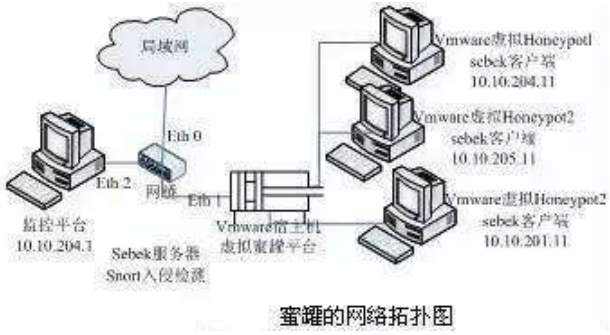
蜜罐的初步了解和 MHN 架构



蜜罐的网络拓扑图

本文主要讨论蜜罐和蜜罐网络，以及如何使用它们保护真实的系统，我们称之为这个系统为 MHN(Modern Honey Network，现代蜜网)，它可以快速部署、使用，也能够快速的从节点收集数据。

一、什么是蜜罐

蜜罐是存在漏洞的，暴露在外网或者内网的一个虚假的机器,具有以下这些特征：

1 其中重要的一点机器是虚假的，攻击者需要花费时间攻破。在这段时间内，系统管理员能够锁定攻击者同时保护真正的机器。

2 能够学习攻击者针对该服务的攻击技巧和利用代码。

3 一些蜜罐能够捕获恶意软件，利用代码等等，能够捕获攻击者的 0day,同时可以帮助逆向工程师通过分析捕获的恶意软件来提高自身系统的安全性。

4 在内网中部署的蜜罐可以帮助你发现内网中其他机器可能存在的漏洞。

蜜罐是把双刃剑，如果不能正确的使用，有可能遭受更多的攻击，模拟服务的软件存在问题，也会产生新的漏洞。

1.1 蜜罐的分类

根据设计的最终目的不同我们可以将蜜罐分为产品型蜜罐和研究型蜜罐两

类。

① 产品型蜜罐一般运用于商业组织的网络中。它的目的是减轻受组织将受到的攻击的威胁，蜜罐加强了受保护组织的安全措施。他们所做的工作就是检测并且对付恶意的攻击者。

⑴这类蜜罐在防护中所做的贡献很少，蜜罐不会将那些试图攻击的入侵者拒之门外，因为蜜罐设计的初衷就是妥协，所以它不会将入侵者拒绝在系统之外，实际上，蜜罐是希望有人闯入系统，从而进行各项记录和分析工作。

⑵虽然蜜罐的防护功能很弱，但是它却具有很强的检测功能，对于许多组织而言，想要从大量的系统日志中检测出可疑的行为是非常困难的。虽然，有入侵检测系统（IDS）的存在，但是，IDS 发生的误报和漏报，让系统管理员疲于处理各种警告和误报。而蜜罐的作用体现在误报率远远低于大部分 IDS 工具，也务须当心特征数据库的更新和检测引擎的修改。因为蜜罐没有任何有效行为，从原理上来讲，任何连接到蜜罐的连接都应该是侦听、扫描或者攻击的一种，这样就可以极大的减低误报率和漏报率，从而简化检测的过程。从某种意义上来讲，蜜罐已经成为一个越来越复杂的安全检测工具了。

⑶如果组织内的系统已经被入侵的话，那些发生事故的系统不能进行脱机工作，这样的话，将导致系统所提供的所有产品服务都将被停止，同时，系统管理员也不能进行合适的鉴定和分析，而蜜罐可以对入侵进行响应，它提供了一个具有低数据污染的系统和牺牲系统可以随时进行脱机工作。此时，系统管理员将可以对脱机的系统进行分析，并且把分析的结果和经验运用于以后的系统中。

② 研究型蜜罐专门以研究和获取攻击信息为目的而设计。这类蜜罐并没有增强特定组织的安全性，恰恰相反，蜜罐要做的是让研究组织面队各类网络威胁，

并寻找能够对付这些威胁更好的方式，它们所要进行的工作就是收集恶意攻击者的信息。它一般运用于军队，安全研究组织。

根据蜜罐与攻击者之间进行的交互，可以分为 3 类：低交互蜜罐，中交互蜜罐和高交互蜜罐，同时这也体现了蜜罐发展的 3 个过程。

① 低交互蜜罐最大的特点是模拟。蜜罐为攻击者展示的所有攻击弱点和攻击对象都不是真正的产品系统，而是对各种系统及其提供的服务的模拟。由于它的服务都是模拟的行为，所以蜜罐可以获得的信息非常有限，只能对攻击者进行简单的应答，它是最安全的蜜罐类型。

② 中交互是对真正的操作系统的各种行为的模拟，它提供了更多的交互信息，同时也可以从攻击者的行为中获得更多的信息。在这个模拟行为的系统中，蜜罐可以看起来和一个真正的操作系统没有区别。它们是真正系统还要诱人的攻击目标。

③高交互蜜罐具有一个真实的操作系统，它的优点体现在对攻击者提供真实的系统，当攻击者获得 ROOT 权限后，受系统，数据真实性的迷惑，他的更多活动和行为将被记录下来。缺点是被入侵的可能性很高，如果整个高蜜罐被入侵，那么它就会成为攻击者下一步攻击的跳板。目前在国内外的主要蜜罐产品有 DTK，空系统，BOF，SPECTER，HOME-MADE 蜜罐，HONEYD， SMOKEDETECTOR，BIGEYE，LABREA TARPIT，NETFACADE，KFSENSOR，TINY 蜜罐，MANTRAP，HONEYNET 十四种。

还有其他类型的蜜罐，比如专门捕获恶意软件的，数据库漏洞利用程序和垃圾邮件等等。当部署两个或者两个以上蜜罐时可以称之为蜜网。

附：一些蜜罐的相关网站

1.什么是蜜罐：

http://www.sans.org/security-resources/idfaq/honeypot3.php

2.蜜网：

http://www.honeynet.org/

3.蜜罐项目：

https://www.projecthoneypot.org/，攻击者的ＩＰ和攻击者的一些数据统计。

4.蜜罐的 wiki：

http://en.wikipedia.org/wiki/Honeypot_(computing)

二、现代密网(MHN)

MHN 是一个开源软件，它简化了蜜罐的部署，同时便于收集和统计蜜罐的数据。用 ThreatStream（http://threatstream.github.io/mhn/）来部署，MHN 使用开源蜜罐来收集数据，整理后保存在 Mongodb 中，收集到的信息也可以通过 web 接口来展示或者通过开发的 API 访问。

MHN 能够提供多种开源的蜜罐，可以通过 web 接口来添加他们。一个蜜罐的部署过程很简单，只需要粘贴，复制一些命令就可以完成部署，部署完成后，可以通过开源的协议 hpfeeds 来收集的信息。

MHN 支持以下蜜罐：

1.Sort:https://www.snort.org/

2.Suricata:http://suricata-ids.org/

3.Dionaea:http://dionaea.carnivore.it/,它是一个低交互式的蜜罐，能够模拟 MSSQL, SIP, HTTP, FTP, TFTP 等服务 drops 中有一篇介绍：

http://drops.wooyun.org/papers/4584

4.Conpot:http://conpot.org/

5.Kippo:https://github.com/desaster/kippo，它是一个中等交互的蜜罐，能够

下载任意文件。 drops 中有一篇介绍：http://drops.wooyun.org/papers/4578

6.Amun:http://amunhoney.sourceforge.net/，它是一个低交互式蜜罐，但是已经从 2012 年之后不在维护了。

7.Glastopf：http://glastopf.org/

8.Wordpot：https://github.com/gbrindisi/wordpot

9.ShockPot：https://github.com/threatstream/shockpot，模拟的 CVE-2014-6271，即破壳漏洞

10.p0f：https://github.com/p0f/p0f

三、MHN 的硬件要求

MHN 服务器要求：

4 GB Ram

Dual Core Processor

40 Gb Drive

蜜罐要求：

512 Mb – 1 Gb

Dual Core CPU

20 Gb Drive

具体部署时取决以蜜罐所在的位置，在防火墙后面或者在直接暴露在互联网上，被攻击次数不同，消耗的资源肯定也不同。如果只是测试着玩 256M 的内存就足够了。

四、MHN 的安装

因为 ThreatStream 有部署脚本，所以安装 MHN 很简单，我们在只安装了 OpenSSH 的 Ubuntu 14.04 LTS (64 bits)上进行测试，安装步骤如下：

```
sudo apt-get update && sudo apt-get upgrade

sudo apt-get install git

sudo git clone https://github.com/threatstream/mhn

cd /opt/mhn/scripts
```

新版本中已经做了修改，安装前可以做一下检查

```
sudo vim install_mnemosyne.sh
```

找到修改 CHANNELS，添加 shockpot.events 修改成下面的样子

```
CHANNELS='amun.events,conpot.events,thug.events,beeswarm.hive,dionaea.capture,dionaea.connections,thug.files,beeswarn.feeder,cuckoo.analysis,kippo.sessions,glastopf.events,glastopf.files,mwbinary.dionaea.sensorunique,snort.alerts,wordpot.events,p0f.events,suricata.events,shockpot.events'

sudo ./install_hpfeeds.sh

sudo ./install_mnemosyne.sh

sudo ./install_honeymap.sh
```

安装完成后执行 sudo supervisorctl status 看到四个服务起来了

机器位于公网，就可以跳过这一步，如果只是放在内网里面，则需要配置 mnemosyne 的配置文件

```
sudo vim /opt/mnemosyne/mnemosyne.cfg

ignore_rfc1918 = False
```

允许节点使用私有地址和服务器进行通信

重启服务

```
sudo supervisorctl restart mnemosyne
```

运行最后一个脚本，对 MHN 进行配置

```
sudo ./install_mhnserver.sh

================================================================
=====

MHN Configuration

================================================================
=====

Do you wish to run in Debug mode?: y/n n

Superuser email: name@example.com

/* <![CDATA[ */!function(){try{var t="currentScript"i
n document?document.currentScript:function(){for(var t=doc
ument.getElementsByTagName("script"),e=t.length;e--;)if(t
[e].getAttribute("cf-hash"))returnt[e]}();if(t&&t.previou
sSibling){var e,r,n,i,c=t.previousSibling,a=c.getAttribut
e("data-cfemail");if(a){for(e="",r=parseInt(a.substr(0,
2),16),n=2;a.length-n;n+=2)i=parseInt(a.substr(n,2),16)^
```

```
r,e+=String.fromCharCode(i);e=document.createTextNode(e),
c.parentNode.replaceChild(e,c)}}}catch(u){}}();/* ]]> */

    Superuser password:

    Superuser password: (again):

    Server base url [“http://1.2.3.4”]: http://192.168.5.3

    Honeymap url [http://1.2.3.4:3000]: http://192.168.5.
3:3000

    Mail server address [“localhost”]:

    Mail server port [25]:

    Use TLS for email?: y/n y

    Use SSL for email?: y/n y

    Mail server username [“”]:

    Mail server password [“”]:

    Mail default sender [“”]:

    Path for log file [“mhn.log”]:
```

这个过程需要比较长的时间，需要初始化数据库，同时还要插入 Snort/Suricata 规则，脚本运行结束后，直接访问配置中定义的 base url，登录后就可以配置

五、MHN 服务器安装

一旦安装了基础的服务，就能够部署蜜罐节点了，通过 web 来展示相关数据等等，可以根据具体的境况做一些简单的调整。

例如要非匿名的将收集到得攻击数据回传到 ThreatStream，要做以下操作

```
cd /opt/mhn/scripts

sudo ./disable_collector.sh
```

执行 enable_collector.sh 可以开启

如果想修改 smtp 服务的配置，可以编辑 config.py，

绝对路径

```
/opt/mhn/server/config.py

cd /opt/mhn/server

sudo vim config.py

sudo supervisorctl restart mhn-uwsgi
```

尽量不要使用超级管理员来配置，可以从 web 页面中添加其他的用户，但是所有用户都是超级用户，没有任何区别，当你删除用户的时候，实际上并没有吧该用户删除，只是在数据库中标记为"not active",同时该用户不能再次被使用，除非更改数据库。

也可以从终端去直接更改用户的密码

```
cd /opt/mhn/

source env/bin/activate

cd server

python manual_password_reset.py
```

```
deactivate
```

六、排错

如果发现服务不正常，你可以使用一些命令和排查一些日志来来判断问题出在哪里，第一个命令就是 supervisorctl，可以看到那些进程出问题了，那些在正常的运行

```
supervisorctlv status

supervisorctl  restart [process|all]

supervisorctl start [process|all]

supervisorctl stop [process|all]
```

如果你发现一个进程的状态为 ERROR 或者 FATAL，就需在 /etc/supervisor/conf.d/找到对应进程的配置文件，查看日志进行分析 https://github.com/threatstream/mhn/wiki/MHN-Troubleshooting-Guide，对遇到的问题寻求帮助：

1.honeymap 在安装的时候报错

```
hg clone http://code.google.com/p/go.net/

mv go.net/ /opt/honeymap/server/src/code.google.com/p/

go build

supervisorctl restart honeymap
```

七、MHN 的 web 接口

MHN 的 web 接口是很简洁明了的，第一次访问 web 截面时，需要输入账户名和密码，登录成功后会看见一个总结性的页面

1.在最近 24 小时内有多少攻击着攻击

2.攻击次数排在前五的 IP

3.被攻击端口排在前五的端口

4.top 5 的攻击签名

还有一些菜单选项可以进行配置或者获取更多的攻击细节

Map:查看攻击者的 IP 在全球的分布

Deploy:添加，编辑和使用蜜罐的部署脚本

Attacks:所有攻击者的列表

Payloads:所有攻击的 payload，其实只有三种蜜罐可以收集 payload(snort,dionaea.glastofp)

Rules:所有的 snort 和 suricata 规则

Seneors:有安装蜜罐节点操作的相关记录

Settings:MHN 服务的设置

攻击来源全球的分布图,运行在 3000 端口，是不需要验证就能看到的，可以做 ACL 开控制访问。

八、蜜罐节点

主要讨论蜜罐，如何安装，多么容易操作，以及会出现的问题，是否需要在进行特殊的配置等等。也会进行一些测试，看攻击者如何攻击，蜜罐对攻击行为的记录。最后会讨论 MHN 如何收集信息，以及 web 所展现的数据。

8.1 安装蜜罐节点

安装蜜罐节点很容易，所有的节点都基于同样的平台，MHN 上对每个蜜罐都有对应的安装脚本，所以安装起来是非常容易的，只需要一个服务器去安装。

我们在 Ubuntu 14.04 LTS 上进行测试。建议通过 ssh 去访问，如果不可以，请安装 ssh，修改对应的 22 端口，同时加防火墙保护

安装 ssh

```
sudo apt-get update && sudo apt-get upgrade && sudo apt-get installopenssh-server
```

修改端口为 2222

```
vim /etc/ssh/sshd_config

Port 2222
```

重启服务

```
sudo service ssh restart
```

定义的安装脚本在 MHN 服务器的 web 界面中，在"Deploy"这个选项下有所有蜜罐的安装脚本。

http://192.168.5.11/ui/manage-deploy/?script_id=11

安装完成后可以通过以下命令来检查

```
sudo netstat –tunlp

#产看当前的网络连接情况
```

```
supervisorctl status
```

8.2 Wordpot

首先，安装一个 wordpress 蜜罐在 Ubuntu 14.04 LTS 上，安装完成后 sudo netstat –tunlp 查看 80 端口是否打开， sudo supervisorctl status 可以查看服务的运行状态。 访问后该 ip,看到下图

如果想更改 wp 的插件和相关设置，

```
vim /opt/wordpot/wordpot.conf
```

使用 nmap 扫描该 80 端口

nmap -A -Pn -p80 192.168.10.21

通过 nmap 识别 80 看到的是 wordpress 2.8，运行的 http 版本是 0.96,识别出来的 python 2.7.6 的版本。这种端口扫描不会被记录，回传到 MHN 服务。

使用 wpscan 进行扫描

```
wpscan –url http://ip/
```

8.3 p0f 被动指纹识别系统

p0f 是一个被动的系统指纹识别工具，借助它可以快速识别操作系统的类型和其他的信息，它在 MHN 中部署也是很容易的，p0f 有可以根据已有的指纹，能够快速的匹配，虽然不是一个精确计算，但是识别度很高。

通过 netstat 命令的输出可以看到没有 p0f 没有去新建新的进程来实现网络监听。

如果所要保护的 ssh 端口处于防火墙后或者前端有一个 IPS/IDS，获取这些数

据的时候就比较麻烦。

如果我们在安装了 wordpot 的机器上安装 p0f，一个访问 wordpot 的 80 端口的请求不会触发 wordpot 的报警，但是 p0f 会触发，MHN 只是能够显示攻击者攻击的端口，但是不能显示关于操作系统类型，uptime 和其他的信息。p0f 的日志是纯文本的，分析和收集数据不是太容易，还好 p0f 存在 API，还有一些其他的工具，可以帮助我们快速分析。

如果需要正在进行攻击的系统系统的信息，可以从/var/log/p0f.out 中提取，可以使用 python、bash、sed、grep 等来帮助你快速的获取想要的信息。 想要获取相关 ip 的系统信息，可以使用以下这个命令：

```
grep -n "\[ 192.168.10.151.*\(syn\)\|os\s*=" /var/log/p0f.out
```

如果想要获得更多关于 p0f 的信息，可以访问 http://www.sans.org/security-resources/idfaq/p0f.php，查看相关接口信息 https://github.com/p0f/p0f/blob/master/docs/README，还有很重要的一点，p0f 把攻击的信息保存在 MHN 服务的 mongodb 中，其他的一些信息，如操作系统类型等信息没有展现出来，有可能下一个版本会对这些信息进行展示。

8.4 kippo 蜜罐

kippo 是一个中等交互式的 ssh 蜜罐，安装它很容易，但是要注意以下

1.ssh 服务默认监听的是 22 端口，/etc/ssh/sshd_config 为默认配置文件，要想正常运行，需要修改监听端口到 2222。

2.安装完成后需要重启服务。

3.supervisor 可以使 kippo 程序运行，但是不能够停止它，这需要修改部署脚本

进入 web 的 deploy 界面，选择 kippo,在部署脚本后添加下面这些东西

```
    command=/opt/kippo/start.sh

    command=sukippo -c "authbind –deep twistd -n -y /opt/kippo/kippo.tac -l /opt/kippo/log/kippo.log –pidfile /opt/kippo/kippo.pid"

    stopsignal=QUIT

    stopasgroup=true
```

编辑完成后点击 update 保存，安装完成后就可以看到 kippo 监听在 22 端口,ssh 服务在 2222 端口，下次 ssh 登录的服务器的时候需要指定 ssh -p 2222 ip，否则登录的为 kippo。

以下是一些关于 kippo 的介绍，包括如何配置以及工作原理：

1.默认的 ssh 登录账户名密码为 root/12345,也可以进行配置，配置的路径在/opt/kippo/data/userdb.txt，同时当攻击者登陆后，使用 useradd 和 passwd 添加账户和密码的时候也会保存在这个文件里面.

2.当攻击者下载文件时，下载的文件会被保存在/opt/kippo/dl

3.可以设置蜜罐中命令执行后的返回值，也可以添加命令．例如来修改 ifconfig 命令的输出，可以编辑/opt/kippo/txtcmds/sbin/ifconfig 这个文本文件。

4.有一个特点就是，当攻击者输入 exit 想退出的时候，其实没有退出，只是显示退出，给攻击者一个假象，以为回到的他的本机，他接下来的操作还是会被记录到日志中。

5.你可以更改/opt/kippo/honeyfs，里面保存模拟的系统的文件等内容，使蜜罐更像真实环境。

6.log 存储在/opt/kippo/log/kippo.log，你也可以修改配置，把 log 存储到数据库中，数据库的表结构在 /opt/kippo/doc/sql 目录中。

7.每一次登录成功后的操作都会把日志单独再存储一份，存储的路径在/opt/kippo/log/tty,可以通过在/opt/kippo/utils 中的 playlog.py 脚本，来重现这个操作过程。

安装和配置 kippo 后，如果有人尝试登录，会被记录登录所使用的用户名和密码。

我们用 nmap 来扫描，看到以下信息

根据 nmap 的输出，我们看到 22 端口已经被识别为 ssh 服务，其中输出的 ssh 版本能够在 kippo.cfg 中被修改,这种简单的扫描是不会在 MHN 服务上产生记录的 如果我们使用 hydra(https://www.thc.org/thc-hydra/),去进行暴力破击(hydra -l root -P darkc0de.lst ssh://192.168.10.21),hydra 在连接的时候 会出现一些问题，但是我们使用 medusa(http://foofus.net/goons/jmk/medusa/medusa.html)时，暴力破解的命令如下

```
medusa -u root -P darkc0de.lst -M ssh -h 192.168.10.21
```

每进行一次尝试登录，都会产生一条扫描记录，如果想更了解 kippo，可以参考 http://edgis-security.org/honeypot/kippo/

8.5 Suricata 网络入侵检测和阻止引擎

Suricata 的部署脚本没有问题，直接执行就好了，部署完成后需要重启服务。Suricata 是一个 IDS,监听的接口为 eth0,规则是通过 crontab 中的 /etc/cron.daily/update_suricata_rules.sh 脚本，每天都从 MHN 服务更新的(可以开启或者关闭规则)

正如 supervisor 显示的，只有一个进程，没有生成其他的服务，因为他只是一个 IDS,如果我们使用 nmap 进行扫描，只看到一个 ssh 服务，这侧扫描会被记录，同时在 MHN 中显示

Sensor Log:

MHN Server:

Suricata 只会把攻击报告给 MHN 服务，但是攻击细节存储在 Suricata 的日志中，不会被显示

1./var/log/suricata.log:日志记录了 Suricata 进程相关的信息，如启动报错，错误的规则等等，这份日志由 supervisor 产生

2./var/log/suricata/：设计到 suricata 操作的输入日志，报警日志，http 请求日志，dns 请求日志等等

3./var/log/suricata/http.log：http 请求，不是报警

4./var/log/suricata/fast.log：一行存储一个报警

5./var/log/suricata/eve.json：格式为 json，包含报警和相关的事件

6./var/log/suricata/unified2.alert：报警文件为 Barnyard2 格式（http://www.forensicswiki.org/wiki/Barnyard2）

Suricata 的配置文件：

1./opt/suricata/etc/suricata/classification.config :报警的优先级

2./opt/suricata/etc/suricata/reference.config ： 使用的漏洞数据库

3./opt/suricata/etc/suricata/suricata.yaml ： suricata 的配置文件

4./opt/suricata/etc/suricata/reference.config ： 通过配置来减少报警的数量，例如不想记录所有 ICMP 请求的来源为一样的ＩＰ

IDS/IPS 配置是很复杂的，这里只是简单的说明，需要详细了解 suricata 的，访问这里 http://suricata-ids.org/docs/

8.6 snort

snort 是像 Suricata 一样的 IDS/IPS，和 Suricata 使用类似的规则，同时工作方式也很像，同样部署脚本没有问题，通过 crontab.daily 来更新规则库，不会其他用来监听连接的服务

它的配置文件和 Suricata 使用的 suricata.yaml 很像，它的配置文件为 snort.conf，日志存储在/var/log/snort/alert 目录下.

想更加了解 snort，访问 https://www.snort.org/

8.7 Shockpot

Shockpot 是一个 web 蜜罐，用来模拟破壳漏洞,CVE-2014-6271 http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2014-6271,破壳漏洞是一个影响很广泛的漏洞，影响 Mac OsX,Linux 和 Unix，这个漏洞产生在 GNU Bash(shell)上，允许远程命令执行，虽然已经出现了补丁，但不是所有的机器都有更新。使用这个蜜罐，可以用来捕获这个漏洞的利用代码，

关于这个漏洞的详情 https://shellshocker.net/

部署脚本有一些问题，进入 Deploy 页面，选择 Shockpot 对期做一点修改，在＂#Config for supervisor. ＂和＂EOF＂中加入一下代码

```
[fetch_public_ip]

enabled = false

urls = [“http://api.ipify.org",”

http://bot.whatismyipaddress.com/”]
```

如果服务器有公网 IP，就加入上面这一段.

安装完成后重启服务

看到以下运行的进程，有一个新的服务在 80 上监听

如果通过浏览器去访问 http://ip ，会看到"It Works!"这个默认的 Apache 页面

```
telnet ip 80

GET / HTTP/1.0
```

返回以下头部

显示这是一个 Debian 服务器，使用了 PHP 和 OpenSSL

```
nmap -A -p80 192.168.10.21
```

上面的这些操作 MHN 是不会记录的，下面我们来攻击蜜罐

```
curl -H “User-Agent: () { :; }; /bin/ping-c 1 TARGET_HOST_IP” http://SHOCKPOT_IP
```

可以看到日志里输出/var/log/shockpot/shockpot.log report

MHN 中的记录

更多关于破壳的利用可以去 https://blog.cloudflare.com/inside-shellshock/上观看

8.8 Conpot

Conpot 是一个工业控制系统和 Scada 蜜罐，针对这些类型的攻击是在近几年快速增长，是因为安全在工业系统中要求较低，才造成了现在这种场面，安全专家们也在想办法保护这些脆弱的系统。

安装简单，安装完成后需要重启服务，使用 supervisorctl status 可以看到一个进程，但是使用 netsta 可以看到好几个监听的端口。

Conpot 使用了一些模块，能够提供以下服务

```
MODBUS TCP -> tcp/502

HTTP -> tcp/80

SNMP -> udp/161

S7COMM -> tcp/102
```

使用 nmap 扫描

```
nmap -A -p1-1000 192.168.10.21

nmap -sU -p161-script snmp-sysdescr 192.168.10.21
```

8.9 Glastopf

Glastopf 是最好的 web 蜜罐，它模拟了很多漏洞，特别是远程文件包含漏洞，可以捕获到攻击只插入的文件．部署依旧很简单，安装完成后重启服务，你会看到一个进程

模拟的 http,mongodb 只在本地监听.

```
nmap -A -p80 192.168.10.21
```

你能看到模拟的是 apache，我们使用 web 漏洞扫瞄器像 niko(https://cirt.net/Nikto2) nikto -h 192.168.10.21

nikto 报告了 5536 个项目，显然蜜罐对攻击者很有吸引力，我们到 MHN 的 web 界面上看到所有的攻击报告，但是如果我们点击 Payload，选择 glastopf.events，我们可以看到很多记录，在“Regex term”中输入"rfi"

对于这些被下载的文件，我们能够看到它们的 md5 值，这些文件都保存在 /opt/glastopf/dataGlastopf 是很好配置的，/opt/glastopf/glastopf.cfg 配置文件，日志/opt/glastopf/log/glastopf.log.

官网

https://www.honeynet.org/sites/default/files/files/KYT-Glastopf-Final_v1.pdf

8.10 dionaea

dionaea 将有漏洞的服务暴露出来，允许攻击者发送保存任何文件，安装简单

服务列表

```
tcp/5060 —> SIP Protocol

tcp/5061 —> SIP Protocol over TLS

tcp/135 —> Remote procedure Call RPC

tcp/3306 —> MySQL Database

tcp/42 —> WINS Protocol
```

```
tcp/21 —> FTP Protocol

tcp/1433 —> MSSQL

tcp/445 —> SMB over TCP

udp/5060 —> SIP Protocol

udp/69 —> TFTP

nmap -sS -sV -p1-65535 192.168.10.21
```

在扫描结果中，我们看到 ftp 和 smb 服务的指纹是＂Dionaea Honeypot＂，这种扫描产生的记录都被记录了，扫描 UDP

```
 nmap -sU -sV -p69,5060 192.168.10.21
```

我们没有修改 banner，会被 nmap 识别，如果需要修改，编辑 /usr/lib/dionaea/python/dionaea 这个 python 文件，需要更加了解他，访问 http://www.securityartwork.es/2014/06/05/avoiding-dionaea-service-identification/?lang=en，配置文件在 /etc/dionaea/dionaea.conf,日志保存在 /var/dionaea/log/dionaea.log

使用 nessus 进行扫描，发现了 53 个问题，45 个 info， 3 个紧急

我们看到有 MS04-007 漏洞，这个漏洞能够执行任意代码，通过向主机发送 ASN.1 编码后的数据包，Metasploit 中使用的模块是“MS04-007 Microsoft ASN.1 Library Bitstring Heap Overflow”,使用 Metasploit 发起攻击,会返回“The SMB server did not reply to our request”，这种攻击会报告给 mhn,攻击的细节保存在 /var/dionaea/bistreams/.

九、总结

通过本文，我们已经大概了解了 mhn 的总体概况，怎么安装和怎么去部署蜜罐节点，还有这些蜜罐大概的一些情况。如果你清楚你想做的，部署蜜网是很有用的．可以获取攻击者更准确的信息，然后来做防御。

在部署蜜罐节点时，尽可能的根据具体情况来组合部署，这样可以是攻击者花费更多的时间，从中获取更多的信息，争取到更多的响应时间。

例如 snort,Glastopf,Dionaea 和 kippo，在具体部署之前先要好好测试，避免在真实环境中出现意想不到的问题。

MHN 中支持的开源蜜罐种类很多，基本已经涵盖了现有所有的开源蜜罐，可以根据具体的业务场景来组合。也可以根据具体的场景来做二次开发，因为回传的数据有些简单了。如果是放置在内网中，报警功能就很有必要性，内网中得蜜罐只是为了能够延缓攻击的进度，以及及时发现入侵，从而切断入口。