高并发开源软件WAF在企业安全实战中的探索

盛洋 (糖果L5Q) | 新浪安全工程师，FreeBuf 年度作者

FIT · FREEBUF

百花齐放WAF时代来临

黑客有时扮演一个破坏者的身份，像幽灵一样渗透进入系统，有时又像用创意洞见世界的艺术家，用创造性的黑客技术打开系统的门，而防火墙平台的策略规则，给了一次黑客与防护者间的对话机会，对于黑客来说，系统的输入不是系统创建者提供的几个表单，而是整个HTTP协议可接受的数据范围内构建请求，当这请求摆脱系统视野时，伏击已经开时了。

黑客的渗透如撒旦的诅咒，攻击者与防御者之间请求与响应的会话，决定了系统是否被渗透，或是攻击被拦截，安全策略规则，如魔法师的咒语一样，识破黑客攻击的意图，驱赶破门而入的幽灵，让其只能在大门外的天空中盘旋。

WAF是一道无型的墙，刻在墙上的魔法就是驱魔的经语，一旦诵错，幽灵将穿墙而入。

开源时间代的杨帆启航

历史上重来我们没有像今天这样依赖开源软件，开源软件从原来的一支独秀，到花团锦簇，至今天的百花齐放，开源软件依靠低成本，社区共同维护等特点，迎来她的春天，四季轮回更替，拥抱开源就是拥抱软件的新纪元。

这是一条充满创造性的探险之路，未来之路，我们在这条路攀登技术的险峰，也在这条路上发现了无数的黄金宝藏，迈进开源世界的一个神秘花园，开源防火墙探索之路，安全运维人员的安全盾。

安防开发人员用热情写出一条条命令语句，集沙成塔，筑成一道墙、一条街、一个要塞。安全渗透专家写下的每一条安全策略规则，如刀叉剑戟一般，刺向来犯者的胸膛。这是在WEB世界的战争，要被黑客攻破，要么拒攻击于千里之外。

回到起点谈需求

实际工作中，业务部门软硬件部署迥异，安全威胁有时来至主业务群，有时来自流量较小的业务，大流量往往会给漏洞更多的曝光机会，得到重点的看护。而漏洞不会因为流量小，就逃过黑客的发现和利用，一些通用型漏洞，因某种原因潜伏在流量比较小，威胁不一定小的线上环境。如果可根据不同的业务部署环境，采用一种比较灵活的安全防护方案， 同时兼具有成本低，高性能的特点，很好的和业务融合，起到比较理想的防护效果，才是喜闻乐见的。

WAF的要求指标是什么

性能：对于起到分析与拦截功能的防火墙来说，性能的好坏直接影响后端服务对客户请求的响应，性能如果成为障碍，是无法应用落地的。

https://github.com/openresty/sregex

成本： 构建一个大型的清洗中心，从硬件成本和软件的授权成本来说都是价格不低的， 如果可以采用高性能的开源解决方案可以减低软件花费成本。

使用体验：安全产品是要交付给安全运维人员使用的，在日常生产防护中，能贴近说安全运维人员的思维和操作习惯的产品才能更受到欢迎。

WAF的要求指标是什么

部署灵活：实际业务中系统环境往往比较复杂，如果安防产品可以灵活的适用于各种环境，才可能正常的实施，如果存在软件兼容性问题也不可以。

有效性：能有效的拦截威胁才是有效的，开源软件是免费的，但核心安全策略需要根据外部环境与时俱进，不断的完善和丰富，如果没有专门的厂商提供服务，就需要社区和安全运维人员进行策略的维护。

开源WAF的起源

今天更多别人们提到的开源的WAF系统，更多的可以已经变成了，基于Nginx LUA的WEB防火墙系统，这种防火墙系统的特点是基于最常见的nginx服务，使用Lua语言及API来实现WAF功能，使用LUA不像使用C写模块的维护成本那么高，LUA小巧性能优越，降低了开发难度和维护成本，Nginx被广泛的应用HTTP7层相关的应用开发，很多人都不陌生。Nginx+lua的开源WAF兴起之后，开源WAF的发展随着Openresty的发展进入了新轨道，国内的开发人员开始基于Openresty写了很多的中间件服务，社区越来越活跃，下面介绍一些社区比较活跃的开源软件。

开源WAF产品

loveshell： LoveShell的WAF是比较多人关注的一个开源WAF版本，是开源nginx lua防火墙的一个启蒙产品，让很多人员走上了Nginx Lua开发防火墙之路。

VeryNginx：VeryNginx的引入更多的设计概念应用到设计实现中，开启了WAF可视化的一个开端，可通过监听的可视化监听看到服务的状态，但不支持集群的集中管理。

Resty-waf：是Resty系列中的一个开源WAF。

开源WAF产品

Orange：Orange其实是一个网关产品，借鉴了kong的设计，Kong用的路由系统是基于Lapis框架的，而Orange使用的是自己的框架LOR，Orang后续还有很的功能加入到系统中。WAF功能，其实是Orange网关产品的一个插件。

Vanilla：Vanilla和LOR一样是openresty lua框架，vanilla集成了Loveshell基于正则的WAF系统，作为一个插件存在。

OpenWAF:是一款不错的WAF产品，可以在阿里云上有开源和商业版二个版本。

XWAF： XWAF是小米安全开发一款灵巧的WAF产品。

开源WAF产品优点

开源WAF的特点的是免费，可定制化能力强，可以依赖社区开发人员，只要有人提供新代代码，就可以体验新功能，如果想加功能，直接改源码就好。

免费：很多开发WAF都可以在github上找到源码，不收取费用。

社区后盾：有社区帮助测试，有问题可以直接提Issue。

源码开放：如果发现有什么功没有，直接动改改，然后提交的主分支。

开源WAF产品缺点

核心策略的更新速度没有商业产品及时，一些软件使用的用户体验不是很好，面临项目被开发人员一起的风险。

安全策略更新：一些开源产品的策略都是演示性， 很多产品是没有核心的策略库的，通用漏洞的覆盖率也不是很高。

用户体验：某些开源产品只是实现了核心的防火逻辑，但是针对策略维护，UI交互实现的不是体验很好。

项目被放弃：开源产品面临项目被放弃的局面， 原本的开放人员放弃了项目的更新，没有后续的支持力量。

框架服务

原本这些框架是和WAF没有直接关系，但这些框架，构成WAF软件的基石，并且Openresty的兴起，让开发人员越来越关注，基于openresty的web中间构建有，社区也开始有较成熟的框架软件出现，有了这些框架，有时才有了后续的WAF产品。

Lapis： Lapis严格的说是用Moonscript实现的Web框架，被预编译成Lua，Kong网关就是使用的lapis的路由系统。

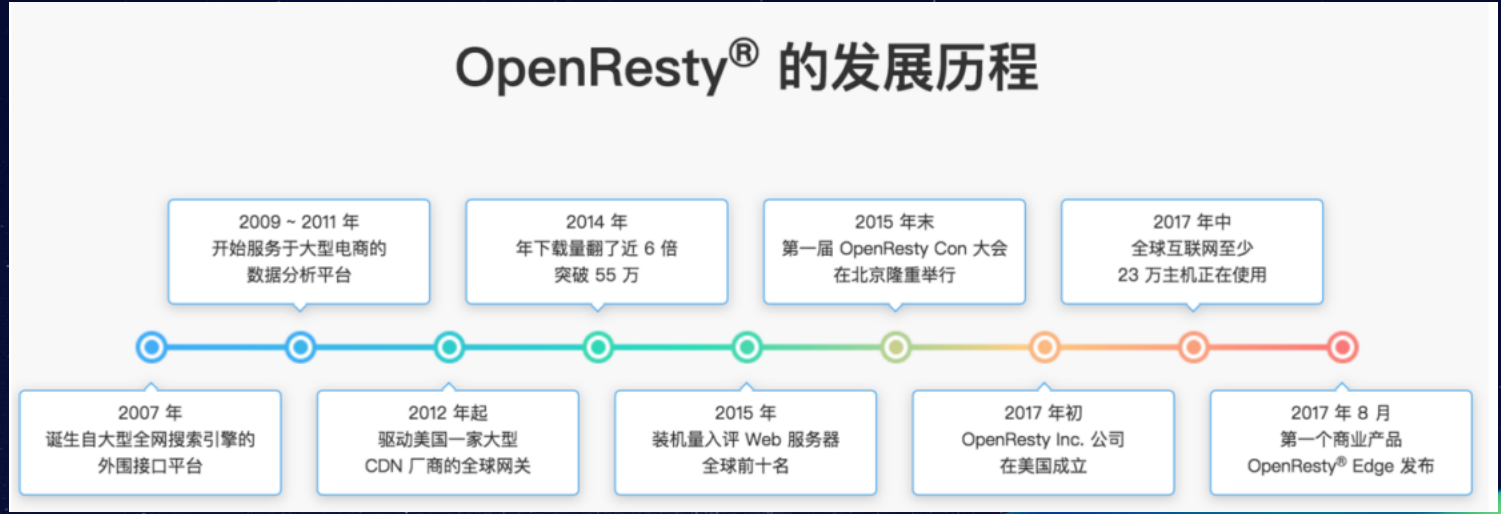
Gin：是一个精巧的RESTAPI的框架，之后因产的Vanilla和Lor都复用了一部分Gin的代码。

框架服务

Vanilla：是微博工程师写的一个框架，基于MVC模式的框架。

LOR：是头条的工程师写的轻量的WEB框架，Orange使用的基础框架就是LOR。

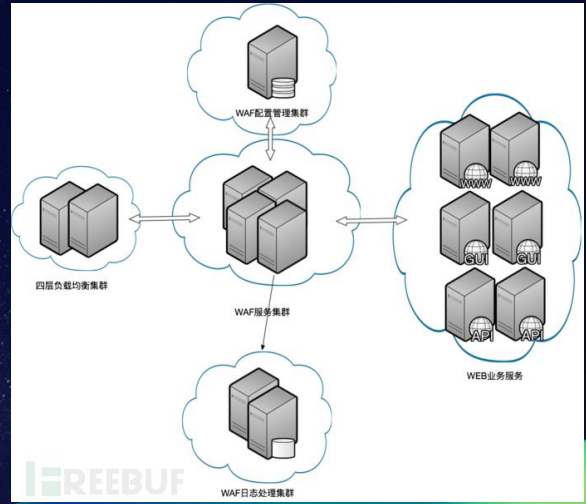
基出平台的发展历程


OpenResty® 的发展历程
2009 ~ 2011 年
开始服务于大型电商的
数据分析平台
2014 年
年下载量翻了近 6 倍
突破 55 万
2015 年末
第一届 OpenResty Con 大会
在北京隆重举行
2017 年中
全球互联网至少
23 万主机正在使用
2007 年
诞生自大型全网搜索引擎的
外围接口平台
2012 年起
驱动美国一家大型
CDN 厂商的全球网关
2015 年
装机量入评 Web 服务器
全球前十名
2017 年初
OpenResty Inc. 公司
在美国成立
2017 年 8 月
第一个商业产品
OpenResty® Edge 发布

测试实践

面对企业内容IT硬件与软件部署的同构化与异构化，基于不同情况采取同一种部署WAF的方式相对比较困难，而采用相对量体裁衣的方式，可能更适用于实际的业务情况， 一个公司有着不同的部门，采用不同年代的机器，安全了各种各样的软件系统，灵活、高效、低成本比较好操作。

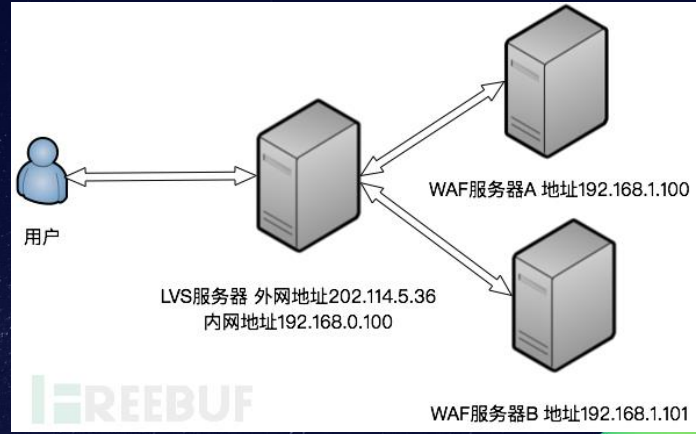
开放人员对自己使用过的开源技术，有一定的亲近感，像Nginx、Openresty这种软件服务的接受度就要高，实际企业当中本身正常的服务，可能就是由多层的Openresty服务搭建的，这样以来，再加上一层Openresty、Nginx服务相对可以很好的兼容过去的服务。



测试实践-串行

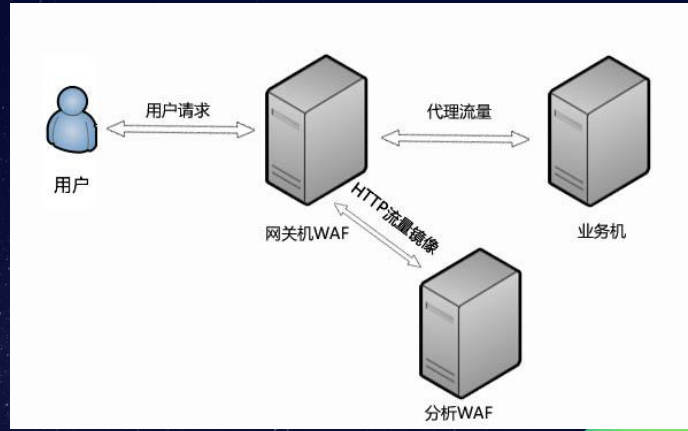
串行链接会对业务方的服务有响应消耗，直接接入到原有业务架构中，因老旧业务使用的软件年代久远，可能会出现不兼容的情况。

好处就是部署简单，但为了不影响后端业务，只有采用集群的方式，才能保证带宽可用，尽量的减低对后端的损耗。



测试实践-并行

这种往往是基于在网络层复制镜像，或是分光技术，7层的的数据在这过程中的精度和字段可能会出现减少，比如HTTP的一些header信息cookie信息，post数据，这种在7层可以被WEB服务看到的仔细的数据的，在基于流量产生的日志中被挥发掉，即使可以通过特定方式落地到日志文中进行分析，原因的web服务所提供的API通用功能也不能被使用，基于4层重写一套方案，基于流量复制和后期整日志整型的方案有类似的问题。可以根据日志进行更复杂的算法分析。

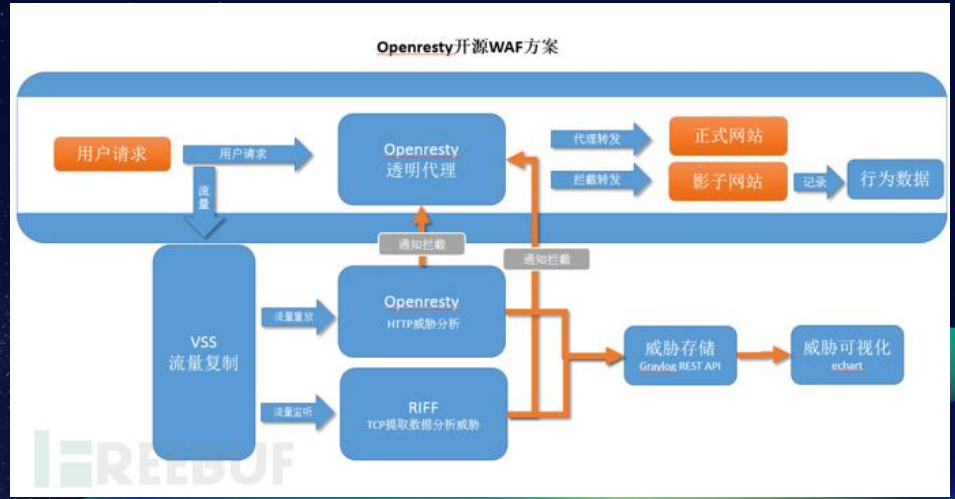


测试实践-并行

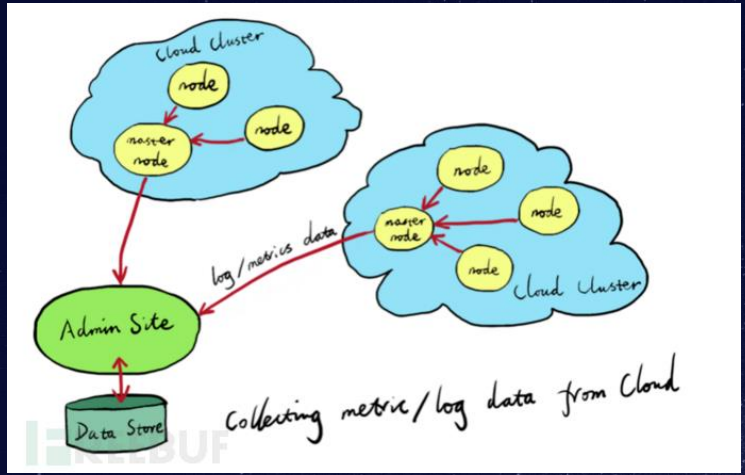
HTTP-MIRROR的出现将改变这种情况，HTTP服务会在7层对数据进行复制，虽然也会出现延时损坏，便提供了更丰富的可能性再后期，不会担心在镜像服务器的错误操作影响后端。

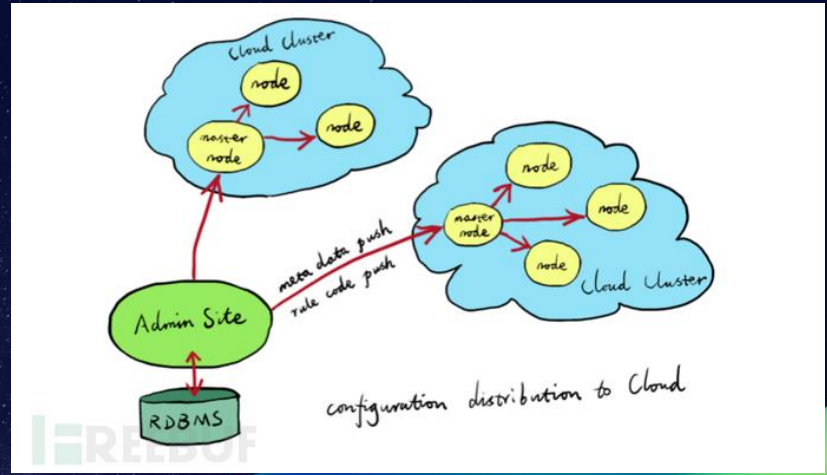
测试实践-串并同行

串并同联的情况，只是将拦与分析这两个功能分开，串接的设备不负责分析，直接将流量代理到后端业务，同时将业务的请求量，镜像给分析设备，经由分析设备进行分析判断，记录威胁，如果确认是攻击，再与拦截模块进行通信进行拦截，最大限度的减少对业务响应损耗。



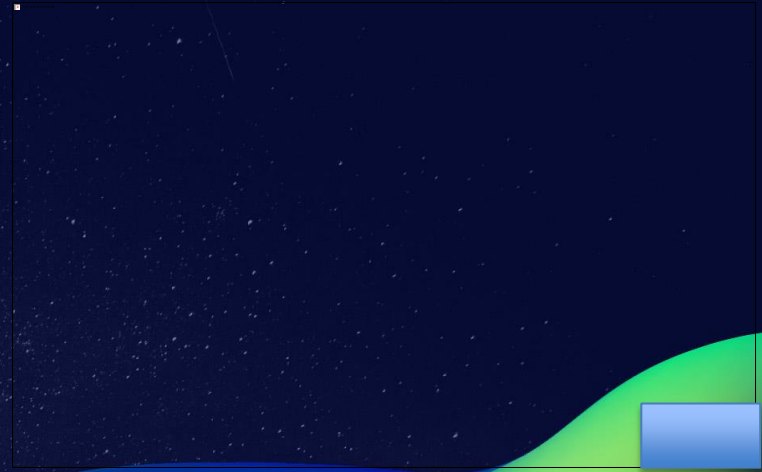
测试实践-串并同行





测试实践-靶机设定

不同的业务的运行环境是不一样的，对所有服务运行环境，采用单一的测策略会消耗大量的时间，PHP的运行环境，不一定适Python环境的测试，靶机测试可以用来测试某特定环境下通用漏洞，但对于防止绕过策略，需要运维人员不断的更新策略，保证系统不被白绕过，比如我们最常见的dvwa就是一个很好用的PHP靶机。

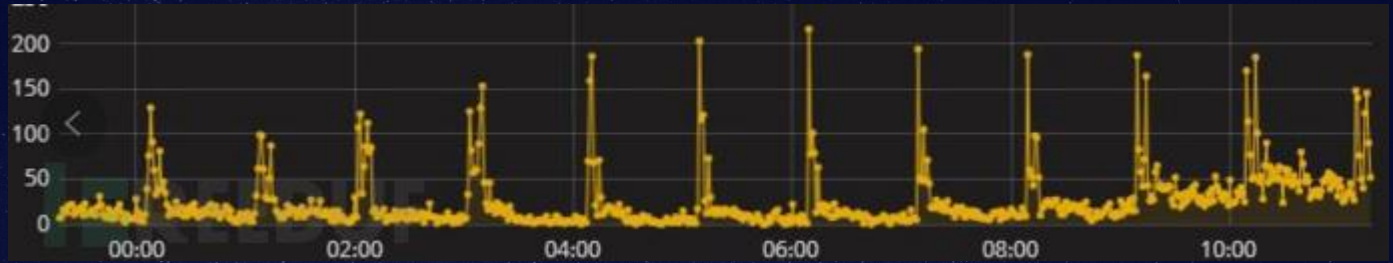


WAF的防护指标-误报率

真实流量是误报率的放大器，漏报率直接检验WAF是否可以有效拦截，误报率与漏报率是除了WAF性能之外，很需要关注的指数。

右图是一个威胁数据可视化的图片，我们可以图上观察到，威胁数据呈现明显的周期性。我们通过观察周期高峰的威胁的源IP和攻击的目地IP分析出，高峰时期出现威胁情况的业务是那些。通过他细的分析，
发现是一个业务启动的周期性的监控业务， 这个业务被报出威胁原因是因为在业务的URL中直接出现的SQL文，防护设备认为这是SQL注入，所有报出大量的威胁报警，因为监控业务数据量本身是周期性的所以报警也发现出这种周期性。

WAF的防护指标-误报率



WAF的防护指标-误报率

基于这种误报，我们除了加白名单，有没有更细粒度策略近制方法。 我们同时可以根据URL中SQL与具体的IP和网段来设置加白，如果用Openresty 的edge小语言描述这个问题，相对就比较好理解。

```
uri contains “SELECT”，
client-addr !~~ 10.210.1.1
=>
waf-mark-evil(message: “sql inject”, level: “super”);
```

WAF的防护指标-误报率

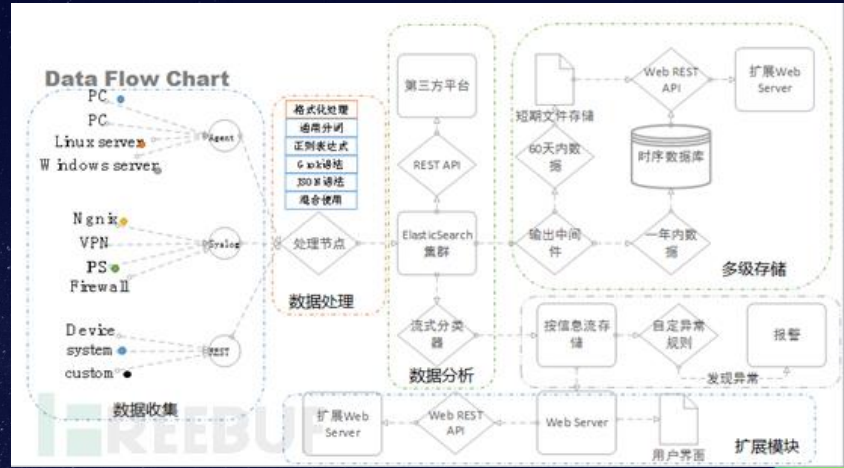
对于基于靶机的定向攻击，在明确已经攻击的种类和数量前提下的攻击，更适合于对防火墙的漏报率测试，我们汇总一些通用的漏洞，或者根据某些特定的应用环境，不熟适用于这一环境的靶机，能相对有效有测试出防火墙的忙点，比如符合某些共同特征的挂吗漏洞、或是不常见的基于cookie的攻击防御。

在进行拦截的情况，误报率会误伤正常的业务， 而基于构建威胁靶机的流量，并不适用于误报的测试， 而将WAF部署于正常的业务流量中，可以将误报的问题放大，我们根据大数据收集与威胁数据可视化的方法，来观察周期性流量中的产生的误报率。

上传挂吗本身有很多的变种，基于样本的漏洞防御本身有很好的针对性，但是还是有会有相应不及时的情况出现，这时基于策略逻辑的，可以起到补充的作用。

大数据可视化

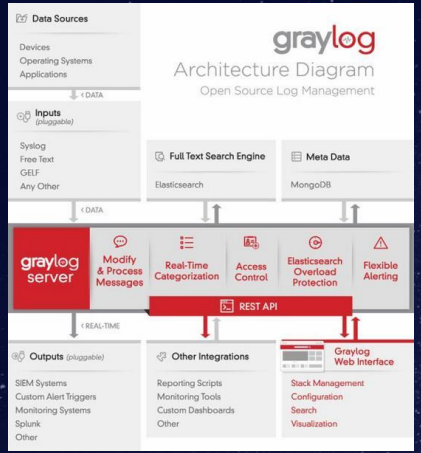
威胁分析：日志的收集与威胁数据可视化。正常的服务流量，会形成一个规律性的特征，如果可视化后，可以通过图像分析出不正常的图像特点，关于威胁可视化也同样的可以使用开源解决方案，比较优秀的日志汇聚工具就是Graylog， 出色的可视化工具Granfana。



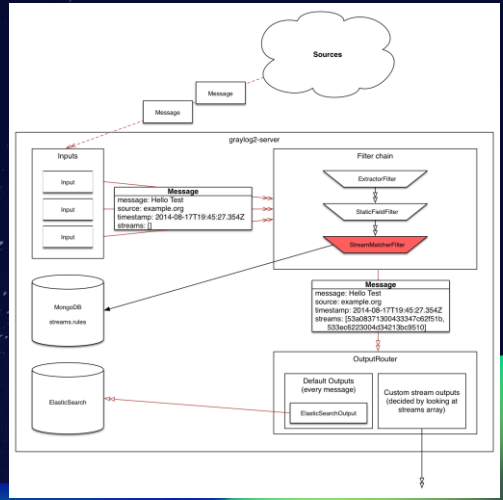
大数据可视化

威胁分析：日志的收集与威胁数据可视化。正常的服务流量，会形成一个规律性的特征，如果可视化后，可以通过图像分析出不正常的图像特点，关于威胁可视化也同样的可以使用开源解决方案，比较优秀的日志汇聚工具就是Graylog， 出色的可视化工具Granfana。

大数据可视化


Data Sources
Devices
Operating Systems
Applications
graylog
Architecture Diagram
Open Source Log Management
Inputs (pluggable)
Syslog
Free Text
GELF
Any Other
Full Text Search Engine
Elasticsearch
Meta Data
MongoDB
graylog server
Modify & Process Messages
Real-Time Categorization
Access Control
Elasticsearch Overload Protection
Flexible Alerting
REST API
Outputs (pluggable)
SIEM Systems
Custom Alert Triggers
Monitoring Systems
Splunk
Other
Other Integrations
Reporting Scripts
Monitoring Tools
Custom Dashboards
Other
Graylog Web Interface
Stack Management
Configuration
Search
Visualization


splunk>
应用
Search & Reporting
Splunk Add-on for Tenable
Tenable NNM (previously PVS) for Splunk
Website Monitoring
浏览 Splunk Enterprise
产品教程
添加数据
Splunk 应用
Splunk 文档
关闭
选择主页仪表板


Sources
Message
Message
graylog2-server
Inputs
Input
Input
Input
Message
message: Hello Test
source: example.org
timestamp: 2014-08-17T19:45:27.354Z
streams: []
Filter chain
ExtractorFilter
StaticFieldFilter
StreamMatcherFilter
MongoDB
streams.rules
Message
message: Hello Test
source: example.org
timestamp: 2014-08-17T19:45:27.354Z
streams: [53a08371300433347c62f51b, 533ec6223004d34213bc9510]
OutputRouter
Default Outputs (every message)
ElasticSearchOutput
Custom stream outputs (decided by looking at streams array)
ElasticSearch

小语言时代的规策略语言

Openresty Edge语言是Openresty作者创建一个小语言，系统会对规则进行编译，生成优化后的Lua语言，然后通过长链推送到边缘节点，一秒中内可以推到全球的节点中，边缘节点上会热加载这些Lua代码实时生效无需reload。

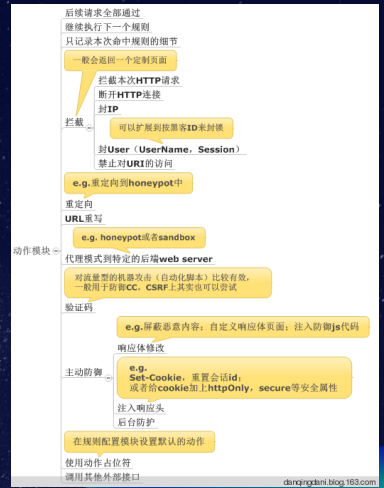
一般的WAF更新完规则后是需要reload，并且规则是通过配置文件下发的。并且服务端口可以直拉用edge切换upstream。

安全策略实践

我们将edge语言视为我们的策略的描述语言，来具体分析几析案例，针对案例攻击中的特点，Payload形式，攻击分类来针对性进行防御策略的编写与制定。

DSL小语言，一种基于领域业务的描述语言把问题描述的更灵活与细致，快速的响应灵活的运用，既有继承又有发展，小语言让策略规则更符合业务和人的思维模式。

DSL对过去LUA写的程序程序进行了更高一层的操作，过去用LUA需要写多个函数完成的工，用edge语言几句话就可以完成任务。



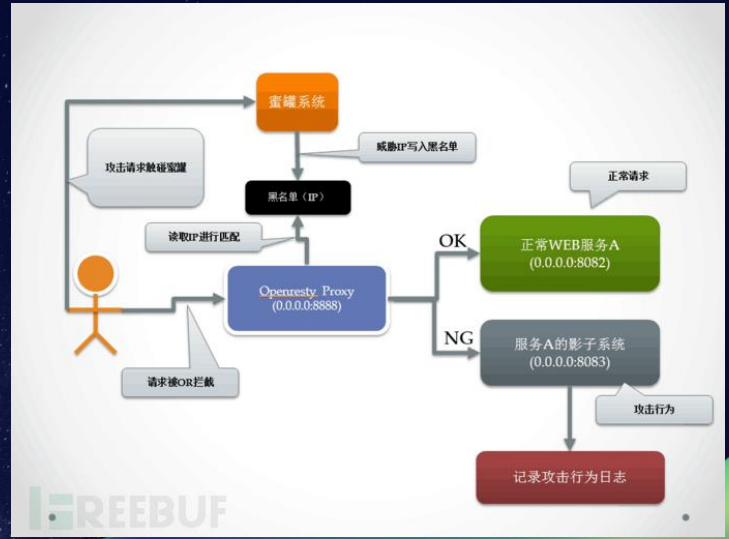
安全策略实践-镜花水月与影子系统

开源WAF系统XWAF中有一种工作模式叫作：镜花水月。这种模式我们也称为是影子系统模式，就是WAF系统确认请求是黑客的可疑攻击时，将请求引入到另外一个很像主系统的子影子系统，返回一些欺骗性数据，同时收集黑客入侵的数据痕迹和相应的资源，如果我们用edge来描述这个过程：

```
uri contains "SQL"=>
set-upstream('HoneyPot_1');
req-header("Content-Type") contains "multipart/form-data",
req-header("Content-Type") !contains rx{^multipart/form-
data[\s\S]+} =>
waf-mark-evil(message: "CVE-2017-5638 Struts", level:
"super"),
set-upstream('HoneyPot_2');
```

安全策略实践-镜花水月与影子系统

开源WAF系统XWAF中有一种工作模式叫作：镜花水月。这种模式我们也称为是影子系统模式，就是WAF系统确认请求是黑客的可疑攻击时，将请求引入到另外一个很像主系统的子影子系统，返回一些欺骗性数据，同时收集黑客入侵的数据痕迹和相应的资源，如果我们用edge来描述这个过程：



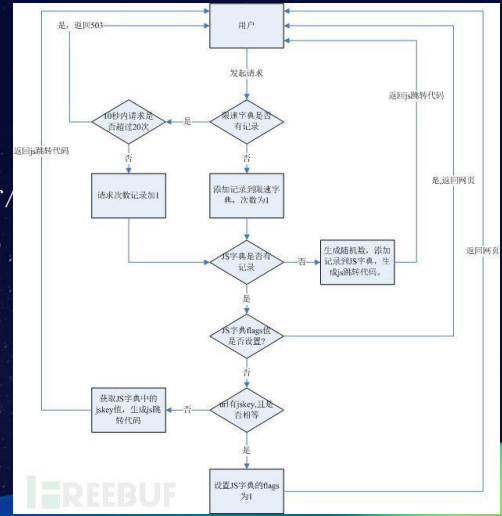
安全策略实践-CC防护限速

这个是限制请求速度，以及返回数据的速度，里面包括了 IP 和是否移动端的判断：

```
uri("/shop"), client-province('Guangdong'),
ua-is-mobile() =>
limit-req-rate(key: client-addr, target-rate: 5 [r/s], reject-rate: 10 [r/
   limit-resp-data-rate(441 [mB/s]);
```

延迟返回

```
uri("/shop"), client-country("US") =>
limit-req-rate(key: client-addr, target-rate: 5 [r/s],
reject-rate: 10 [r/s]),    sleep(0.5);
```



企业安全工坊