

实验报告

网络入侵与防御实验

**专业班级: 信安 21XX 班**

**学 号： U2021XXXXX**

**姓 名： MTX**

**指导教师： 余添龙**

**报告日期： 2023 年 1 月 16 日**

**网络空间安全学院**

目 录

[一、 实验内容 3](#_Toc156658661)

[二、 实验过程 3](#_Toc156658662)

[**实验1：利用入侵检测系统检测 ICMP恶意扫描 3**](#_Toc156658663)

[**1. 下载docker desktop（windows） 3**](#_Toc156658664)

[**2. 安装snort 4**](#_Toc156658665)

[**3. 检测ICMP数据包 5**](#_Toc156658666)

[**4. 检测NMAP扫描 8**](#_Toc156658667)

[**实验2：利用网络入侵检测系统检测注入式攻击 9**](#_Toc156658668)

[**练习1：物联网Mirai爆破攻击 11**](#_Toc156658669)

[三、 实验心得 13](#_Toc156658670)

1. **实验内容**

实验1：利用入侵检测系统检测 ICMP恶意扫描

实验2：Snort 检测 Log4j 攻击

练习1：物联网 Mirai 爆破攻击

1. **实验过程**

## 实验1：利用入侵检测系统检测 ICMP恶意扫描

### 下载docker desktop（windows）

之前并没有怎么仔细了解过docker，借此机会，仔细在网上查阅了一番：Docker是一个开源的容器化平台，用于构建、发布和运行应用程序和服务。它提供了一种轻量级的虚拟化技术，使应用程序能够在隔离的容器中运行，而不需要传统的完整的虚拟机。以下是Docker的一些关键特点和概念：

1）容器化：Docker利用Linux内核的容器功能，将应用程序及其依赖项打包成一个独立的、可移植的容器。每个容器都是一个隔离的运行环境，具有自己的文件系统、进程空间和网络接口，可以在任何支持Docker的主机上运行。

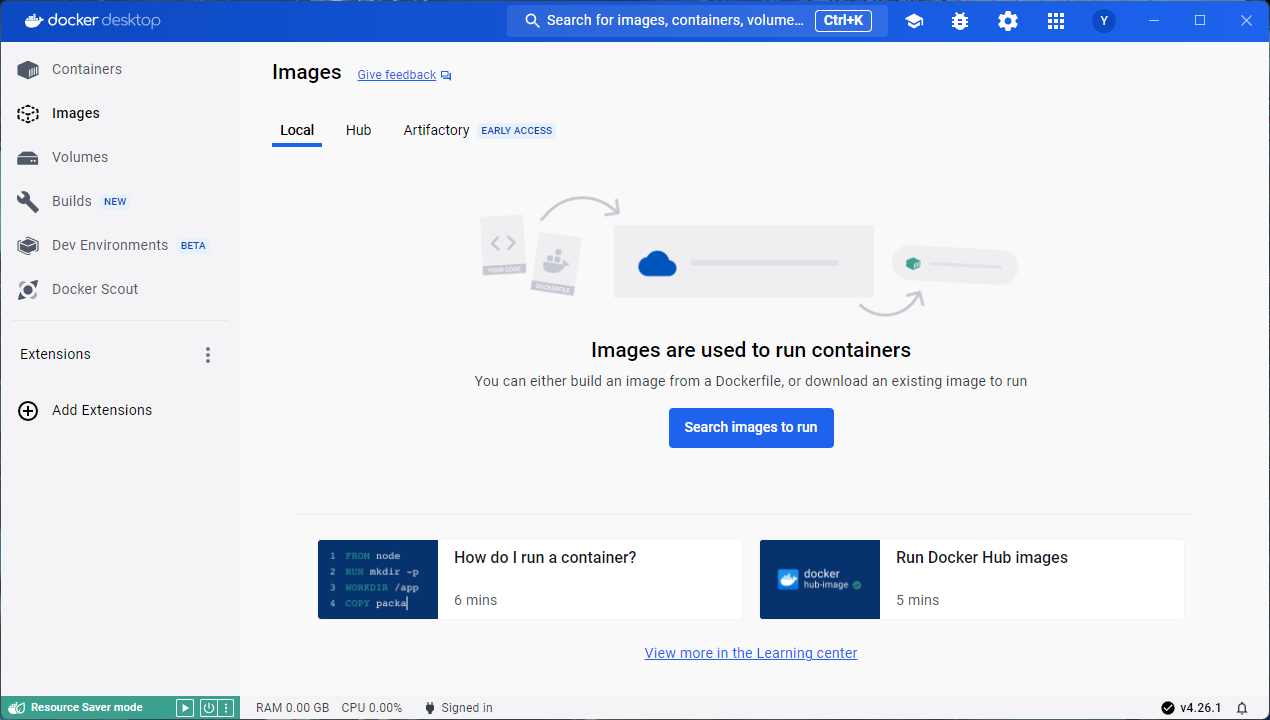
2）轻量级和快速：相比传统虚拟机，Docker容器更加轻量级，启动和停止速度更快。这是因为容器共享主机的操作系统内核，并且没有额外的操作系统运行。、

3）可移植性：Docker容器可以在不同的操作系统和云平台上运行，提供了极高的可移植性。开发人员可以在其开发环境中构建和测试容器，然后轻松地将其部署到生产环境中，而无需担心依赖项和环境的差异。

4）高度可扩展：Docker可以轻松地进行水平扩展，通过在多个主机上运行容器实例来分布负载。同时，它还提供了强大的编排工具，如Docker Compose和Kubernetes，用于管理和协调大规模容器化应用的部署和运行。

5）生态系统和镜像仓库：Docker拥有庞大的生态系统，提供了大量的公共镜像仓库，供开发人员共享和获取预构建的容器镜像。这使得应用程序的部署和管理变得更加简单和高效。

在给定网站<https://docs.docker.com/desktop/install/windows-install/>安装即可，下载后如图1所示.

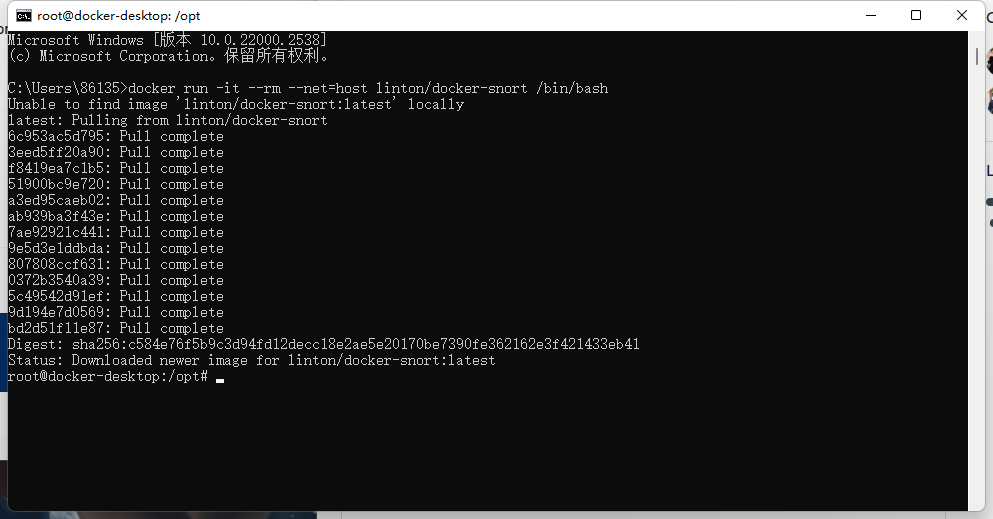


**图1 docker desktop**

### 安装snort

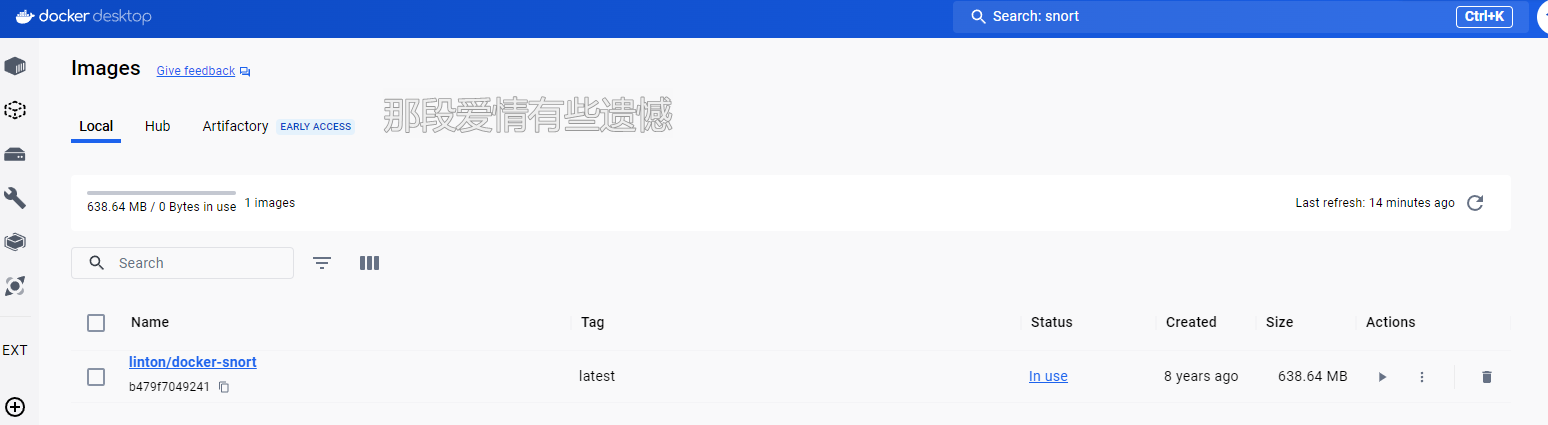
Snort 能够对网络上的数据包进行抓包分析，但区别于其它嗅探器的是，它能根据所定义的规则进行响应及处理。Snort 通过对获取的数据包，进行各规则的分析后，根据规则链，可采取 Activation（报警并启动另外一个动态规则链）、Dynamic（由其它的规则包调用）、Alert（报警），Pass（忽略），Log（不报警但记录网络流量）五种响应的机制。 Snort 有数据包嗅探，数据包分析，数据包检测，响应处理等多种功能，每个模块实现不同的功能，各模块都是用插件的方式和 Snort 相结合，功能扩展方便。例如，预处理插件的功能就是在规则匹配误用检测之前运行，完成 TCP 碎片重组，http 解码，telnet 解码等功能，处理插件完成检查协议各字段，关闭连接，攻击响应等功能，输出插件将得理后的各种情况以日志或警告的方式输出。

根据命令$ docker run -it --rm --net=host linton/docker-snort /bin/bash安装snort image，如图2所示



**图2 安装Snort**

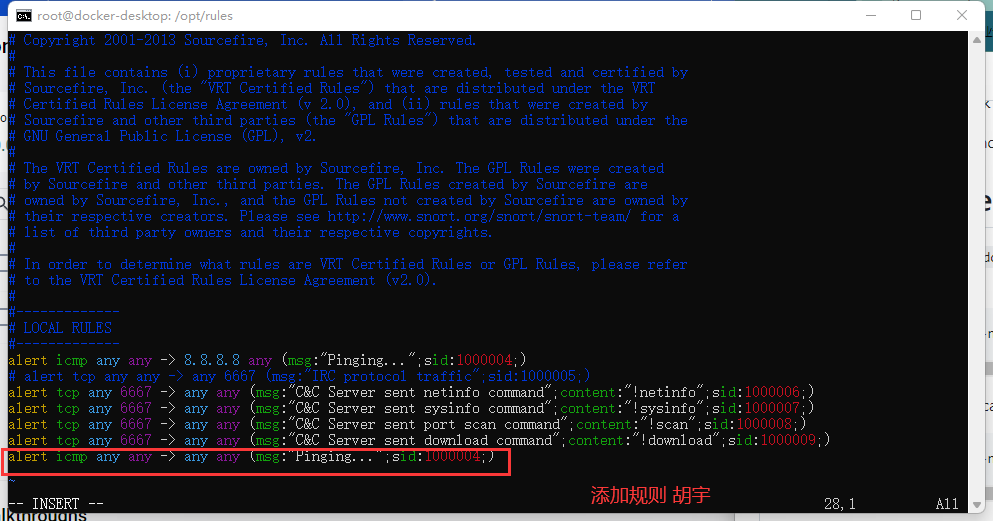
此时我们能发现一个snort image已经in use了，如图3所示，证明安装成功



**图3 snort image状态 in use**

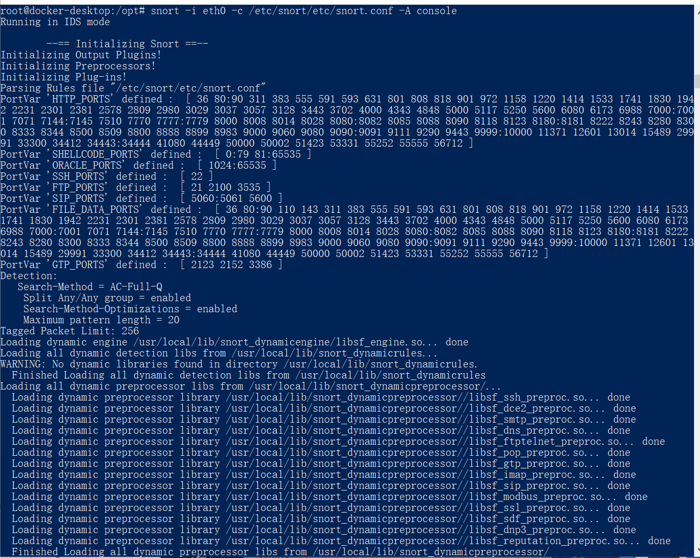
### 检测ICMP数据包

运行snort，并打开rules，补充一条Ping规则：alert icmp any any -> any any (msg:"Pinging...";sid:1000004;)，如图4所示



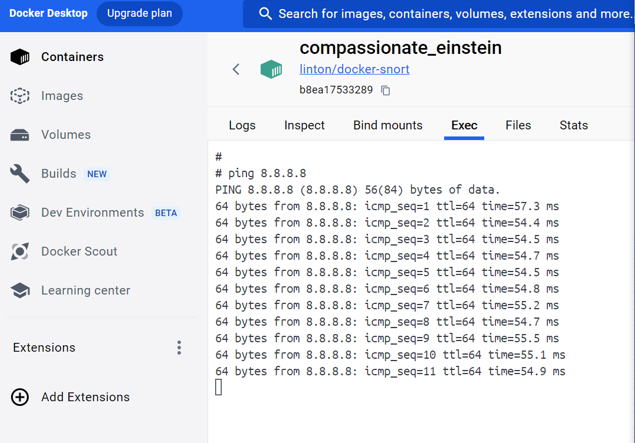
**图4 补充规则**

在Snort-Docker的Terminal中开启Snort，命令：snort -i eth0 /etc/snort/etc/snort.conf -A console，如图5所示。



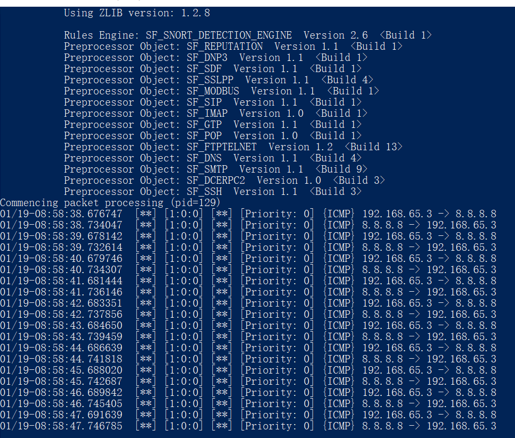
**图5 运行Snort**

在Docker中用Ping进行验证，如图6所示。



**图6 ping命令检测**

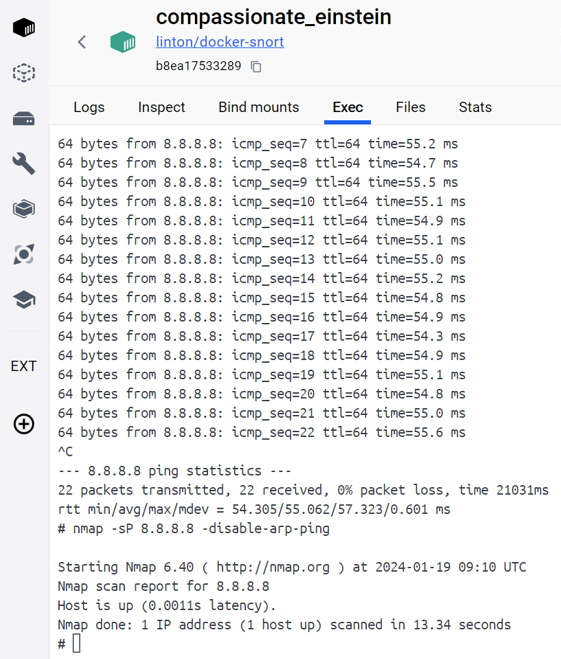
其结果如图7所示，snort成功检测！



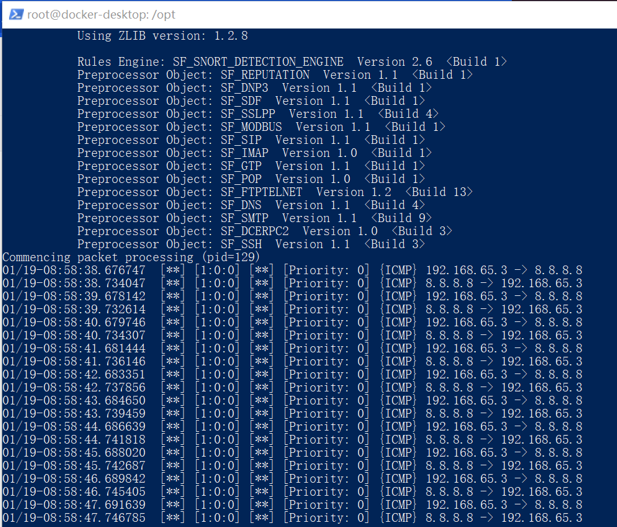
**图7 snort成功检测**

### 检测NMAP扫描

首先，使用命令：sudo apt-get install nmap安装NMAP ，并和上文一样在Snort规则中加入规则：alert icmp any any -> 8.8.8.8 any (msg:"NMAP Ping Sweep Scan";dsize:0;sid:1000004;)，然后在Docker UI界面使用命令 nmap -sP 8.8.8.8 -disable-arp-ping 开启NMAP扫描，如图8所示，可以发现snort检测到NMAP扫描结果，如图9所示。



**图8 Docke命令**



**图9 snort成功检测**

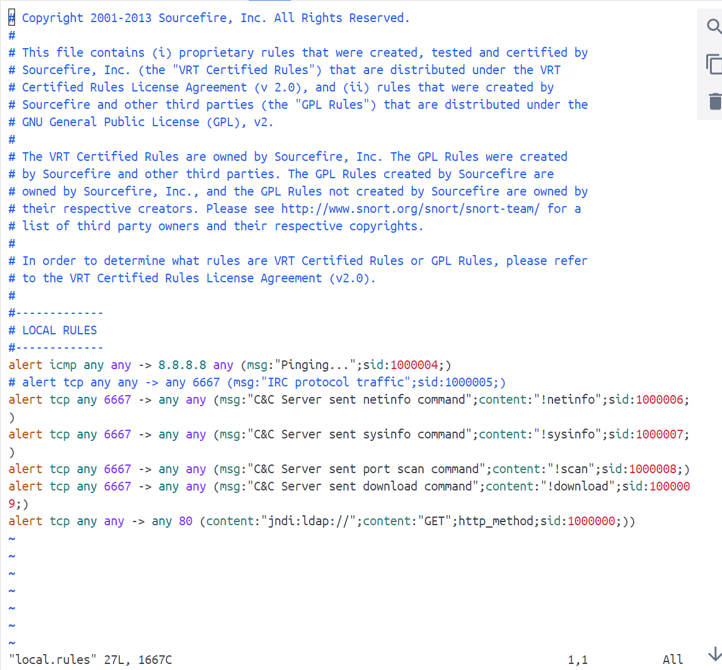
## 实验2：Snort 检测 Log4j 攻击

用 wireshark 打开 log4j-attempts.pcap 文件，并在 filter 栏中输入： http.user\_agent contains "jndi"，观察 http user agent 域的网络流量规律，不难发现http报文中的user\_agent为jndi:ldap，所以构造Snort规则：alert tcp any any -> any 80 (content:"jndi:ldap://"; content:"GET"; http\_method; sid:1000000;)。这条规则是一个基于Snort规则语法的网络入侵检测系统（Intrusion Detection System，IDS）规则。它的目的是检测TCP协议的流量，从任何源IP和任何源端口发送到目标端口80的流量。

http\_method：这个选项用于指定要匹配的HTTP方法。HTTP方法是在HTTP请求中指定的动作，常见的方法包括GET、POST、PUT、DELETE等。通过在规则中使用http\_method选项，可以只检测特定的HTTP方法的请求。例如，http\_method: GET;表示只匹配GET方法的请求。

content：这个选项用于指定要匹配的内容。内容可以是任意字符串，包括请求头、请求正文或响应内容等。可以使用通用的字符串匹配规则，也可以使用正则表达式进行更灵活的匹配。在规则中使用content选项，可以检测特定的字符串或模式是否存在于HTTP流量中。例如，content:"jndi:ldap://";表示匹配包含字符串"jndi:ldap://"的HTTP流量。

然后将对应规则添加在snort中，如图10所示。



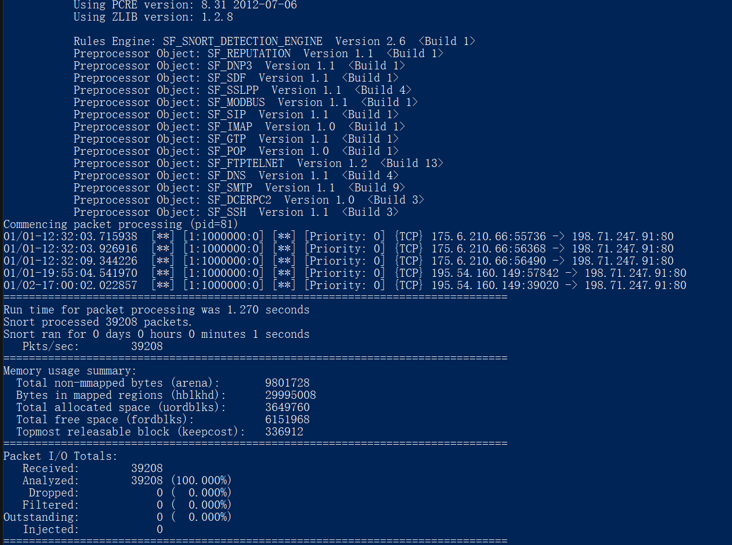
**图10 添加规则**

接着用命令拷贝log4j-attempts.pcap到docker中，如图11所示。

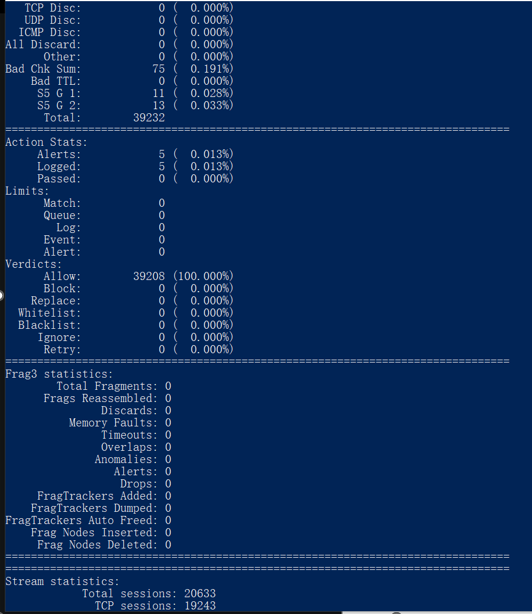


**图11 成功拷贝log4j-attempts.pcap 文件**

最后，启动Snort检测，查看捕获的攻击网络包与安全警告，对应命令如下： Snort -r ./pcap/loj4-attempts.pcap -c /etc/snort/etc/snort.conf -A console，如图12，13所示。



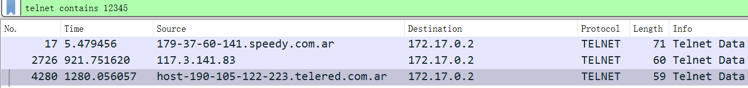
**图12 Snort检测结果1**



**图13 Snort检测结果2**

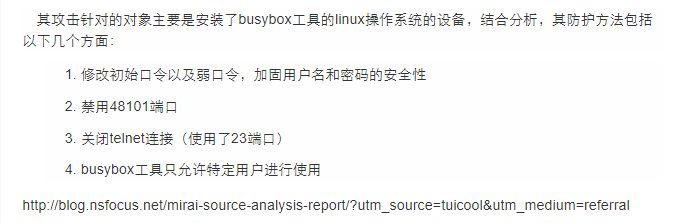
## 练习1：物联网Mirai爆破攻击

通过使用Wireshark打开名为"mirai.pcap"的数据包文件，并按照提示的攻击方法，在过滤器中输入"telnet contains 12345"的过滤条件，可以看到相关的报文信息，如图14所示



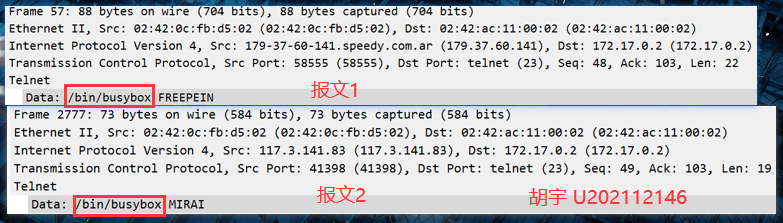
**图14 Wireshark过滤报文**

上网查阅关于Mirai的资料，发现其攻击对象主要是安装了busybox工具的Linux操作系统的设备，如图15所示

-

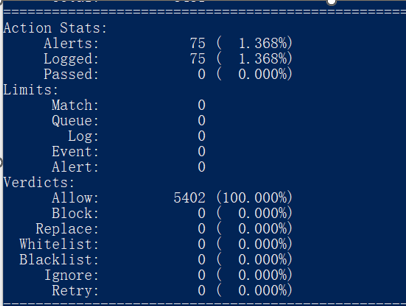
**图15 mirai攻击特性**

而查看对应报文，发现确实都是用了bin/busytext，如图16所示，于是我们可以用此到做检测命令，编写snort规则：alert tcp any any -> any any (content: "/bin/busybox",sid:1000000;)。

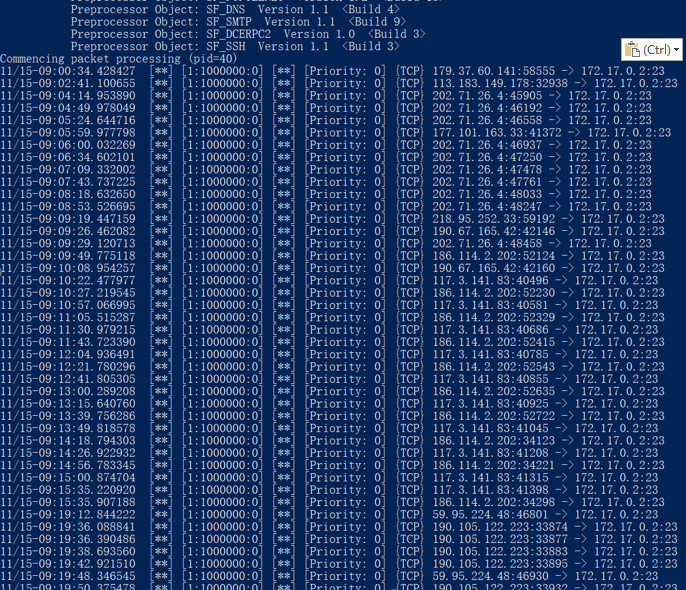


**图16 报文bin/busybox特征**

然后使用Snort分析，检测结果如图17，18所示，成功检测！



**图17 Snort检测结果1**



**图18 Snort检测结果2**

1. **实验心得**

本次实验旨在使用Docker容器化环境，并结合Snort和Wireshark来检测Mirai僵尸网络攻击。实验的重点是通过Snort规则中的content：/bin/busybox来识别Mirai恶意软件的活动。

首先，我们使用Docker创建了一个隔离的环境，其中包括运行Snort和Wireshark的容器。这样可以确保在实验过程中不会影响主机系统的安全和稳定性。

接下来，我们配置了Snort规则，将关注点放在Mirai恶意软件的特征之一——使用/bin/busybox作为可执行文件的攻击。我们修改了Snort规则文件，通过添加content：/bin/busybox来检测数据包中是否包含此关键字。这样，当Mirai恶意软件尝试利用/bin/busybox进行攻击时，Snort将触发警报。

然后，我们使用Wireshark捕获网络流量，并将过滤器设置为仅显示与我们关注的Mirai攻击相关的数据包。通过在过滤器中输入"content：/bin/busybox"，我们只看到与该关键字相关的数据包。这使我们能够更直观地分析Mirai攻击活动，包括源IP地址、目标IP地址、协议类型等。

总结而言，通过使用Docker容器化环境，结合Snort规则中的content：/bin/busybox和Wireshark的过滤功能，我们能够有效地检测和分析Mirai恶意软件的活动。这种综合的方法可以增强网络安全的监测和应对能力，帮助我们及时发现并应对潜在的僵尸网络攻击。