基于NIDS构建纵深防御体系

点融-陈越

目录

为什么我们需要NIDS

1.bypass WAF/FW的可能性(大包/绕过/0day/SSRF)

2.内向外的恶意行为(反弹shell/内鬼/边界存在漏洞的情况/C&C通讯)

3.多数为买来的WAF/WF,自主性,灵活性差

点融的NIDS架构

1.镜像负载均衡/核心交换流量

2.Packetbeat/Bro对流量进行DPI

3.用ES/Kibana进行存储展示(原始数据,告警数据),图数据使用ArangoDB

4.主要有规则引擎/异常检测模块/资产模块组成

关于规则引擎

1.自主研发的好处:更加灵活

2.支持常规检测(正则/多模匹配等),支持频率检测(频率检测/自定义类型频率检测),支持外部函数(写外部函数/规则引擎调用,满足场景如:威胁情报),支持自定义告警方式/自定义联动方式等

场景一

1.某部门需要删除某临时表想得到实时反馈

2.添加临时规则,检测SQL语句,符合条件向相关人员发送邮件

场景二

1.某组件出现0day,无法及时升级版本修复有问题的组件

2.根据PoC编写临时规则,并联动WAF/FW进行封禁IP处理

思考

1.规则引擎可以帮助我们快速的发现一些已知的安全问题,但是这足够么?

2.甲方安全人员相对较少,不可能全部精力放在编写规则上,如何自动的发现安全问题?

异常检测模块

场景一:数据库操作

1.假设某公司仅有一个数据库,且仅有一个业务:登陆

2.那么对于数据库的操作应该仅有:

```
select uid from user where uid = ? and pwd = ?
```

场景二

1.公司业务如上,仅有登陆操作

2.那么相关的HTTP请求(提交参数)应该也仅有一个:

www.test.com/login.php　　{“uid”:”123@qq.com”,“pwd”:“123”}

举例

1.SQL注入:

HTTP:uid从邮箱变成了包含其他多种符号的字符串

SQL:AST由select uid from user where uid = ? and pwd = ? 变成了

select uid from user where uid = ? or ? = ? and pwd = ?

2.脱裤:

出现了不常见的AST:select * from user

异常检测模块

1.根据上面的思想:根据当前业务自动进行白名单的生成,以HTTP协议为例,我们以接口为单位,生成接口-参数key-参数的白名单

2.白名单生成的算法目前我们使用多种无监督聚类算法和异常点检测算法实现

3.针对业务更新迭代专门做了优化和调整

异常检测模块流程(以HTTP协议为例)

1.Request请求包含参数,且Response Code不为4XX/5XX

2.进行Path分析(Request Path中经常包含参数,如:/get/12345/name)

3.对Request Params进行解析和Feature计算

4.保存Feature,进行机器学习计算,得到相关模型

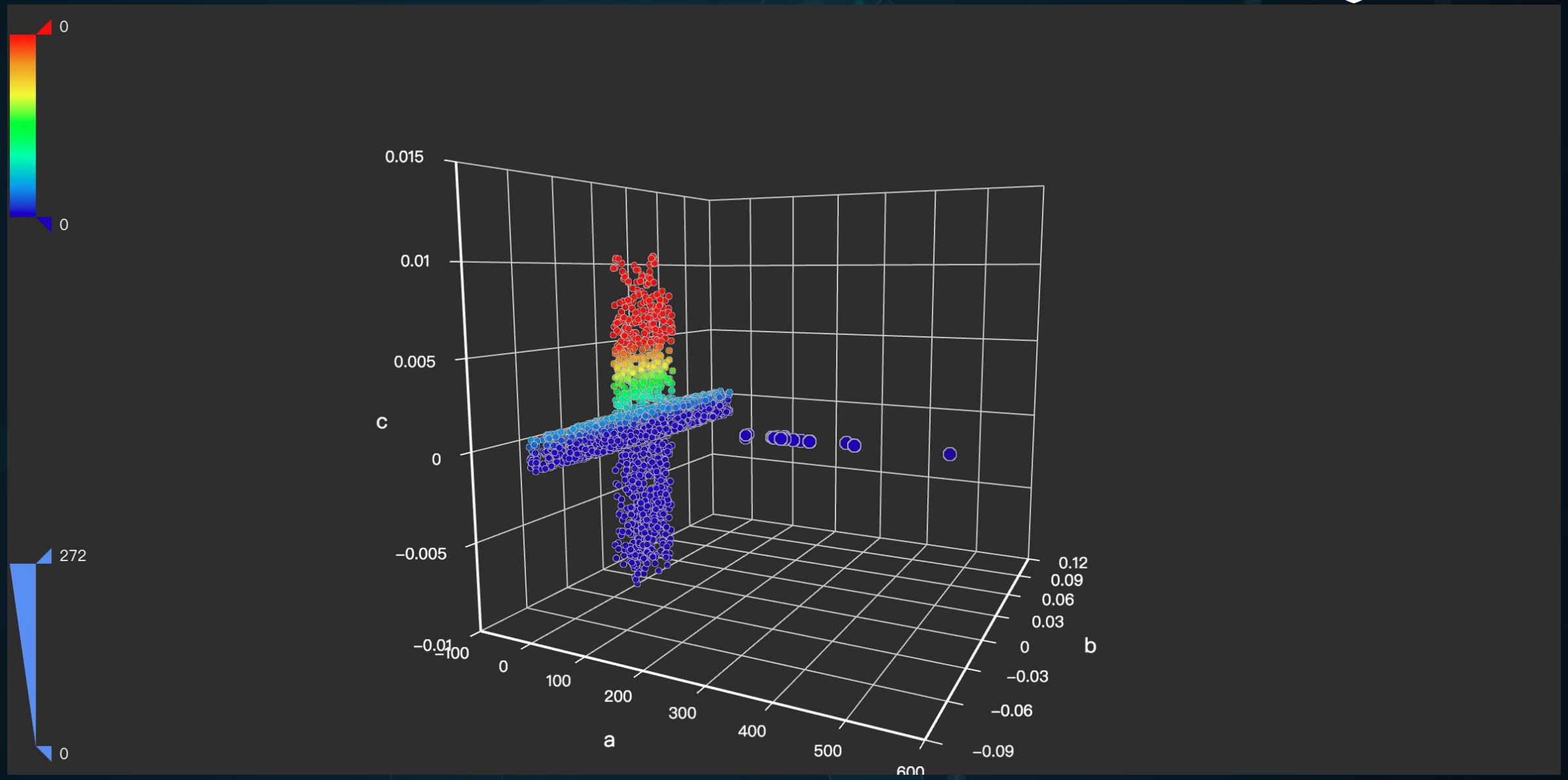
5.定期更新模型

=ajksdlfklj2l3kj14","a=hjjkhkkbjkvasdb","a=321947u81oih34uih1","a=9012348u98oihkrnjkfsd","a=as89d7f8y9huik","a=阿斯顿卡上课了12321","a=123@qq.com","a=ajsdklkjlklkl1232","a=<xml>asdad</xml>","a=curl dkaskj.dasj.com","a=or 1=1;;",]

analyser_test

```
/usr/local/bin/python3 /Users/E_Bwill/DR/smithnids/NIDS/test/analyser_test.py
[ 1  1  1 -1 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
  1  1  1  1  1  1  1  1  1 -1  1 -1 -1 -1]

Process finished with exit code 0
```


0
0
272
0
0.015
0.01
0.005
0
−0.005
−0.01
c
−100
0
100
200
300
400
500
600
a
0.12
0.09
0.06
0.03
0
−0.03
−0.06
−0.09
b

异常检测的缺点

1.异常不等于攻击(某用户喜欢脸滚键盘,但不会一直滚)

2.面对富文本型接口误报/漏报高且难以调整

3.性能问题(但是通过白名单的数据不需要过规则引擎)

4.可解释性差

5.需要一定的数据量进行白名单模型的生成

异常检测的优点

1.可以和黑名单形成互补

2.不仅可以发现攻击行为,也可以帮助甲方梳理自己的业务

3.面对大多数攻击(除逻辑漏洞),漏报极低

异常检测算法的成绩

1.国内某亿级用户的大型互联网公司

2.抽取线上流量进行测试

3.使用内部自研扫描器进行漏报检测

4.零漏报零误报

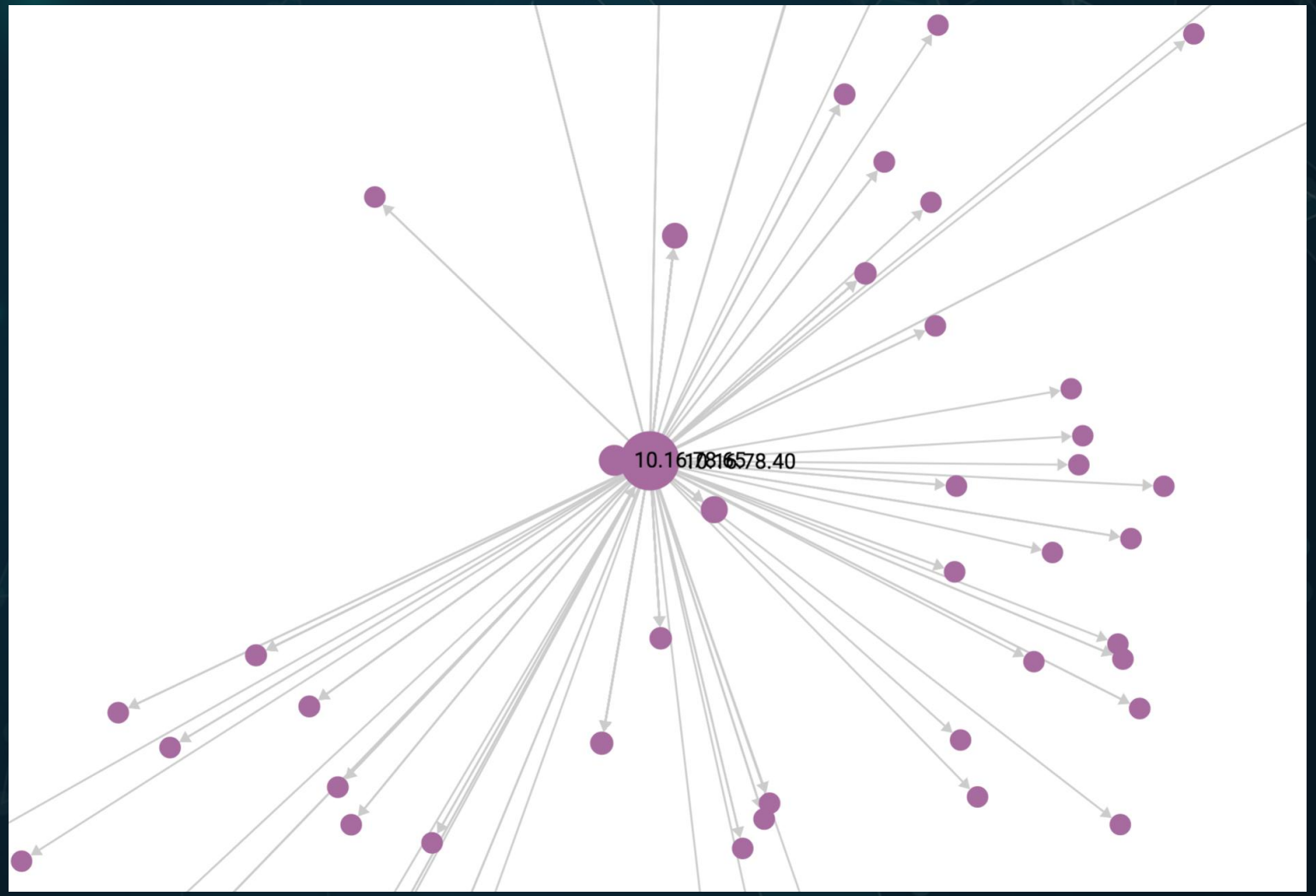
异常检测拓展

1.可以适用于多种协议

2.我们使用异常检测算法对ICMP隐秘通道检测的情况是零漏报

3.其他类似场景(可通过数据建立白名单的场景)

NIDS的资产模块



NIDS的资产模块

1.实时梳理资产信息:端口,服务,版本

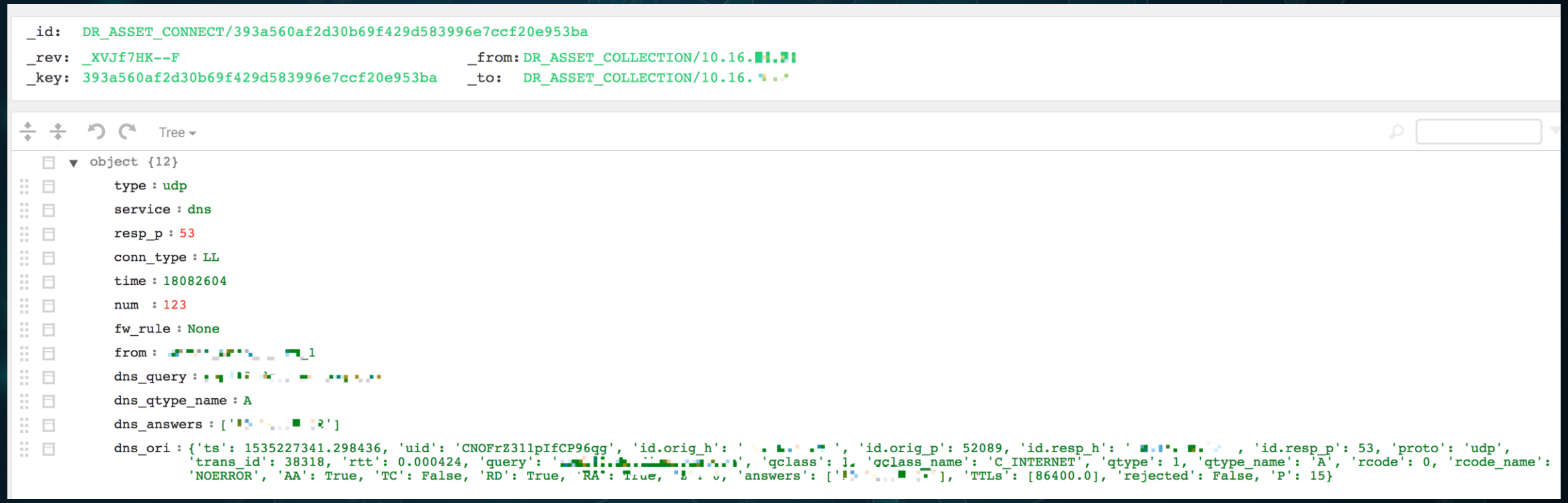
2.实时记录资产间关系:协议,频次,对于部分协议进行全量记录

3.按小时分片,帮助安全事件调查,溯源,分析

NIDS的资产模块


_id: DR_ASSET_COLLECTION/10.16.74
_rev: _W9FetrS--_
_key: 10.16.74
Code
{
"ip": "10.16.74
"port": "443;80",
"tag": "80-HTTP SERVER-Tengine",
"local": true,
"desc": "",
"other_config": "{
}

NIDS的资产模块



```
_id:  DR_ASSET_CONNECT/393a560af2d30b69f429d583996e7ccf20e953ba
_rev: _XVJf7HK--F                                  _from: DR_ASSET_COLLECTION/10.16.[redacted]
_key: 393a560af2d30b69f429d583996e7ccf20e953ba      _to:   DR_ASSET_COLLECTION/10.16.[redacted]

Tree

▼ object {12}
    type : udp
    service : dns
    resp_p : 53
    conn_type : LL
    time : 18082604
    num  : 123
    fw_rule : None
    from : [redacted]_1
    dns_query : [redacted]
    dns_qtype_name : A
    dns_answers : ['[redacted]']
    dns_ori : {'ts': 1535227341.298436, 'uid': 'CNOFrZ311pIfCP96qg', 'id.orig_h': '[redacted]', 'id.orig_p': 52089, 'id.resp_h': '[redacted]', 'id.resp_p': 53, 'proto': 'udp',
              'trans_id': 38318, 'rtt': 0.000424, 'query': '[redacted]', 'qclass': 1, 'qclass_name': 'C_INTERNET', 'qtype': 1, 'qtype_name': 'A', 'rcode': 0, 'rcode_name':
              'NOERROR', 'AA': True, 'TC': False, 'RD': True, 'RA': True, [redacted] 'answers': ['[redacted]'], 'TTLs': [86400.0], 'rejected': False, 'P': 15}
```

NIDS的资产模块

根据资产间通信和规则引擎,可以编写基于行为的规则,如:

1.X秒连接X个端口被RST视作端口扫描行为

2.WebServer通过高权限登陆数据库或SSH登陆其他任何主机视为异常

3.主动连接外网服务器会进行威胁情报检测

等等

NIDS的联动能力

即使得到告警,我们信息也有限,我们需要获得更多的信息

1.联动CMDB,得到IP基本信息和业务信息,如PM/DBA/DevOps等

2.联动FW,获得连接规则

3.联动Nginx配置文件,获得配置信息

4.联动FW白名单,避免误报

5.联动云管理端,获得云资产列表信息

等等

NIDS的不足

1.Packetbeat/Bro不支持的协议,加密的协议,场景:

- 各种反弹shell
- 各种后门

2.信息过少,安全工程师无法准确分析,场景:

- 威胁情报半夜3点告警一台服务器连接恶意服务器
- 此时抓包已经来不及,我们也不知道具体是什么进程/文件的行为

点融的HIDS

1.轻量级,仅支持Centos7(其他版本未进行稳定性测试,理论也支持不少)

2.支持与点融自研NIDS联动,实时查询进程/端口占用/文件等信息

3.对登陆/线上操作/关键文件变动记录等信息会记录并同步到Server

4.实现了一小部分可信计算的东西

5.支持各种姿势检测Rootkit

6.基线检查

7.规则引擎语法与NIDS相同,NIDS联动方式仅仅是在规则里增加一个字段

NIDS+HIDS

我们面对的信息不再是割裂的,是完整的:

- PID和PPID的cmdline;cwd;user;exe
- TCP/UDP五元组,部分协议的原始数据
- 业务信息:APPID
- FW_RULE
- NIDS/HIDS规则ID
- 威胁情报

从这里可以看到,NIDS已经不仅仅是NIDS,由于原本的规则引擎,异常检测,资产模块的完全共用,无论是N还是H都只是一个数据源而已

思路的进化

黑名单>白名单>对每次连接都要有认知能力

规则引擎>异常检测算法>HIDS/CMDB/FW等等的联动

彩蛋

考虑开源(HIDS+NIDS框架+规则引擎+现有规则)


THANKS
Q&A

点融黑帮公众号

点融黑帮技术交流群