 与测试案例 2 的测试结果, web 服务器部署的页面类型会影响测试结果, 然而测试目的是为了验证 web 服务器本身的性能, 只需控制页面(web 应用)这一参数固定不变, 故采用 1kb 动态页面(吞吐量测试时采用 32kb 动态网页)。对比测试案例 2 与测试案例 3 的测试结果, 数据库的使用会使得 web 服务系统的整体性能有所下降, 但是这种现象均存在于 web 拟态防御原理验证系统和普通 web 服务系统中, 不是引入拟态防御架构导致的, 此外本节测试结果为 4.3 节和 4.4 节测试提供基准。

**表 3　基准虚拟 web 服务器性能测试内容与结果**

| 序号 | 测试内容 | 服务器配置 | 测试结果 |
|---|---|---|---|
| 1 | 单台 web 虚拟机服务器访问性能测试 | 虚拟机软件: Vmware<br>cpu: 双核双线程<br>内存: 2G<br>页面: 1kb 静态页面(吞吐量测试时采用 32kb 静态网页) | TFS:5114<br>MCTCC:186652<br>吞吐量(kbps):472144<br>RTT(ms):0.58 |
| 2 | 单台 web 虚拟机服务器访问性能测试 | 虚拟机软件: Vmware<br>cpu: 双核双线程<br>内存: 2G<br>页面: 1kb 动态页面(吞吐量测试时采用 32kb 动态网页) | TFS:4362<br>MCTCC:179230<br>吞吐量(kbps):465716<br>RTT(ms):4.766 |
| 3 | 单台 web 虚拟机服务器+挂载数据库访问性能测试 | 虚拟机软件: Vmware<br>数据库软件: Mysql<br>cpu: 双核双线程<br>内存: 2G<br>页面: 1kb 动态页面(吞吐量测试时采用 32kb 动态网页) | TFS:708<br>MCTCC:29984<br>吞吐量(kbps):135665<br>RTT(ms):58.814 |

### 4.3　DIL 模块性能测试

web 服务器拟态防御原理验证系统中部署了数据库和数据库代理, 并基于数据库代理开发了系统的 DIL 模块, 故本节的测试样例为了测试 DIL 模块对 web 服务器拟态原理系统整体性能的影响, 测试内容与测试结果如表 4 所示。

**表 4　DIL 模块性能测试内容与结果**

| 序号 | 测试内容 | 服务器配置 | 测试结果 |
|---|---|---|---|
| 1 | 单台 web 虚拟机服务器+数据库代理+挂载数据库访问性能测试 | 虚拟机软件: Vmware<br>数据库软件: Mysql<br>数据库代理: Amoeba<br>cpu: 双核双线程<br>内存: 2G<br>页面: 1kb 动态页面(吞吐量测试时采用 32kb 动态网页) | TFS:612<br>MCTCC:26375<br>吞吐量(kbps):110012<br>RTT(ms):55.67 |
| 2 | 单台 web 虚拟机服务器+数据库代理+挂载数据库+DIL 模块访问性能测试 | 虚拟机软件: Vmware<br>数据库软件: Mysql<br>数据库代理: Amoeba<br>cpu: 双核双线程<br>内存: 2G<br>页面: 1kb 动态页面(吞吐量测试时采用 32kb 动态网页) | TFS:480<br>MCTCC:24002<br>吞吐量(kbps):109343<br>RTT(ms):61.592 |

对比本节测试案例 1 与本节测试案例 2 的测试结果, 其中 DIL 模块的应用会使得每秒事务处理数产生变化, 下降 21.56%, 说明 SQL 指令的异构化处理与执行带来了少部分性能损耗, DIL 模块在设计实现中仍需要优化。

### 4.4 系统整体性能测试

对 web 虚拟机服务器拟态原理系统整体性能进行测试, 测试对象为 RDB+DRRV+DIL+三台 web 虚拟机服务器挂载数据库与数据库代理, 测试内容与测试结果如表 5 所示。

对比 4.3 节测试案例 1 与本节测试案例的测试结果, 并以之前各小节的测试结果为基础, 其中 web 拟态防御原理验证系统的每秒事务处理数和响应时间产生变化, 分别下降 20.26%, 提高 96.23%。然而由 4.3 节的测试结果可知每秒事务处理数值的下降主要因为 DIL 模块的应用, 响应时间的拖长则是因为 RDB 模块和 DRRV 模块的应用。测试结果显示来看, 与无任何防护的基准虚拟 web 服务器相比, web 服务器拟态防御原理验证系统具有一定的性能损耗, 但 109.246ms 毫秒级的响应时间并不会给用户的使用体验带来影响, 在可接受范围内。

**表 5 系统整体性能测试内容与结果**

| 序号 | 测试内容 | 服务器配置 | 测试结果 |
|---|---|---|---|
| 1 | RDB 模块+DRRV 模块+三台 web 虚拟机服务器+数据库代理+挂载数据库+DIL 模块访问性能测试 | 虚拟机软件: Vmware<br>数据库软件: Mysql<br>数据库代理: Amoeba<br>cpu: 双核双线程<br>内存: 2G<br>页面: 1kb 动态页面(吞吐量测试时采用 32kb 动态网页) | TFS:488<br>MCTCC:23139<br>吞吐量(kbps):109458<br>RTT(ms):109.246 |

## 5 HTTP 通信协议一致性测试实施

HTTP 协议是基于请求和响应的也就是我们常说的客户端服务器端交互模式。当一个客户端和服务器端建立连接之后,就可以发送请求给服务器申请服务, 目前为止最新的协议版本为 1.1 版。使用 HTTP 协议进行通信是由一个客户端对服务器上的某个资源进行请求,并且得到服务器端反馈的过程。例如,一个最简单的情况就是客户端和服务器通过一个单独的连接来进行通信。在互联网中, HTTP 是构建在 TCP/IP 通信协议之上被广泛应用的协议。HTTP 连接通用的端口是 80,但是其他的端口同样可以使用, 此外 HTTP 是一个可靠的面向连接的传输协议。

HTTP 定义了客户端与服务器交互的几种不同的方式,其中最基本的通信方式有 4 种[9],分别是 GET、POST、PUT、DELETE, 但在本次测试中, 为了保证测试的完整性, 依据 HTTP/1.1 (RFC 2616, June 1999) [10]对 web 服务器拟态防御原理验证系统开展了全面的 HTTP 通信协议测试, 测试内容及测试结果如表 6 所示。

**表 6 HTTP 通信协议测试内容与结果**

| 序号 | 名称 | 测试内容 | 测试结果 |
|---|---|---|---|
| 1 | GET 请求测试 | 使用 HTTP 协议测试工具对 web 服务器拟态防御原理验证系统发送 GET 请求, 查看是否服务器会返回响应信息(Header 和 Body) | 可以接收 GET 请求并返回正确的响应信息 |
| 2 | POST 请求测试 | 1 号样例同理, 发送 POST 请求 | 可以接收 POST 请求并返回正确的响应信息 |
| 3 | PUT 请求测试 | 1 号样例同理, 发送 PUT 请求 | 可以接收 PUT 请求并返回正确的响应信息 |
| 4 | DELETE 请求测试 | 1 号样例同理, 发送 DELETE 请求 | 可以接收 DELETE 请求并返回正确的响应信息 |
| 5 | HEAD 请求测试 | 1 号样例同理, 发送 HEAD 请求 | 可以接收 HEAD 请求并返回正确的响应信息 |
| 6 | OPTIONS 请求测试 | 1 号样例同理, 发送 OPTIONS 请求 | 可以接收 OPTIONS 请求并返回正确的响应信息 |
| 7 | TRACE 请求测试 | 1 号样例同理, 发送 TRACE 请求 | 可以接收 TRACE 请求并返回正确的响应信息 |
| 8 | 持久性连接测试 | 通过测试仪向网页发起请求, 并观察 TCP 连接情况 | 支持 HTTP1.1 持久性连接 |
| 9 | 非持久连接测试 | 通过测试仪向网页发起请求, 并观察 TCP 连接情况 | 支持 HTTP1.1 非持久性连接 |

## 6 拟态防御原理功能测试实施

web 服务器拟态防御原理验证系统是由信息碎片随机化传输模块、请求分发均衡模块、响应多余度表决器、非相似 web 虚拟机池、动态执行体调度器、web 服务物理主机、中心调度器、数据库指令异构化模块等组成, 本节将测试系统中各模块功能是否符合拟态原理, 测试内容及测试结果如表 7 所示。

**表 7　拟态防御原理功能测试内容与结果**

| 序号 | 测试对象 | 测试方法 | 测试结果 |
|---|---|---|---|
| 1 | 信息碎片随机化传输模块 | 随机配置三组端口, 请求发生后观察响应信息的路径 | 每组端口响应总数相同, 但不同组端口响应比例不同 |
| 2 | 请求分发均衡模块 | 抓取前端出口数据, 观察数据目的地址及内容 | 输出到每个虚拟机服务器池的数据包相同 |
| 3 | 响应多余度表决器 | 在三个在线虚拟机服务器池中, 更改其中一个池中的网页, 观察变化 | 网站正常提供服务, 并且服务器控制端返回错误信息 |
| 4 | 动态执行体调度器 | 限定虚拟机服务器池中有 5 台设备, 且全部处于在线状态, 更改其中某台在线服务器的网页, 观察变化 | 基于异常事件驱动, 更改过的服务器被清洗恢复 |
| 5 | 中心调度器 | 切换不同的调度策略, 观察系统是否能正常运行 | 系统正常运行 |
| 6 | 数据库指令异构化模块 | 是否有指令指纹异构能力、指令离线判决能力、维持数据库一致性能力 | 测试能力全部通过 |
| 7 | 应急处置能力测试 | 某虚拟机服务器池故障断电 | 系统正常运行 |

以上测试结果证明, web 服务器拟态原理验证系统中各模块功能均符合拟态防御原理。

## 7　兼容性测试实施

用户体验是 web 服务器着重考虑的因素, 由于主流浏览器的不断扩充(包括 IE、Firefox、Chrome 等), 显示器设备的不断丰富, 使得 web 服务器系统需要兼容各种用户环境, 在本章节测试工作中, 对比 web 服务器拟态防御原理验证系统与普通服务器提供的 web 页面, 判断是否存在页面显示完整性、页面排版布局、网页功能、JS 兼容性、不同分辨率下网页布局方面的差异, 测试内容及测试结果如表 8 所示。

结果证明, web 服务器拟态原理验证系统不会存在网页兼容性方面的差异。

**表 8　兼容性测试内容与结果**

| 序号 | 测试内容 | 测试环境 | 测试结果 |
|---|---|---|---|
| 1 | 页面显示完整性测试 | 操作系统+浏览器:<br>Linux+Chrome、Win8+Chrome、Win7+Chrome、Win10+Chrome、Linux+Firefox、Win8+ Firefox、Win7+Firefox、Win XP+ Firefox、Win7+IE、Win8+ IE、Win XP+IE | 系统页面显示完整性方面与普通 web 服务器表现一致 |
| 2 | 网页功能完整性测试 | 操作系统+浏览器:<br>Linux+Chrome、Win8+Chrome、Win7+Chrome、Win10+Chrome、Linux+Firefox、Win8+ Firefox、Win7+Firefox、Win XP+ Firefox、Win7+IE、Win8+ IE、Win XP+IE | 系统页面功能完整性方面与普通 web 服务器表现一致 |
| 3 | JS 脚本兼容测试 | 操作系统+浏览器:<br>Linux+Chrome、Win8+Chrome、Win7+Chrome、Linux+Firefox、Win8+ Firefox、Win7+Firefox、Win XP+ Firefox、Win7+IE、Win8+ IE、Win XP+IE | 系统 JS 脚本兼容性与普通 web 服务器表现一致 |
| 4 | 排版布局测试 | 操作系统+浏览器:<br>Linux+Chrome、Win8+Chrome、Win7+Chrome、Win XP+Chrome、Linux+Firefox、Win8+ Firefox、Win7+Firefox、Win XP+ Firefox、Win7+IE、Win8+ IE、Win XP+IE | 系统网页排版布局与普通 web 服务器表现一致 |
| 5 | 不同分辨率下的网页布局 | 操作系统+浏览器+分辨率:<br>Linux+Chrome+w-800 h-600、Linux+Chrome+w-1280 h-720、Linux+Chrome+w-1855 h-1056、Linux+Firefox+w-800 h-600、Linux+Firefox+w-1280 h-720、Linux+Firefox+w-1855 h-1056、Win7+Firefox+w-800 h-600、Win7+Firefox+w-1280 h-720、Win7+Firefox+w-1920 h-1080、Win7+Firefox+w-1928 h-1044、Win7+IE+w-800 h-600、Win7+IE+w-1280 h-720、Win7+IE+w-1920 h-1080、Win7+IE+w-1928 h-1044 | 系统在不同分辨率下网页布局与普通 web 服务器表现一致 |

## 8 安全性测试实施

拟态安全防御原理旨在提高系统的安全性, web 服务器拟态防御原理验证系统正是在研究了传统 web 服务器及其经常面临的安全威胁的基础上, 结合拟态安全防御的基本思想研制而成的基于不安全组件的 web 服务器, 其能够允许漏洞和后门存在的情况下, 仍能提供安全可靠的服务; 并且从根本上改变传统网络安全防御方法, 阻断利用漏洞或后门的攻击链; 达到能够有效防御未知后门和漏洞的效果, 因此, 安全性测试成为了本次测试工作中关键。

### 8.1 安全性测试原则

基于拟态防御原理的 web 服务器作为新型的 web 安全防御手段, 为验证其防御效果有效性, 保证测试结果的完整性和客观性, 测试过程中制定以下原则:

(1) 被测系统开发人员不得进行增量开发;

(2) 确认受测系统本征功能和性能满足国家或行业标准前提下开展测试工作;

(3) 被测系统不采用现有的安全防护手段, 例如防火墙、入侵检测系统等;

(4) 因为利用未知漏洞和后门发动未知攻击的测试场景无法构造, 所以在不采取已有安全防护手段的情况下, 部署已知漏洞和后门, 来模拟未知漏洞和后门。

### 8.2 安全性测试标准

由于安全性测试是一个比较复杂的系统工程, 目前有许多不同的测试标准, 例如, 美国的信息安全测试和评估技术指南[11]、德国联邦信息安全办公室发布的渗透测试模型[12] (A penetration test model)、web 安全组织 OWASP 发布的测试指南[13] (OWASP Testing Guide V 3.0)、开源安全测试方法指南[14] (The Open Source Security Testing Methodology Manual)等, 虽然不同的标准是从理论的角度整体上对不同的安全问题进行了分类总结, 但不同的标准对安全问题的分类不尽相同。本文选取了《GB/T 20984-2007 信息安全技术信息安全风险评估规范》[15]、《GB/T 30279-2013 信息安全技术安全漏洞等级划分指南》[16]、《通用漏洞评分标准》[17]和《GB/T 18336-2008 信息技术安全性评估准则》作为本次安全性测试的主要测试标准。

然而 web 服务器拟态防御原理验证系统作为新兴的安全防御系统, 已有的安全测试标准并不完全适用, 于是在安全性测试过程中需要测试人员参照标准, 设计合适的防御效果评判标准, 本文丰富了原有的漏洞利用复杂度含义, 从而以此评判 web 服务系统的安全防御能力。

### 8.3 漏洞和后门的利用复杂度

漏洞和后门的等级划分主要依据三个因素[18]: 访问路径, 利用复杂度, 影响程度。

访问路径: 攻击者利用安全漏洞影响目标系统的路径前提, 分为“远程”,“邻接”和“本地”三种方式。

利用复杂度: 安全漏洞被利用影响目标系统的难度, 分为“简单”和“复杂”。

影响程度: 利用安全漏洞对目标系统造成的损害程度, 针对机密性、完整性和可用性(CIA)某一方面的影响程度, 分为“完全”、“部分”、“轻微”和“无”四种情况。

虽然以上的漏洞和后门等级划分标准相对全面, 但划分较为粗略, 对于本次测试, 无法通过已有的等级标准来评估 web 服务系统的安全防御能力, 因而本文对利用复杂度这一因素做出新的划分。

将利用复杂度划分为以下两个组成因素:

攻击链的复杂度(Complexity of Attack Chain, C): 利用漏洞或后门进行攻击的过程所形成的有向图的复杂程度, 由攻击链中每一攻击步骤的实施代价决定。C 值越大, 证明攻击目标部署于的 web 服务系统抵御该种攻击的能力越强;

攻击场景的可再现度(Re-exploitability, RE): 漏洞或后门被再次成功利用的概率, 可根据实际测试结果取概率值, 也可划分为“不可再现”“偶然”和“必然”三种等级。RE 值越小, 证明攻击目标部署于的 web 服务系统抵御该种攻击的能力越强。

以图 4 所示攻击链为例, 假设利用服务器软件的配置漏洞篡改了服务器的页面, 攻击的最终结果为篡改成功, 影响了服务器的完整性。

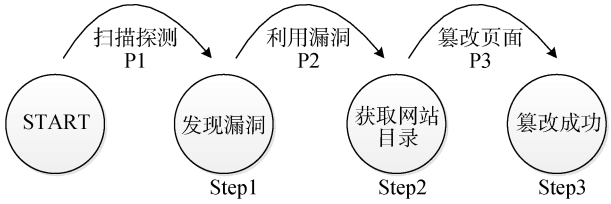


**图 4 普通 web 服务器中的篡改网页攻击链**

图中的攻击链, 第 $i$ 攻击步骤的攻击代价为 $P_i$ ($0<P_i\leqslant 1$), $P_i$ 越大攻击难度越大, 则攻击链的复杂度 $C$, 攻击场景的可再现度 $RE$ 分别为:

$$C=\sum_{i=1}^{3}P_i\ ,$$

$$RE=1$$

在 web 服务器拟态防御原理验证系统中, 由于异构、冗余、动态、清洗机制的防御作用, 攻击链的复杂度发生了变化, 如图 5 所示。

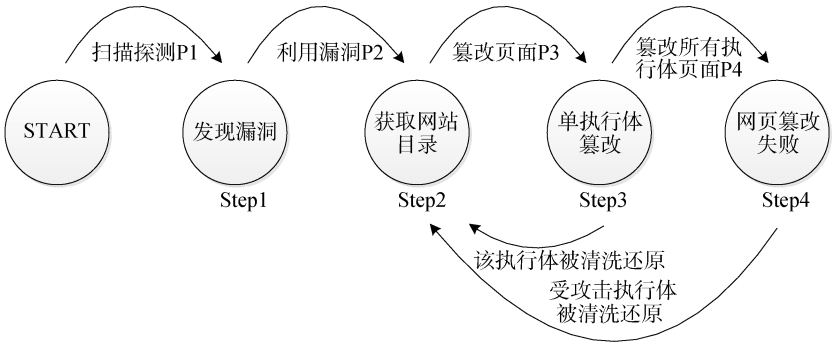


**图 5　web 服务器拟态防御原理验证系统中的网页篡改攻击链**

因为动态机制和清洗机制的存在, 增加了攻击多重循环的代价, 攻击链的复杂度 $C$ 为:

$$C=\sum_{i=1}^{2}P_i+n\times(m\times P_3+P_4)$$

其中 $n=+\infty$

因为攻击不能成功, 所以 $RE=0$。

## 8.4　测试过程

在测试过程中, 某些测试是多个环节进行的, 完成某一环节是进行后一环节的前提条件, 因此, 为有效开展后续环节的测试工作, 这就要求系统开发人员公开系统组成结构、实现细节、进行人为设置漏洞、后门、病毒、木马等操作, 从攻击角度来评估拟态防御技术的有效性。在本次安全性测试中, 根据是否采用基于让步规则的灰色测试，是否需要系统开发人员配合协助将测试工作分为协助测试阶段和互联网渗透测试阶段。

### 8.4.1　web 安全协助测试

测试阶段中通过白盒测试和基于让步规则的灰盒测试等开放式手段, 在相同的测试条件下, 比较受测系统和关闭拟态防御功能的普通 web 服务器之间的受攻击效果差异, 来评估拟态防御技术的有效性。根据攻击类型、受测主体、关键模块、系统结构等分类制定了相应的测试案例, 共抽选漏洞 7 类 11 种, 例如: 恶意文件上传漏洞、文件包含漏洞、命令执行漏洞、路径暴露漏洞、SQL 注入漏洞、目录遍历漏洞、解析漏洞等; 抽选木马 2 类 21 种, 例如: Windows 远控木马、webshell 小马、webshell 大马等; 抽选病毒 5 类 21 种, 例如: 恶意弹窗致资源耗尽病毒、服务器关键信息泄露病毒、系统致瘫病毒等。使用了中国菜刀、Nmap、Wireshark、Burpsuite、Http Fuzzer、Weevely 客户端、sqlmap、Darkcomet-RAT 等多种流行攻击工具。

### 8.4.2　web 安全互联网渗透测试

web 安全互联网渗透测试[19]是完全模拟黑客可能使用的漏洞发现技术和攻击技术, 对目标 web 服务系统的安全防御能力做深入的测试, 将受测设备暴露在更真实的使用环境中, 从而发现系统最脆弱的环节。测试一般是经过客户授权的, 采用可控制的方法和手段发现目标服务器、web 应用程序和网络配置中存在的弱点, 其中包括黑盒测试和白盒测试, 最终本阶段测试根据测试人员模拟黑客的攻击效果来验证 web 服务器拟态防御原理验证系统的防御效果。

因为 web 安全渗透测试流程完全模拟黑客攻击流程, 所以渗透测试的流程从黑客攻击 web 的角度可以分为探测扫描、发动攻击、植入后门、消除痕迹四大步骤, 依据此原则, 此阶段测试人员分别进行了恶意扫描探测测试、SQL 注入攻击测试、Weevely 工具入侵测试、中国菜刀工具入侵测试、预置后门测试、植入木马测试, 全部测试中测试人员均不能达到预期的攻击效果, 证明了 web 服务器拟态防御原理验证系统的防御有效性。

## 8.5　安全性测试结果分析

本节汇总协助测试阶段与渗透测试阶段的测试结果, 将所有测试案例分为 5 方面的测试内容, 分别是扫描探测测试、抗病毒木马测试、web 应用攻击测试、数据安全测试和操作系统安全测试, 又将每个测试内容根据攻击方式分为不同的测试类别。还通过测试结果分析发挥防御作用的拟态防御机理, 确定拟态防御机理发挥作用后系统漏洞或后门的状态(在线/离线)。

恶意攻击者通过扫描探测, 可以获取目标系统的关键信息, 从而制定攻击方案。若 web 服务系统能够抵御恶意攻击者的扫描探测行为, 则会大大提升

web 服务系统的安全防护能力, 从而削弱恶意攻击者的破坏能力, 扰乱其攻击计划。在扫描探测测试中, 根据获取目标系统的信息类别划分为指纹信息扫描测试和漏洞信息扫描测试, 测试结果如表 9 所示。

**表 9　扫描探测测试结果**

| 测试内容 | 测试类别 | 测试结果 | 防御机理 | | | | 漏洞或后门状态 |
|---|---|---|---|---|---|---|---|
| | | | 异构 | 冗余 | 动态 | 清洗 | |
| 扫描探测测试 | 指纹信息扫描测试 | 测试通过, 执行体间组成异构导致指纹信息不一致, 输出结果无法通过表决 | √ | √ | √ | | 在线 |
| | 漏洞信息扫描测试 | 测试通过, 执行体间间组成异构漏洞信息不一致, 输出结果无法通过表决 | √ | √ | √ | | 在线 |

病毒、木马作为恶意攻击者的有利武器, 危害大, 变种多, 涉及范围广。若 web 服务系统能够抵御病毒木马的攻击行为, 则强有力地遏制大部分攻击的发生。在抗病毒木马测试中, 根据病毒木马的攻击方式类别划分为木马连接测试、木马执行测试、恶意弹窗致资源耗尽病毒测试、服务器信息泄露病毒测试、系统致瘫测试和网站内容篡改测试等, 测试结果如表 10 所示。

web 应用作为 web 服务系统提供给用户最重要的内容, 直接暴露给用户。正因为此, web 应用成为了恶意攻击者最直接的攻击目标。同时, web 应用开发时由于开发人员的疏漏, 或多或少会存在安全漏洞。若 web 服务系统能够抵御针对 web 应用的攻击, 则建立起了 web 服务系统的第一道安全屏障。在 web 应用攻击测试中, 根据常见的攻击类型分为目录配置漏洞测试、SQL 注入测试、解析漏洞测试、文件包含测试和 DoS 漏洞测试, 测试结果如表 11 所示。

**表 10　抗病毒木马测试结果**

| 测试内容 | 测试类别 | 测试结果 | 防御机理 | | | | 漏洞或后门状态 |
|---|---|---|---|---|---|---|---|
| | | | 异构 | 冗余 | 动态 | 清洗 | |
| 抗病毒木马测试 | 木马连接测试 | 测试通过, 执行体间操作系统不一致, 连接信息无法通过表决 | √ | √ | | √ | 离线 |
| | 木马执行测试 | 测试通过, 执行体间操作系统不一致, 执行结果无法通过表决 | √ | √ | | √ | 离线 |
| | 恶意弹窗致资源耗尽病毒测试 | 测试通过, 执行体间操作系统不一致, 病毒无法在全部执行体中成功执行 | √ | √ | | √ | 离线 |
| | 服务器信息泄露病毒测试 | 测试通过, 执行体间服务器信息不一致, 输出结果无法通过表决 | √ | √ | | √ | 离线 |
| | 系统致瘫病毒测试 | 测试通过, 执行体间操作系统不一致, 病毒无法在全部执行体中成功执行 | √ | √ | | √ | 离线 |
| | 网站内容篡改病毒测试 | 测试通过, 执行体间操作系统不一致, 病毒无法在全部执行体中成功执行 | √ | √ | | √ | 离线 |

**表 11　web 应用攻击测试结果**

| 测试内容 | 测试类别 | 测试结果 | 防御机理 | | | | 漏洞或后门状态 |
|---|---|---|---|---|---|---|---|
| | | | 异构 | 冗余 | 动态 | 清洗 | |
| web 应用攻击测试 | 目录配置漏洞测试 | 测试通过, 执行体间服务器软件不一致, 并非所有执行体存在目录配置漏洞, 攻击无法在全部执行体中成功执行 | √ | √ | | √ | 离线 |
| | SQL 注入测试 | 测试通过, 执行体间 SQL 指纹不一致, 攻击无法在全部执行体中成功执行 | √ | √ | √ | √ | 在线 |
| | 解析漏洞测试 | 测试通过, 执行体间服务器软件不一致, 并非所有执行体存在解析漏洞, 攻击无法在全部执行体中成功执行 | √ | √ | | √ | 离线 |
| | 上传漏洞测试 | 测试通过, 执行体操作系统、服务器软件等层次存在不一致, 无法通过上传的脚本发动攻击 | √ | √ | | √ | 在线 |
| | 文件包含测试 | 测试通过, 执行体操作系统、服务器软件等层次存在不一致, 无法通过包含执行的脚本发动攻击 | √ | √ | | √ | 在线 |
| | DoS 漏洞测试 | 测试通过, 执行体间服务器软件不一致, 并非所有执行体存在 DoS 漏洞, 攻击无法在全部执行体中成功执行 | √ | √ | | √ | 离线 |

数据作为恶意攻击者的重要获取目标，如何保护数据安全也是web服务系统面临的重要安全问题。在数据安全测试中，根据获取或破坏数据的不同方式分为传输节点嗅探测试、传输节点致瘫测试、网站目录提取测试、SQL 指令指纹破坏测试和表决器逻辑逃逸测试，测试结果如表 12 所示。

**表 12　数据安全测试结果**

| 测试内容 | 测试类别 | 测试结果 | 防御机理 | | | | 漏洞或后门状态 |
|---|---|---|---|---|---|---|---|
| | | | 异构 | 冗余 | 动态 | 清洗 | |
| 数据安全测试 | 传输节点嗅探测试 | 测试通过，信息碎片随机化传输模块中数据进行节点随机化传输，单一节点嗅探无法得到全部信息 | | √ | √ | | 在线 |
| | 传输节点致瘫测试 | 测试通过，信息碎片随机化传输模块中数据进行节点随机化传输，单一节点致瘫不影响信息传输 | | √ | √ | | 在线 |
| | 网站目录提取测试 | 测试通过，网站目录中生成随机数据，造成网站目录信息不一致，输出结果无法通过表决 | √ | | √ | | 在线 |
| | SQL 指令指纹破坏测试 | 测试通过，SQL 指令动态随机切换，被破坏执行体被清洗还原 | √ | | √ | | 离线 |
| | 表决器逻辑逃逸测试 | 测试通过，执行体动态切换，无法发起可靠、持续的协同逃逸攻击 | | √ | √ | | 离线 |

操作系统作为较底层的软件栈，承载着上层软件栈的稳定运行和正常使用，若 web 服务器操作系统被恶意攻击，将带来巨大的安全威胁。在操作系统安全测试中，根据常见的攻击类型分为操作系统提权测试、操作系统控制测试、操作系统文件目录泄露测试和操作系统致瘫测试，测试结果如表 13 所示。

**表 13　操作系统安全测试结果**

| 测试内容 | 测试类别 | 测试结果 | 防御机理 | | | | 漏洞或后门状态 |
|---|---|---|---|---|---|---|---|
| | | | 异构 | 冗余 | 动态 | 清洗 | |
| 操作系统安全测试 | 操作系统提权测试 | 测试通过，执行体操作系统不一致，并非所有执行体存在提权漏洞，且提权方法不一致，攻击无法在全部执行体中成功执行 | √ | √ | | √ | 离线 |
| | 操作系统控制测试 | 测试通过，执行体操作系统不一致，系统控制程序依赖系统环境，且控制方法不一致，攻击无法在全部执行体中成功执行 | √ | √ | | √ | 离线 |
| | 操作系统文件目录泄露测试 | 测试通过，文件目录中生成随机数据，造成文件目录信息不一致，输出结果无法通过表决 | √ | | √ | | 离线 |
| | 操作系统致瘫测试 | 测试通过，执行体操作系统不一致，并非所有执行体存在致瘫漏洞，且致瘫方法不一致，攻击无法在全部执行体中成功执行 | √ | √ | | √ | 离线 |

结合协助测试阶段与渗透测试阶段的结果，本文基于以下原则选取其中的典型案例进行结果分析，来证明 web 服务器拟态原理验证系统的防御效果。

(1) 依据攻击类型或阶段的安全威胁，选择其中具有代表性的案例进行分析;

(2) 依据攻击的广泛性和危害性，选择其中具有代表性的案例进行分析;

(3) 为了进行结果分析，聚焦安全防御基本点，所有案例分析均可开放前置条件、跳过前置步骤，直接对特定攻防环节进行分析。

根据以上原则，结合 2015 年漏洞攻击类型分布情况选取分布广、危害大的扫描探测、SQL 注入攻击测试、服务器软件漏洞攻击测试、命令注入攻击测试的测试结果进行分析。

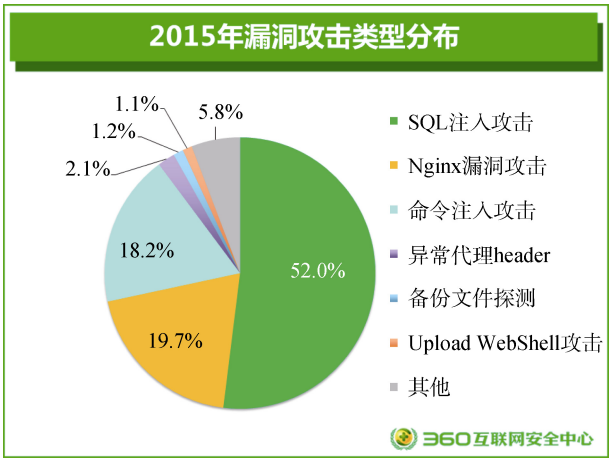


**图 6　2015 年漏洞攻击类型分布图**[20]

### 8.5.1 扫描探测

作为黑客发起网络攻击的起点, 扫描探测是网络攻击中不可或缺的一个环节, 通过探测扫描阶段, 黑客可以获取目标系统的指纹信息、漏洞类型等重要信息, 并以此制定相应的攻击方案。本文设计如下测试案例测试 web 服务器抵御恶意扫描探测系统指纹信息的能力。

操作步骤:

(1) 准备一组 web 服务执行体, 安装相同的 web 应用, 关闭拟态防御功能;

(2) 对 web 服务器拟态防御原理验证系统进行正常访问、扫描探测等操作;

(3) 对步骤(2)中探测重复进行 5 次, 查看扫描探测结果和 web 服务执行体请求接入情况;

(4) 开启拟态防御功能, 重复上述操作;

(5) 结果比较

如图 7 所示, 关闭拟态防御功能时, 每次扫描结果相同, 而当开启拟态防御功能时, 扫描结果如图 8 所示, 产生了扫描结果不一致现象。

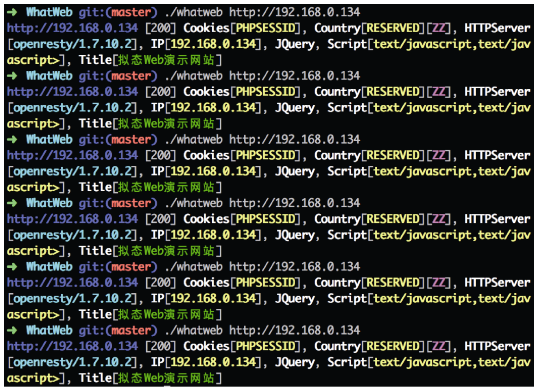


图 7 关闭拟态防御功能时的恶意扫描探测结果

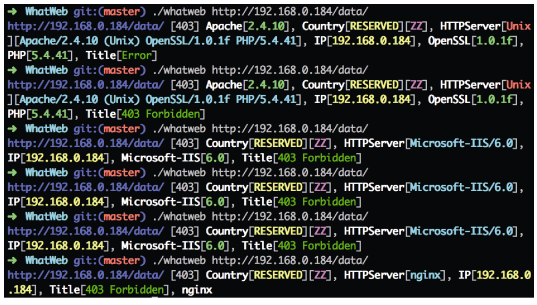


图 8 开启拟态防御功能时的恶意扫描探测结果

web 拟态防御系统中, 执行体的异构冗余与动态特性使得攻击者远程扫描的结果不确定, 从而增加了攻击者的扫描难度, 扰乱了扫描的可锁定性, 从而无法开展下一步的攻击操作, 说明在不消除漏洞和后门前提下改变漏洞和后门的呈现性质。

### 8.5.2 SQL 注入攻击测试

SQL 注入攻击作为主流的攻击方式, 其应用方式主要集中利用服务端口接收用户输入的功能[21], 将构造的语句传给数据库服务器, 让其执行开发者规定外的任务。目前至少有 70%以上的 web 站点存在着 SQL 注入的缺陷, 黑客可以精心构造的非法语句侵入服务器获得网站用户信息和数据库内容, 严重的还可以获得整个服务器所在内网的系统信息。设计如下测试案例测试 web 服务器抵御 SQL 注入攻击的能力。

操作步骤:

(1) 准备一组 web 服务执行体, 关闭拟态防御功能, 利用部署网站上存在的 SQL 注入漏洞, 使用 sqlmap、pangolin 等 SQL 注入测试工具实施基本的 SQL 注入攻击;

(2) 再次利用部署网站上存在的 SQL 注入漏洞, 实施各类高级 SQL 注入攻击;

(3) 开启拟态防御功能, 重复上述操作;

(4) 在攻击失败的情况下, 构造含有某一执行体指纹标签的 SQL 注入语句, 再次发起上述类型的 SQL 注入攻击;

(5) 攻击依然失败, 再次构造含有所有执行体指纹标签的 SQL 注入语句, 再次发起上述类型的 SQL 注入攻击。

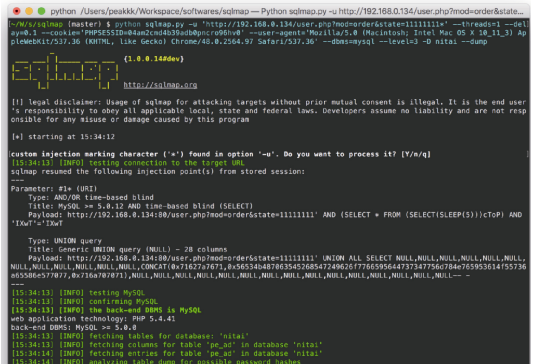


图 9 关闭拟态防御功能时的 SQL 注入攻击结果

如图 9 所示, 关闭拟态防御功能时, 通过 sqlmap 工具进入了数据库中获取了数据库信息, 而当开启拟态防御功能时, 结果如图 10 所示, 攻击失败。

开启拟态防御功能时, SQL 注入攻击链如图 11 所示, 因为 DIL 模块的指令异构作用, 以及指令标签的动态变换, 使得攻击链发生往复循环现象, 并且攻击无法成功, 攻击链的复杂度可计算为:

$$C_1 = \sum_{i=1}^{2} P_i + n \times (m \times P_3 + P_4)$$

其中 $n = +\infty$

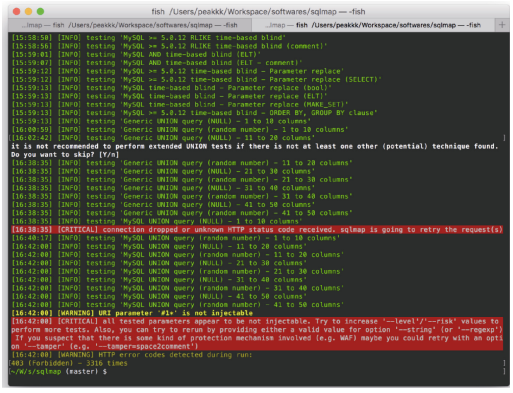

图 10　开启拟态防御功能时的 SQL 注入攻击结果

而在关闭拟态防御功能时，攻击到达攻击链的 step2 步骤时，攻击会成功，此时攻击的复杂度可计算为:

$$C_2 = \sum_{i=1}^{2} P_i$$

计算结果显示 $C_1 >> C_2$，web 服务器拟态防御原理验证系统具有显著的防御效果，DIL 模块实现的基础是 SQL 脚本的异构化，该技术不依赖于 SQL 指令的标签指纹的私密性，而是通过异构后的 SQL 脚本在执行过程中的冗余表决来防御攻击。

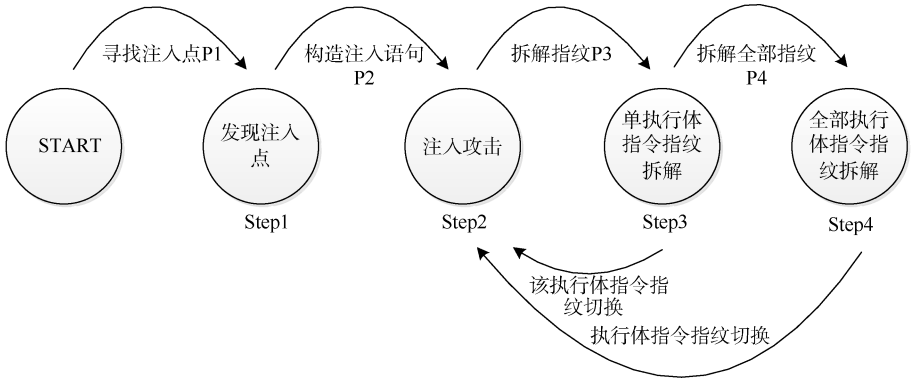


图 11　开启拟态防御功能时的 SQL 注入攻击链

### 8.5.3　服务器软件漏洞攻击

服务器软件在 web 服务系统中扮演着承载 web 应用程序的主要任务，由于如此的基础地位，危害程度较高的漏洞往往存在于服务器软件层。本文设计如下测试案例，通过对已有服务器软件漏洞的防护能力验证 web 服务器抵御针对某特定 web 服务器未知漏洞攻击的能力。

操作步骤:

(1) 准备一组 web 服务执行体并部署具有 hash collision 漏洞的 Apache 服务器、Nginx 服务器和 IIS 服务器，关闭拟态防御功能，对单一 web 服务执行体(Apache)执行 DoS 攻击(基于 Apache 版本的 hash collision)，观察结果，服务崩溃，结果如图 12 所示；

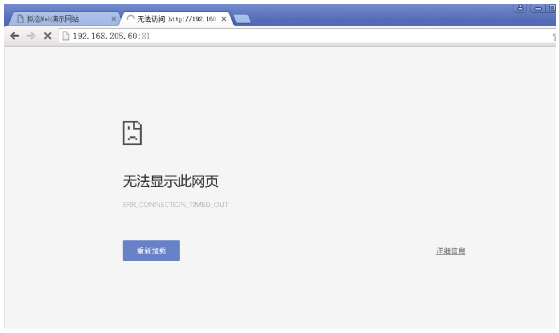


图 12　关闭拟态防御功能时 DoS 攻击结果

(2) 开启拟态防御功能，对网站进行 DoS 攻击(基于 Apache 版本的 hash collision)，观察结果。单一 web 服务执行体(Apache)的拒绝服务信息被屏蔽，系统正常服，结果如图 13 所示;

(3) 结果比较。


图 13　开启拟态防御功能时的 DoS 攻击结果

开启拟态防御功能时，拒绝服务攻击链如图 14 所示，因为异构性和动态机制发挥作用，在具有拒绝服务漏洞的执行体停止服务后，其余在线执行体正常提供服务的同时，该执行体被不具有该漏洞的执行体替换，当攻击者再次动攻击，执行体中该漏洞消失，不得不循环进行扫描探测行为，造成攻击无法成功，攻击链的复杂度可计算为:

$$C_1 = n \times P_1 + P_2$$

其中 $n = +\infty$

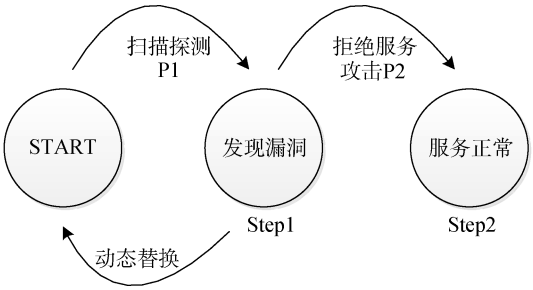


**图 14　开启拟态防御功能时的拒绝服务攻击链**

而在关闭拟态防御功能时, 攻击可以成功, 此时拒绝服务的攻击链如图 15 所示, 攻击链的复杂度可计算为:

$$C_2 = P_1 + P_2$$

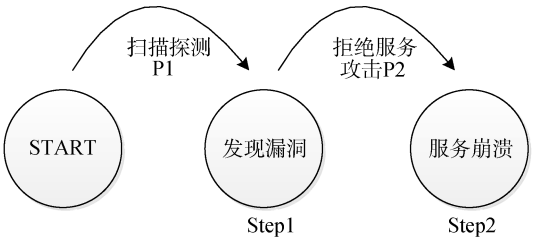


**图 15　关闭拟态防御功能时的拒绝服务攻击链**

计算结果显示 $C_1 >> C_2$, web 服务器拟态防御原理验证系统具有显著的防御效果, 测试人员通过预置的服务器软件漏洞向 web 服务器拟态原理验证系统发起拒绝服务攻击, 由于系统中不同执行体间采用了 IIS、Nginx、Apache 等多种服务器软件, 测试人员发起的拒绝服务攻击只能触发采用 Apache 服务器执行体中的相关漏洞, 而无法破坏整个 web 服务系统。说明 web 服务器拟态原理验证系统可以防御针对特定版本服务器软件发起的恶意攻击, 并在允许漏洞存在, 恶意攻击发生的情况下正常对外提供 web 服务。

### 8.5.4　命令注入攻击

黑客攻击者在获取 web 服务系统权限后通常会采取命令执行攻击来获取受控主机的关键信息或者破坏受控主机系统。本文设计如下测试案例, 通过人为预置后门模拟 web 应用程序漏洞, 测试人员连接后门执行命令注入攻击, 本文设计如下测试案例测试 web 服务器抵御命令执行攻击的能力。

操作步骤:

(1) 准备一组 web 服务执行体, 开启拟态防御功能;

(2) 向单一执行体中植入 php 木马;

(3) 使用 Weevely 客户端尝试连接木马, 连接失败;

(4) 向所有执行体中植入 php 木马;

(5) 使用 Weevely 客户端尝试连接木马, 连接成功后执行 dir 命令, 命令执行失败如图 16 所示。

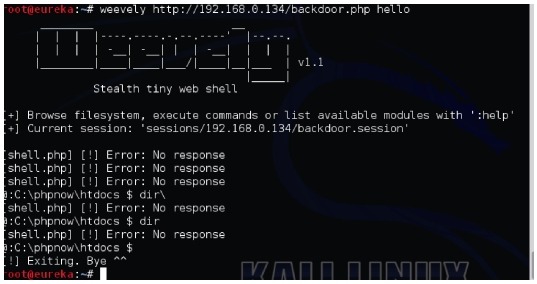


**图 16　命令执行攻击结果**

开启拟态防御功能时, 命令执行攻击链如图 17 所示, 因为动态清洗机制发挥作用, 在木马连接失败和命令执行失败后, 存在木马的执行体被清洗还原, 攻击者需要重新发动攻击, 使得攻击链发生往复循环现象, 并且攻击无法成功, 攻击链的复杂度可计算为:

$$C_1 = n \times (k \times P_1 + P_2 + m \times P_3 + P_4)$$

其中 $n = +\infty$

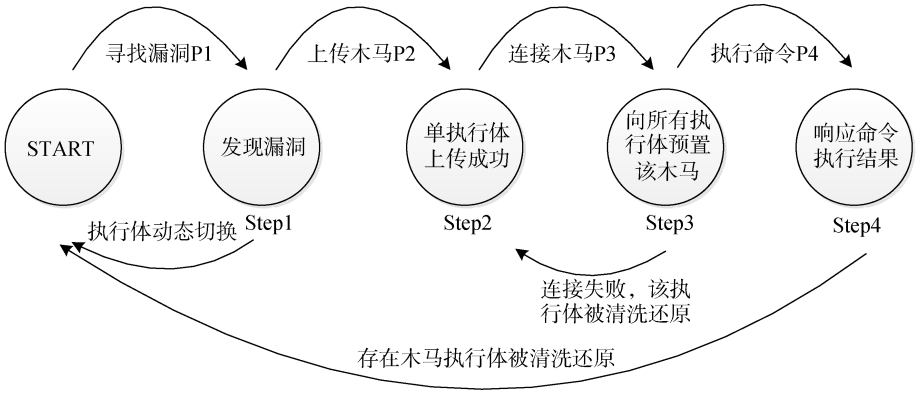


**图 17　开启拟态防御功能时的命令执行攻击链**

而在关闭拟态防御功能时, 攻击可以成功, 此时命令执行的攻击链如图18所示, 攻击链的复杂度可计算为:

$$C_2 = \sum_{i=1}^{4} P_i$$

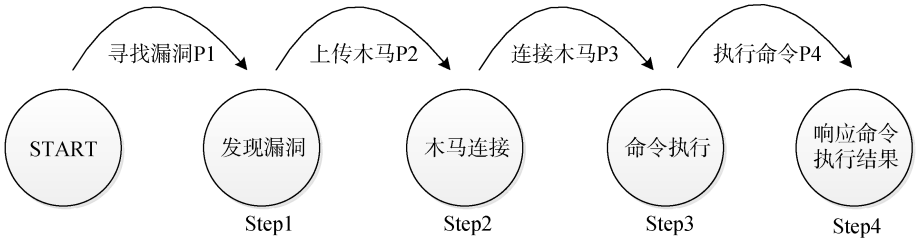


**图 18　关闭拟态防御功能时的命令执行攻击链**

计算结果显示 $C_1 >> C_2$, 同样可证明 web 服务器拟态防御原理验证系统具有显著的防御效果, 因为执行体间的操作系统不同, 命令语句也存在差异, 例如在执行获取网卡信息命令时, Linux 系统中对应的命令是 ifconfig, Windows 系统中则需要执行 ipconfig 命令, 然而当需要执行的命令一致时, 因为各执行体间的响应结果不一致, 经过表决, 测试人员无法得到相应的输出结果, 导致攻击失败, 如上述案例结果所示。说明操作系统层的异构环境, 不仅可以防范针对单一操作系统漏洞的攻击, 还能防御攻击者入侵系统后所执行的命令, 这是在分层的异构环境中, 较下层面的异构特性为针对上层攻击带来的安全防御增益。

## 9　测试分析

web 服务器拟态原理验证系统测试中进行了 7 类 70 项 161 例测试, 其中性能测试包括 15 项 15 例、HTTP 通信协议一致性测试包括 7 项 7 例、兼容性测试包括项 59 例、拟态防御原理功能测试包括 9 项 18 例、接入测试包括 1 项 1 例, web 安全协助测试包括 25 项 55 例、web 安全互联网渗透测试包括 6 项 6 例, 所有测试案例通过。

通过对测试结果进行分析, 与普通 web 服务器相比, 因为 DIL 模块的应用使得 web 服务器拟态防御原理验证系统在性能指标上有所下降, 说明在该功能模块设计上存在不足, 关键算法需要优化, 为今后的研制工作提供了借鉴意义。性能测试中参照基准为不具有任何安全防护措施的普通 web 服务器, 虽然 web 服务器拟态防御原理验证系统在性能方面有较小的下降, 但用不影响用户使用的性能损耗代价换取了更显著的安全防护效果。为使得性能测试更为准确有效, 在下阶段的性能测试中可以与采取了安全防护措施的 web 服务器进行对比测试。

安全性测试则验证了拟态防御原理应用于 web 服务器的可行性, 并能够达到改变漏洞或后门的呈现形式, 阻断大多数针对 web 服务器漏洞或后门攻击的响应链, 增大漏洞或后门利用难度的防御效果。

综合测试结果来看, web 服务器拟态防御原理验证系统在满足通用 web 服务器功能、性能标准的同时, 能够抵御动态异构冗余结构中基于漏洞、后门等的已知风险和未知威胁, 其叠加与迭代效应能够非线性地增加攻击难度。除此之外, 因为动态性、冗余性的防御作用, 在动态异构冗余结构中实现可靠的、持续的协同逃逸攻击变得几乎不可能。

## 10　结论

本文提出了一种适用于拟态防御架构的 web 服务器测试方法, 通过在 web 服务器拟态防御原理验证系统中实施开展, 从获得的测试结论能够证明该方法的合理有效性, 同时也证明了拟态防御思想的正确性; 基于让步规则的灰盒测试方法既充分发挥了测试样例的阶段性渗透性, 又全面地验证了拟态架构的防御有效性; 丰富了漏洞或后门的利用复杂度, 并以此评估 web 服务器安全性的方法, 也为新型的 web 安全防护系统测试工作提供了指导作用。下一步将通过形式化描述的方法来验证拟态防御架构的防御有效性。

**致　谢**　感谢在本文涉及的测试工作中, 国家信息技术安全研究中的徐茜, 中国科学院信息工程研究所的汪秋云、周振飞, 第 61 研究所的吕宇鹏, 中国信息通信研究院的崔涛、廖璇, 上海交通大学的林培胜、潘思远、胡嘉熙, 浙江大学的郑秋华, 北京奇虎科技有限公司的张冬岩、丁俊文, 启明星辰信息安全技术有限公司的贾文晓、张伟, 安天科技股份有限公司的崔宇楠、王鹏所做出的贡献。

## 参考文献


[1] J.X. Wu, “Meaning and Vision of Mimic Computing and Mimic Security Defense,” Journal of Telecommunications Science, vol.30, no. 7, pp. 1-7 (in Chinese), 2014.
(邬江兴, “拟态计算与拟态安全防御原理的愿意和愿景”, 电信科学, 2014, 30(7): 1-7.)

[2] J.X. Wu, “Cyberspace Mimic Defense”. Technical report, National Digital Switching System Engineering & Technological R&D Center, 2015.
(邬江兴, “网络空间拟态防御”, 技术报告, 国家数字交换系统工程技术研究中心, 2015。)

[3] “The prototype of cyberspace mimic defense in web servers”, Technical report, State Key Laboratory of Mathematical Engineering and Advanced Computing, 2015.
( “web 服务器拟态防御原理验证系统”. 技术报告, 国家数字交换系统工程技术研究中心, 2015。)

[4] W.T. Lu, “The Test and analysis of Web application system[Master dissertation],”Beijing Jiaotong University, Beijing, 2011.

[5] T. Berners-Lee, L. Mas inter and M. McCahill. Uniform Resource Locators(URL). RFC1738, IETF, December 2004.

[6] Alberto Avritzer, JW.eyuker. “The Role of Modeling in the Performance Testing of E-commerce Applications[J]”. IEEE Transactions on Software Engineering, 2004, 30(12): 1072- 1083.

[7] Newman D. Benchmarking Terminology for Firewall Performance[S]. RFC 2647, 1999

[8] H.B. Sun, M. Chen, Y.B. Cai, “Research on Benchmarking Method of Ipv4/Ipv6 Transition Gateway,” Journal of Computer Engineering, vol.32, no. 24, pp. 93-95 (in Chinese), 2006.
(孙红兵, 陈沫, 蔡一兵, “Ipv/4Ipv6 转换网关性能测试方法研究”, 计算机工程, 2006, 32(24): 93-95。)

[9] W. Liu, “Research on the Transformation Between WSP and HTTP Based on the Message Type [Master dissertation],”National University Deefnse Teehnology, Changsha, 2004.

[10] “RFC 2616: Hypertext Trannsfer Protocol--HTTP/1.1[EB/OL].” http://www.w3.org/Protocols.

[11] NIST Special Publication 800-115. Technical Guide to Information Security Testing and Assessment.

[12] BSI. Study A Penetration Testing Model. October, 2003.

[13] “OWASP Testing Guide v3 Table of Contents.” http://www.owasp.org/index.php/OWASP_Testing_Guide_v3_Table_of_Contents.

[14] OSSTMM. “The Open Source Security Testing Methodology Manual”.

[15] “China national vulnerability database of information security[DB/OL].”http://www.cnnvd.org.cn/vulnerability/statistics,2015-11-10.

[16] Okhravi H, Hobson T, Bigelow D, et al. “Finding Focus in the Blur of Moving Target Techniques[J],” IEEE Security & Privacy, 2013 (1): 1.

[17] Ron D, Shamir A. Quantitative analysis of the full bitcoin transaction graph[M]//Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2013: 6-24.

[18] Q.X. Liu, C.B. Zhang, Y.Q. Zhang “Research on key technology of vulnerability threat classification,” Journal on Communications, no. s1, pp. 79-87 (in Chinese), 2012.
(刘奇旭, 张翀斌, 张玉清, “安全漏洞等级划分关键技术研究”, 通信学报, 2012, s1: 79-87。)

[19] S. Pu, “Research on Web Security Penetration Testing[Master dissertation],”Xidian University, Xi’an, 2010.

[20] China internet security report (2016). http://zt.360.cn/1101061855.php?dtid=1101062370&did=1101654296, 2016-03-01.
(中国互联网安全报告（2016）. http:// http://zt.360.cn/1101061855.php?dtid=1101062370&did=1101654296, 2016-03-01。)

[21] Z. Zhang, “SQL Injection Attack Techniques and Countermeasures Analysis [Master dissertation],”Shanghai Jiaotong University, Shanghai, 2007.


**张铮** 于 2006 年在解放军信息工程大学计算机科学与技术专业获得博士学位。现任数学工程与先进计算国家重点实验室副教授。研究领域为网络安全、先进计算。研究兴趣包括: 主动防御技术、高性能计算。Email: ponyzhang@126.com

**马博林** 于 2015 年在哈尔滨工业大学信息安全专业获得学士学位。现在解放军信息工程大学计算机科学与技术专业攻读硕士学位。研究领域为网络安全。研究兴趣包括: 主动防御技术。Email: msd_mbl@qq.com

**邬江兴** 现任国家数字交换系统工程技术研究中心主任, 教授, 博导。研究领域为信息通信网络、网络安全。Email: 17034203@qq.com