Openresty中文编程网

version 0.1

Last generated: February 16, 2017

CandyLab

© 2016 OpenResty中文编程网 http://www.openresty.com.cn

Table of Contents

X-WAF介绍

Summary: 关于中小型企业防火墙X-WAF

X-WAF

X-WAF是一款适用中、小企业的云WAF系统，让中、小企业也可以非常方便地拥有自己的免费云WAF。

文档地址：https://waf.xsec.io/ (https://waf.xsec.io/)

主要特性

- 支持对常见WEB攻击的防御，如sql注入、xss、路径穿越，阻断扫描器的扫描等
- 对持对CC攻击的防御
- waf为反向模式，后端保护的服务器可直接用内网IP，不需暴露在公网中
- 支持IP、URL、Referer、User-Agent、Get、Post、Cookies参数型的防御策略
- 安装、部署与维护非常简单
- 支持在线管理waf规则
- 支持在线管理后端服务器
- 多台waf的配置可自动同步
- 跨平台，支持在linux、unix、mac和windows操作系统中部署

架构简介

x-waf由waf自身与Waf管理后台组成：

- waf (https://github.com/xsec-lab/x-waf)：基于openresty + lua开发。
- waf管理后台 (https://github.com/xsec-lab/x-waf-admin)：采用golang + xorm + macrom开发的，支持二进制的形式部署。

waf和waf-admin必须同时部署在每一台云WAF服务器中。

下载安装

waf安装

centos平台

从openresty (http://openresty.org/en/download.html)官方下载最新版本的源码包。

编译安装openresty：

```
yum -y install pcre pcre-devel
wget https://openresty.org/download/openresty-1.9.15.1.tar.gz
tar -zxvf openresty-1.9.15.1.tar.gz
cd openresty-1.9.15.1
./configure
gmake && gmake install

/usr/local/openresty/nginx/sbin/nginx  -t
nginx: the configuration file /usr/local/openresty/nginx/conf/nginx.conf syntax is ok
nginx: configuration file /usr/local/openresty/nginx/conf/nginx.conf test is successful
/usr/local/openresty/nginx/sbin/nginx
```

ubuntu平台安装

编译安装openresty：

```
apt-get install libreadline-dev libncurses5-dev libpcre3-dev libssl-dev perl make build-e
ssential
sudo ln -s /sbin/ldconfig /usr/bin/ldconfig
wget https://openresty.org/download/openresty-1.9.15.1.tar.gz
tar -zxvf openresty-1.9.15.1.tar.gz
cd openresty-1.9.15.1
make && sudo make install
```

安装waf管理后台x-waf-admin

二进制安装

直接从github中下载对应操作系统的二进制版本，https://github.com/xsec-lab/x-waf-admin/releases

源码安装

- 首先需要搭建好go语言开发环境，可以参考Go Web 编程 (https://github.com/astaxie/build-web-application-with-golang/blob/master/zh/01.1.md)
- 安装依赖包

```
go get gopkg.in/macaron.v1
go get gopkg.in/ini.v1
go get github.com/go-sql-driver/mysql
go get github.com/go-xorm/xorm
go get github.com/xsec-lab/x-waf-admin
```

- 从github中下载最新的版本
- 执行go build server.go编译出二进制版本，然后将server、conf、publib和templates目录一起打包上传到服务器中即可运行。

致谢

1. 感谢春哥开源的openresty (https://openresty.org)
2. 感谢unixhot开源的waf (https://github.com/unixhot/waf)
3. 感谢无闻开源的macron (https://go-macaron.com/)和peach (https://peachdocs.org/)
4. 感谢lunny开源的xorm (https://github.com/go-xorm/xorm)

X-WAF下载安装

Summary: 关于中小型企业防火墙X-WAF

下载安装

waf安装

centos平台

从openresty (http://openresty.org/en/download.html)官方下载最新版本的源码包。

编译安装openresty：

```
yum -y install pcre pcre-devel
wget https://openresty.org/download/openresty-1.9.15.1.tar.gz
tar -zxvf openresty-1.9.15.1.tar.gz
cd openresty-1.9.15.1
./configure
gmake && gmake install

/usr/local/openresty/nginx/sbin/nginx  -t
nginx: the configuration file /usr/local/openresty/nginx/conf/nginx.conf syntax is ok
nginx: configuration file /usr/local/openresty/nginx/conf/nginx.conf test is successful
/usr/local/openresty/nginx/sbin/nginx
```

ubuntu平台安装

编译安装openresty：

```
apt-get install libreadline-dev libncurses5-dev libpcre3-dev libssl-dev perl make build-essential
sudo ln -s /sbin/ldconfig /usr/bin/ldconfig
wget https://openresty.org/download/openresty-1.9.15.1.tar.gz
tar -zxvf openresty-1.9.15.1.tar.gz
cd openresty-1.9.15.1
make && sudo make install
```

安装waf管理后台x-waf-admin

二进制安装

直接从github中下载对应操作系统的二进制版本，https://github.com/xsec-lab/x-waf-admin/releases

源码安装

1. 首先需要搭建好go语言开发环境，可以参考Go Web 编程 (https://github.com/astaxie/build-web-application-with-golang/blob/master/zh/01.1.md)
2. 安装依赖包

```
go get gopkg.in/macaron.v1
go get gopkg.in/ini.v1
go get github.com/go-sql-driver/mysql
go get github.com/go-xorm/xorm
go get github.com/xsec-lab/x-waf-admin
```

1. 从github中下载最新的版本
2. 执行go build server.go编译出二进制版本，然后将server、conf、publib和templates目录一起打包上传到服务器中即可运行。

X-WAF部署配置

Summary: 关于中小型企业防火墙X-WAF

部署配置

waf部署与配置

openresty的配置

将x-waf的代码目录放置到openresty的 /usr/local/openresty/nginx/conf 目录下，然后在openresty的conf的目录下新建vhosts目录

```
cd /usr/local/openresty/nginx/conf/
git clone https://github.com/xsec-lab/x-waf
mkdir -p /usr/local/openresty/nginx/conf/vhosts
```

以下为openresty的配置范例：

```
user  nginx;
worker_processes auto;
worker_cpu_affinity auto;

#error_log  logs/error.log;
#error_log  logs/error.log  debug;
#error_log  logs/error.log  info;

#pid        logs/nginx.pid;

events {
    worker_connections  409600;
}

http {
    include       mime.types;
    lua_package_path "/usr/local/openresty/nginx/conf/x-waf/?.lua;/usr/local/lib/lua/?.lua;;";
    lua_shared_dict limit 100m;
    lua_shared_dict badGuys 100m;
    default_type  application/octet-stream;

        #开启lua代码缓存功能
        lua_code_cache on;

        init_by_lua_file /usr/local/openresty/nginx/conf/x-waf/init.lua;
    access_by_lua_file /usr/local/openresty/nginx/conf/x-waf/access.lua;

        #log_format shield_access    '$remote_addr - $http_host - "$request" - "$http_cookie"';
        #access_log pipe:/usr/local/shield/redisclient shield_access;

    #ssl on;
    #ssl_certificate certs/cert_chain.crt;
    #ssl_certificate_key certs/server.key;
    ssl_session_timeout     5m;
    ssl_protocols SSLv2 SSLv3 TLSv1;
    ssl_ciphers ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP;
    ssl_prefer_server_ciphers on;


    sendfile        on;
    #tcp_nopush     on;

    #keepalive_timeout  0;
    keepalive_timeout  65;
```

```
    #gzip  on;
    include vhosts/*.conf;

    server {
        listen       80;
        server_name  localhost;

        #charset koi8-r;

        #access_log  logs/host.access.log  main;

        location / {
            root   html;
            index  index.html index.htm;
                }
        }
}
```

waf的配置

waf的配置文件位于 /usr/local/openresty/nginx/conf/waf/config.lua 中，详细的配置项如下：

```
-- WAF config file, enable = "on", disable = "off"

local _M = {
    -- waf status
    config_waf_enable = "on",
    -- log dir
    config_log_dir = "/tmp/waf_logs",
    -- rule setting
    config_rule_dir = "/usr/local/openresty/nginx/conf/x-waf/rules",
    -- enable/disable white url
    config_white_url_check = "on",
    -- enable/disable white ip
    config_white_ip_check = "on",
    -- enable/disable block ip
    config_black_ip_check = "on",
    -- enable/disable url filtering
    config_url_check = "on",
    -- enalbe/disable url args filtering
    config_url_args_check = "on",
    -- enable/disable user agent filtering
    config_user_agent_check = "on",
    -- enable/disable cookie deny filtering
    config_cookie_check = "on",
    -- enable/disable cc filtering
    config_cc_check = "on",
    -- cc rate the xxx of xxx seconds
    config_cc_rate = "10/60",
    -- enable/disable post filtering
    config_post_check = "on",
    -- config waf output redirect/html/jinghuashuiyue
    config_waf_model = "html",
    -- if config_waf_output ,setting url
    config_waf_redirect_url = "http://xsec.io",
    config_expire_time = 600,
    config_output_html=[[
    <html>
    <head>
    <meta charset="UTF-8">
    <title>xsec waf</title>
    <style type="text/css">
        body {
      font-family: "Helvetica Neue", Helvetica, Arial;
      font-size: 14px;
      line-height: 20px;
      font-weight: 400;
      color: #3b3b3b;
      -webkit-font-smoothing: antialiased;
```

```
  font-smoothing: antialiased;
  background: #f6f6f6;
}
.wrapper {
  margin: 0 auto;
  padding: 40px;
  max-width: 980px;
}
.table {
  margin: 0 0 40px 0;
  box-shadow: 0 1px 3px rgba(0, 0, 0, 0.2);
  display: table;
}
@media screen and (max-width: 580px) {
  .table {
    display: block;
  }
}
.row {
  display: table-row;
  background: #f6f6f6;
}
.row:nth-of-type(odd) {
  background: #e9e9e9;
}
.row.header {
  font-weight: 900;
  color: #ffffff;
  background: #ea6153;
}
.row.green {
  background: #27ae60;
}
.row.yellow {
  background: #FF8C00;
}
@media screen and (max-width: 580px) {
  .row {
    padding: 8px 0;
    display: block;
  }
}
.cell {
  padding: 6px 12px;
  display: table-cell;
}
@media screen and (max-width: 580px) {
```

```
    .cell {
      padding: 2px 12px;
      display: block;
    }
  }
  </style>
  </head>
    <body>
      <div class="wrapper">
    <div class="table">
      <div class="row header yellow">
        <div class="cell">
          您的IP为 %s
        </div>
        <div class="cell">
          欢迎在遵守白帽子道德准则的情况下进行安全测试。
        </div>
        <div class="cell">
          联系方式：x@xsec.io
        </div>
      </div>
    </div>
  </div>
    </body>
  </html>
  ]],
}

return _M
```

waf测试

使用root权限执行以下命令测试配置文件的正确性，如果测试结果返回ok则表示配置是正确的。

```
$ sudo /usr/local/openresty/nginx/sbin/nginx -t
[sudo] hartnett 的密码：
nginx: the configuration file /usr/local/openresty/nginx/conf/nginx.conf syntax is ok
nginx: configuration file /usr/local/openresty/nginx/conf/nginx.conf test is successful
```

如果配置文件正常就可启动waf：

```
$ sudo /usr/local/openresty/nginx/sbin/nginx
```

WAF防御效果测试

在服务器中提交 curl http://127.0.0.1/\?id\=1%20union%20select%201,2,3

如果返回的内容中包含 欢迎在遵守白帽子道德准则的情况下进行安全测试 等字样就表示waf已经在正常运行了。

waf-admin配置

waf-admin需要mysql的支持，事先需要准备一个mysql数据库的账户，以下为app.ini的配置范例：

```
RUN_MODE = dev
;RUN_MODE = prod

[server]
HTTP_PORT = 5000
API_KEY = xsec.io||secdevops.cn
NGINX_BIN = /usr/local/openresty/nginx/sbin/nginx
NGINX_VHOSTS = /usr/local/openresty/nginx/conf/vhosts/
API_SERVERS = 127.0.0.1, 8.8.8.8

[database]
HOST = mysqlhost:3306
USER = waf-admin
PASSWD = passw0rd
NAME = waf

[waf]
RULE_PATH = /usr/local/openresty/nginx/conf/waf/rules/
```

- RUN_MODE为运行模式，dev为开发模式，prod为线上模式，正式上线前请将运行模式改为prod
- HTTP_PORT为waf-admin监听的端口
- API_KEY为多台waf-admin同步配置信息时用的加密key，建议设置一个复杂的字符串
- NGINX_BIN为nginx的可执行文件的物理路径
- NGINX_VHOSTS为nginx的虚拟主机目录的物理路径

- API_SERVERS表示有几台waf服务器，多台waf服务器之间的ip用英文逗号分割
- database节为mysql的配置信息，分别用数据库地址，用户名、密码以及库名
- waf节中的RULE_PATH表示waf的规则存放的位置

配置完成后在当前目录执行./server测试程序是否可以正常启动，第一次启动的时候，如果数据库能正常连接，则会自动初始化默认的waf规则，以及新建一个用户名为admin，密码为：x@xsec.io的用户。

waf-admin需要操作nginx的master进程，所以需要以root权限启动，可以使用supversisor、nohup、systemd等将waf-admin跑在后台。

X-WAF介绍

Summary: 关于中小型企业防火墙X-WAF

name: X-WAF简介 —

X-WAF

X-WAF是一款适用中、小企业的云WAF系统，让中、小企业也可以非常方便地拥有自己的免费云WAF。

主要特性

- 支持对常见WEB攻击的防御，如sql注入、xss、路径穿越，阻断扫描器的扫描等
- 对持对CC攻击的防御
- waf为反向模式，后端保护的服务器可直接用内网IP，不需暴露在公网中
- 支持IP、URL、Referer、User-Agent、Get、Post、Cookies参数型的防御策略
- 安装、部署与维护非常简单
- 支持在线管理waf规则
- 支持在线管理后端服务器
- 多台waf的配置可自动同步
- 跨平台，支持在linux、unix、mac和windows操作系统中部署

架构简介

x-waf由waf自身与Waf管理后台组成：

- waf (https://github.com/xsec-lab/x-waf)：基于openresty + lua开发。
- waf管理后台 (https://github.com/xsec-lab/x-waf-admin)：采用golang + xorm + macrom开发的，支持二进制的形式部署。

waf和waf-admin必须同时部署在每一台云WAF服务器中。

X-WAF高级用法

Summary: 关于中小型企业防火墙X-WAF

如何修改默认的虚拟主机模板

如果waf管理后台默认生成主要的模板无法满足您的业务需求，您可以到x-waf-admin 的 templates 目录下修改模板文件 proxy.tmpl ，默认的模板内容如下：

```
upstream proxy_ {
        server  max_fails=3  fail_timeout=20s;
        }

server  {
        listen       ;
        ssl          ;
        server_name  ;
        client_max_body_size 100m;
        charset utf-8;
        access_log      /var/log/nginx/-access.log;
        error_log       /var/log/nginx/-debug.log ;

        location ~* ^/ {
            proxy_pass_header Server;
            proxy_set_header Host $http_host;
            proxy_set_header Host api.miren.t.n.mi.com;
            proxy_redirect off;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Scheme $scheme;
            proxy_pass $scheme://proxy_;
            }

            error_page  404              /index.html;
            error_page   500 502 503 504  /index.html;
        }
```

后端反向代理使用了Go语言内置的模板引擎 html/template 来渲染，详细语法可参考官方文档 (https://godoc.org/html/template)与《Go WEB编程》模板处理章节 (https://github.com/astaxie/build-web-application-with-golang/blob/master/zh/07.4.md)

X-WAF开始使用

Summary: 关于中小型企业防火墙X-WAF

后端服务器管理

当多台waf做负载均衡时，只需登录其中一台进行管理即可，多台waf的所有的配置信息会自动同步到所有的服务器中。

管理地址为：http://ip:5000/login/

管理后台的默认的账户及口令分别为：admin，x@xsec.io，请管理员部署系统后第一时间修改密码，防止被攻击者使用默认口令登入胡乱改动waf的配置。

新增站点

在 Site Manager 选项中，可以新增一个后端服务器，需要填写以下内容：

- Site Name，表示要加入waf的网站的域名
- 80表示该网站监听的端口
- Backend，表示有多少个后台app server，可以写多个（换行分割），例如： bash 1.1.1.1:80 8.8.8.8:80
- SSL Status，表示是否启用ssl，参数为on或off（如果要启用的话，需要在nginx中配置有效的证书）
- Debug Level，表示日志级别，可选的参数有
  debug, info, notice, warn, error, crit, alert, emerg

站点配置同步

新增站点后，需要在后台同步站点信息后方可生效，同步的方式有2种：

1. 全部同步
2. 针对某一新增的站点进行同步

waf规则管理

在 waf Rules 选项中，可以修改waf的规则，修改完后可以点击“同步全部策略”按钮，将最新的规则同步到所有的服务器并让openresty重新加载。

X-WAF常见问题

Summary: 关于中小型企业防火墙X-WAF

常见问题

openresty的语法正确，却启动失败

以下的错误原因为有些linux发布版本要求监听1024以下的端口的服务必须以root权限启动。

```
/usr/local/openresty/nginx/sbin/nginx -t
nginx: [warn] the "user" directive makes sense only if the master process runs with su
per-user privileges, ignored in /usr/local/openresty/nginx/conf/nginx.conf:1
nginx: the configuration file /usr/local/openresty/nginx/conf/nginx.conf syntax is ok
nginx: [emerg] bind() to 0.0.0.0:80 failed (13: Permission denied)
nginx: configuration file /usr/local/openresty/nginx/conf/nginx.conf test failed
```

解决方案：以root账户或用sudo命令执行。

在waf admin中同步全部策略报错

返回 {"Status":1,"Message":"reload nginx configure faild"} 错误的原因为waf admin是以普通用户权限启动的，在同步完策略后无法操作openresty的master进程完成策略加载的操作。

解决方案：以root账户或用sudo命令启动waf admin。

正常访问被误报为CC攻击

如果waf前端有高可用VIP，需要将用户真实地址传递给后端waf，否则waf获取到的是vip的地址，访问量大到触发cc攻击的阀值后就会将正常的请求误报为cc攻击。

解决方案：在前端VIP中将用户真实地址传递到后端服务器。

X-WAF开发计划

Summary: 关于中小型企业防火墙X-WAF

发起工单

如果您发现bug或有一些期望的新功能，可以通过发起工单 (https://github.com/xsec-lab/xsec-waf-docs/issues) 的形式告知作者。

开发计划

- 增加攻击统计报表
- 增加waf性能监控