服务安全与监控

用户帐号安全

设置帐号有效期

chage命令的语法格式

chage -l 账户名称 //查看账户信息

chage -E 时间 账户名称 //修改账户有效期

chage -E 2015-05-15 zhangsan

帐号的锁定/解锁

强制定期修改密码

伪装登陆提示

文件系统安全

程序和服务控制

sudo -l

/etc/sudo.conf

锁定/解锁保护文件

chatter +/- a/i file

lsatter file

审计：基于事先配置的规则生成日志，记录可能发生在系统上的事件，（正常或非正常行为的事件），审计不会为系统提供额外的安全保护，但他会发现并记录违反安全策略的人及其对应的行为

审计能够记录的日志内容

a）日期与时间以及事件的结果

b）触发事件的用户

c）所有认证机制的使用都可以被记录，如ssh等

d）对关键数据文件的修改行为等都会被记录

软件包名：audit

配置文件：/etc/audit/auditd.conf

日志文件路径/var/log/audit/audit.log

auditctl -s //查询状态

auditctl -l //查看规则

auditctl -D //删除所有规则

语法格式：auditctl -w path -p permission -k key\_name

path为需要审计的文件或目录

权限可以使rwxa（文件或目录的属性发生变化）

key\_name为可选项，方便识别那些规则生成特定的日志项

例：

audit -w /etc/passwd -p wa -k passwd\_change //设置规则所有对帕斯所谓的文件的写、属性修改等操作都会被记录审计日志

audit -w /etc/selinux -p wa -k selinux\_change //设置规则，监控/etc/selinux

nginx限制并发量

ddos攻击者会发送大量的并发链接，占用服务器的资源（包括链接数、带宽等）会导致正常用户处于等待或无法访问服务器的状态

nginx提供了一个ngx\_http\_limit\_req\_module模块可以有效降低ddos攻击的风险。

http {

limit\_req\_zone $binary\_remote\_addr zone=one:10m rate=1r/s;

server {

listen 80;

server\_name localhost;

limit\_req zone=one burst=5;

}

}

//把客户端的信息存入一个共享内存，内存空间为10M，1M可以储存8千个IP信息，容量可以调整

//每秒钟仅接受一个IP的一个请求，多余的放入漏斗（等待服务器响应的状态）；漏斗超过五个则会报错

拒绝非法的请求

网站使用的是http协议，该协议中定义了很多方法，可以让用户链接服务器，获得需要的资源，但实际应用中一般仅需要get和post。

|  |  |
| --- | --- |
| 请求方法 | 功能描述 |
| GET | 请求指定的页面信息，并返回实体主体 |
| HEAO | 类似于get的请求，只不过返回的响应中没有具体的内容，用于获取报头 |
| POST | 向指定资源提交数据进行处理请求（例如提交表单或者上传文件） |
| DELETE | 请求服务器删除指定的页面 |
| PUT | 向服务器特定位置上传资料 |

设置nginx拒绝非法的请求方法

http {

server {

listen 80;

if ($request\_method !~ ^(GRIT|POST)$) {

return 444;

}

}

}

防止buffer溢出

当客户端链接服务器时，服务器会启用各种缓存，用来存放链接的状态信息。

如果攻击者发送大量的链接请求，而服务器部队缓存做限制的花，内存数据就有可能溢出（空间不足）。

修改nginx配置文件，调整各种buffer参数，可以有效降低溢出风险

http {

client\_body\_buffer\_size 1K;

client\_header\_buffer\_size 1K;

client\_max\_body\_size 1K;

large\_client\_header\_buffers 2 1K;

......

}

数据库安全

初始化安全脚本

安装完mariadb或mysql后，默认root没有密码，并且提供了一个任何人都可以操作的test测试数据库，有一个名为mysql\_secure\_installation的脚本，该脚本可以帮助我们为root设置密码，并禁止root从远程其他主机登陆数据库，并删除测试性数据库test。

数据库改密码：

#mysqladmin -uroot -p123456 password 654321

//123456是原来的旧密码，654321是新修改的密码

进入数据库修改密码：

> set password for root@'localhost'=password redhat

这两种修改密码的方式虽然密码会加密，但是修改的密码会被明文记录

所以登陆后登陆完要记得清理历史，或者不要直接在命令行输入密码，而是使用交互式修改和登陆数据库，并选择5.6以后的版本

日志、行为审计（找到行为人），使用防火墙从tcp层设置acl（禁止外网接触数据库）。

数据库备份与还原

# mysqldump -uroot -predhat mydb（库名） table（表名） > table.sql

//备份数据库中的某个数据表

# mysqldump -uroot -predhat mydb > mydb.sql //备份某个数据库

# mysqldump -uroot -predhat --all-databases > all.sql //备份所有数据库

数据库自带的备份程序，备份速度慢，效率低，只适用于小型数据库

[roo@svr5 ~]# mysql -uroot -predhat mydb < table.sql //还原数据表

[roo@svr5 ~]# mysql -uroot -predhat mydb < mydb.sql //还原数据库

[roo@svr5 ~]# mysql -uroot -predhat < all.sql //还原所有数据库

数据库创建用户：

grant all on \*.\* to tom@'%' identified by '123';

远程登陆数据库：

mysql -utom -p123 -h 192.168.2.100 //数据为明文传输，会被抓包

可以通过ssh远程链接然后从本地登陆数据库，避免了数据的泄露

mysql支持ssl加密（确保网络中传输的数据是加密的）

使用diff和patch工具打补丁

程序总会有一些问题与漏洞，目前主流解决方法是为有问题的程序打补丁，升级新版本。

在Linux系统中diff命令可以为我们生成补丁文件，然后使用patch命令为有问题的程序代码打补丁

diff原则是：

告诉我们怎么修改第一个文件后得到第二个文件

如果第一个版本的脚本有漏洞，我们不需要将整个脚本都替换，金需要修改有问题的小部分代码即可。

像Linux这种大型程序，一个小小的漏洞我们不可能把整个内核都重新下载，重新替换一遍，而仅仅需要更新有问题的那一小部分代码即可

选项

-u 输出同一内容的头部信息（打补丁使用,计算机知道是哪个文件需要修改）

-r 第归对比目录中的所有资源（可以对比目录）

-a 所有文件是为文本（包括二进制程序）

-N 无文件视为空文件（空文件怎么变成第二个文件）

-N选项备注说明

A目录下没有txt文件，B目录下有txt文件

diff比较两个目录时，会默认提示txt仅在B目录有（无法对比差异，修复文件）

diff比较时使用N选项，则diff会拿下B的txt与A下的空文件对比，补丁信息会明确的说如何从空文件修改后变成txt文件，打补丁即可成功

iptables防火墙基本管理

防火墙是一道具有保护性的安全屏障，具有保护和隔离的作用

rhel7中默认使用的是firewalld作为防火墙。但firewalld底层还是调用包过滤防火墙iptables

包过滤匹配流程

规则链内的匹配程序

顺序比对，匹配即停止（LOG除外）

若误任何匹配，则按该链的默认策略处理

管理程序的位置在/sbin/iptables

指令组成

iptables [-t表名] 选项 [链名] [条件] [-j目标操作]

iptables基本用法

注意事项/整体规律

可以不指定表，默认为filter表

可以不指定链，默认为对应表的所有链

除非设置默认策略，否则必须指定匹配条件

选项/链名/目标操作用大写字母，其余都小写

基本的目标操作

|  |  |
| --- | --- |
| ACCEPT | 允许通过/放行 |
| DROP | 直接丢弃，不会给任何回应 |
| REJECT | 直接通过，必要时会给出提示 |
| LOG | 记录日志，然后传给下一条规则 |

iptables的四个表（区分大小写）

|  |  |
| --- | --- |
| nat表 | 地址转换表 |
| filter表 | 数据过滤表 |
| raw表 | 状态跟踪表 |
| mangle表 | 包标记表 |

iptables的五个链（区分大小写）

|  |  |
| --- | --- |
| INPUT链 | 入站规则 |
| OUTPUT链 | 出站规则 |
| FORWARD链 | 转发规则 |
| PREROUTING链 | 路由前规则 |
| POSTROUTING链 | 路由后规则 |

iptables常用选项

|  |  |  |
| --- | --- | --- |
| 类别 | 选项 | 描述 |
| 添加规则 | -A | 最佳一条放和火枪规则至链的末尾位置 |
| -I | 插入一台哦防火墙规则至链的开头 |
| 查看规则 | -L | 查看iptables所有规则 |
| -n | 以数字形式显示地址、端口等信息 |
| --line-numbers | 查看规则时，显示规则的行号 |
| 删除规则 | -D | 删除链内指定序号（或内容）的一条规则 |
| -F | 清空所有的规则 |
| 默认规则 | -P | 为指定的链设置默认规则 |

设置默认规则

所有链的初始默认规则均为ACCEPT

通过-P选项可重置默认规则

ACCEPT或者DROP

iptables过滤条件

yum -y install iptables-service

systemctl restart iptables.serviec

监控的目的

报告系统运行状况

每一部分必须同时监控

内容包括吞吐量、反应时间、使用率等

提前发现问题

进行服务器性能调整前，知道调整什么

找出系统瓶颈在什么地方

监控资源的类别

公开数据

Web ftp ssh 数据库等应用服务

Tcp或UDP端口

私有数据

CPU、内存、磁盘、网卡流量等使用信息

用户、进程等运行信息

系统监控命令

ps ifconfig uptime netstat|ss free ping swapon -s traceroute df -h iostat

自动化监控系统

Cacti；基于SNMP协议的监控软件，强大的绘图能力

Nagios：基于Agent监控，强大的状态检查与报警机制；插件及多，自己写监控见本潜入到Nagios非常方便

Zabbix：基于多种监控机制，支持分布式监控

Zabbix是一个高度集成的监控解决方案；可以实现企业级的开源分布式监控；通过C/S模式采集监控数据；通过B/S模式实现Web管理

监控服务器：监控服务器可以通过SNMP或Agent采集数据；数据可以写入Mysql、Oracle等数据库中；服务器使用LNMP实现web前端的管理

被检控主机：被监控主机需要安装Agent；常见的网络设备一般支持SNMP