安全牛课堂
edu.aqniu.com

CTF从入门到提升

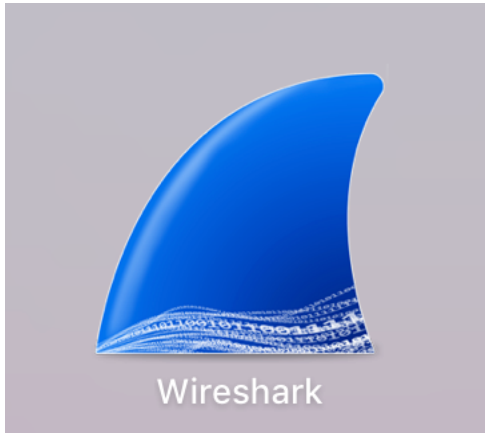
MISC-流量分析

讲师： andog

安全牛课堂
电话：010-51626887-813
地址：北京市海淀区中关村南大街2号807

Wireshark
简介

Wireshark（前称Ethereal）是一个网络封包分析软件。网络封包分析软件的功能是撷取网络封包，并尽可能显示出最为详细的网络封包资料。Wireshark使用WinPCAP作为接口，直接与网卡进行数据报文交换。



捕捉过滤器：数据经过的第一层过滤器，它用于控制捕捉数据的数量，以避免产生过大的日志文件。用于决定将什么样的信息记录在捕捉结果中。需要在开始捕捉前设置。

显示过滤器：在捕捉结果中进行详细查找。它允许您在日志文件中迅速准确地找到所需要的记录。他们可以在得到捕捉结果后随意修改。

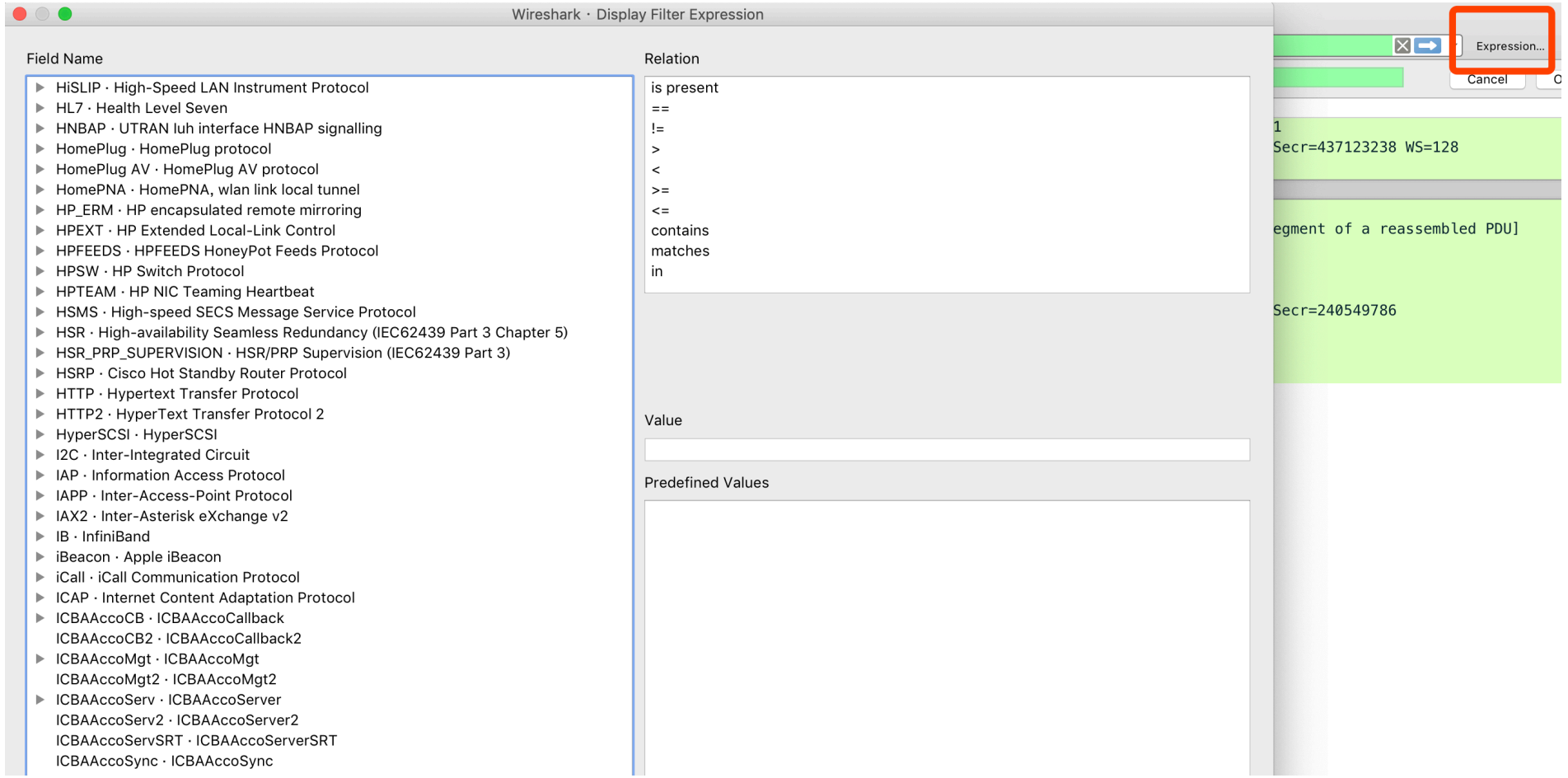
显示过滤器
基本语法

语法：

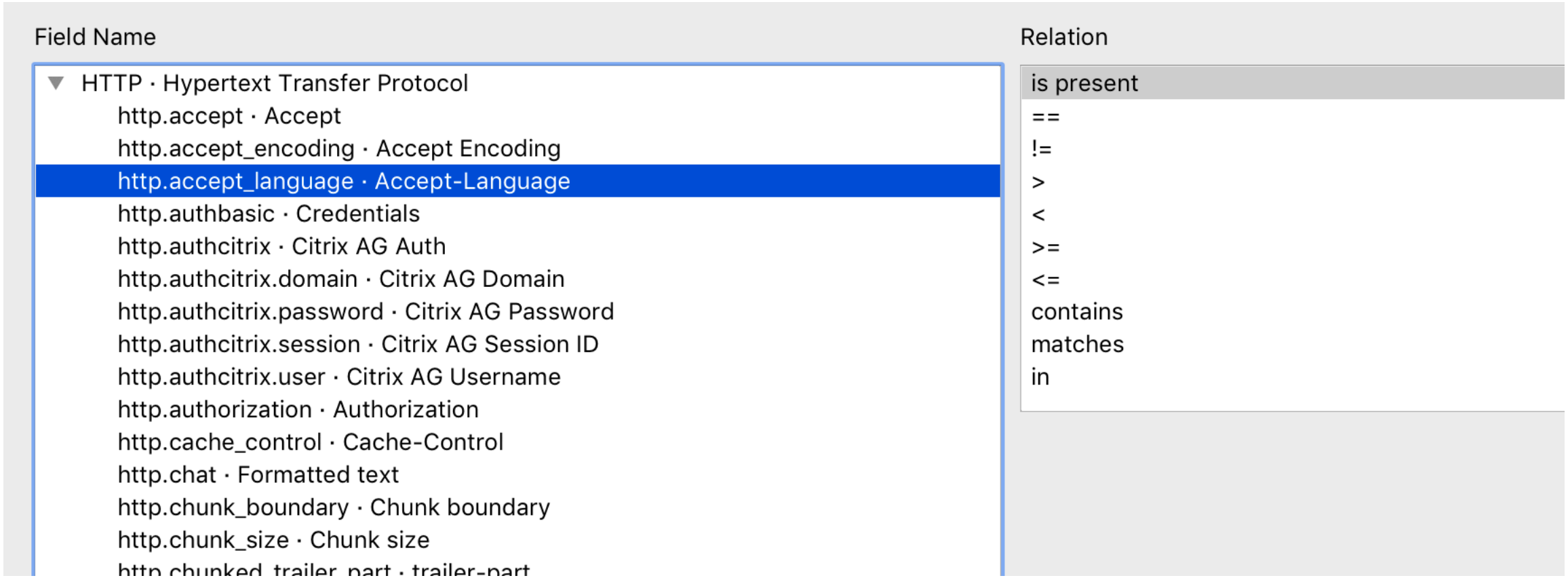
语法:	Protocol	.	String 1	.	String 2	Comparison operator	Value	Logical Operations	Other expression
例子:	http		request		method	==	"POST"	or	icmp.type

string1和string2是可选的。
依据协议过滤时，可直接通过协议来进行过滤，也能依据协议的属性值进行过滤

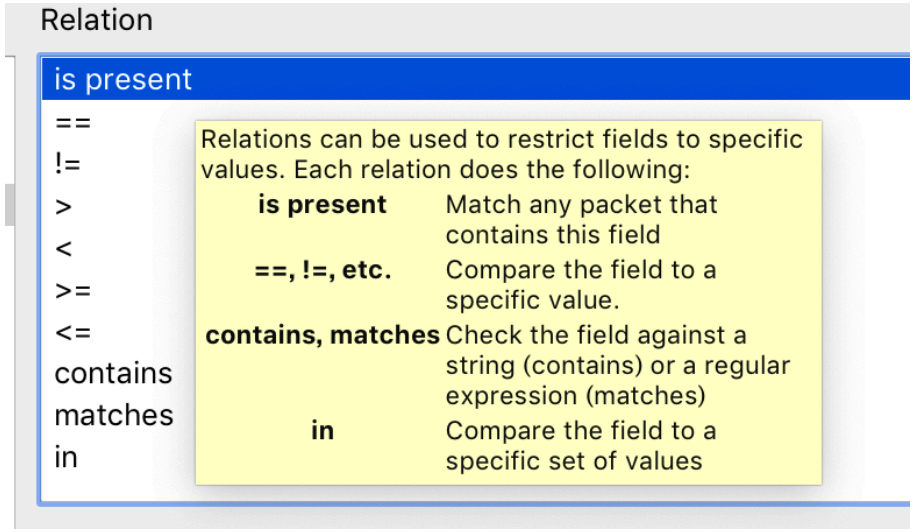
Protocol（协议）



string1和string2



Comparison operators （比较运算符）



英文写法：	C语言写法：	含义：
eq	==	等于
ne	!=	不等于
gt	>	大于
lt	<	小于
ge	>=	大于等于
le	<=	小于等于

Logical expressions

英文写法：	C语言写法：	含义：
and	&&	逻辑与
or	||	逻辑或
xor	^^	逻辑异或
not	!	逻辑非

注意：逻辑异或是一种排除性的或。当其被用在过滤器的两个条件之间时，只有当且仅当其中的一个条件满足时，这样的结果才会被显示在屏幕上。
让我们举个例子："tcp.dstport 80 xor tcp.dstport 1025"
只有当目的TCP端口为80或者来源于端口1025（但又不能同时满足这两点）时，这样的封包才会被显示。

Wireshark过滤数据

按协议进行过滤：

snmp || dns || icmp	显示SNMP或DNS或ICMP封包。

按协议的属性值进行过滤：

ip.src == 10.230.0.0/16	显示来自10.230网段的封包。
tcp.port == 25	显示来源或目的TCP端口号为25的封包。
tcp.dstport == 25	显示目的TCP端口号为25的封包。
http.request.method== "POST"	显示post请求方式的http封包。
http.host == "tracker.1ting.com"	显示请求的域名为tracker.1ting.com的http封包。

内容过滤语法

contains ： Does the protocol, field or slice contain a value

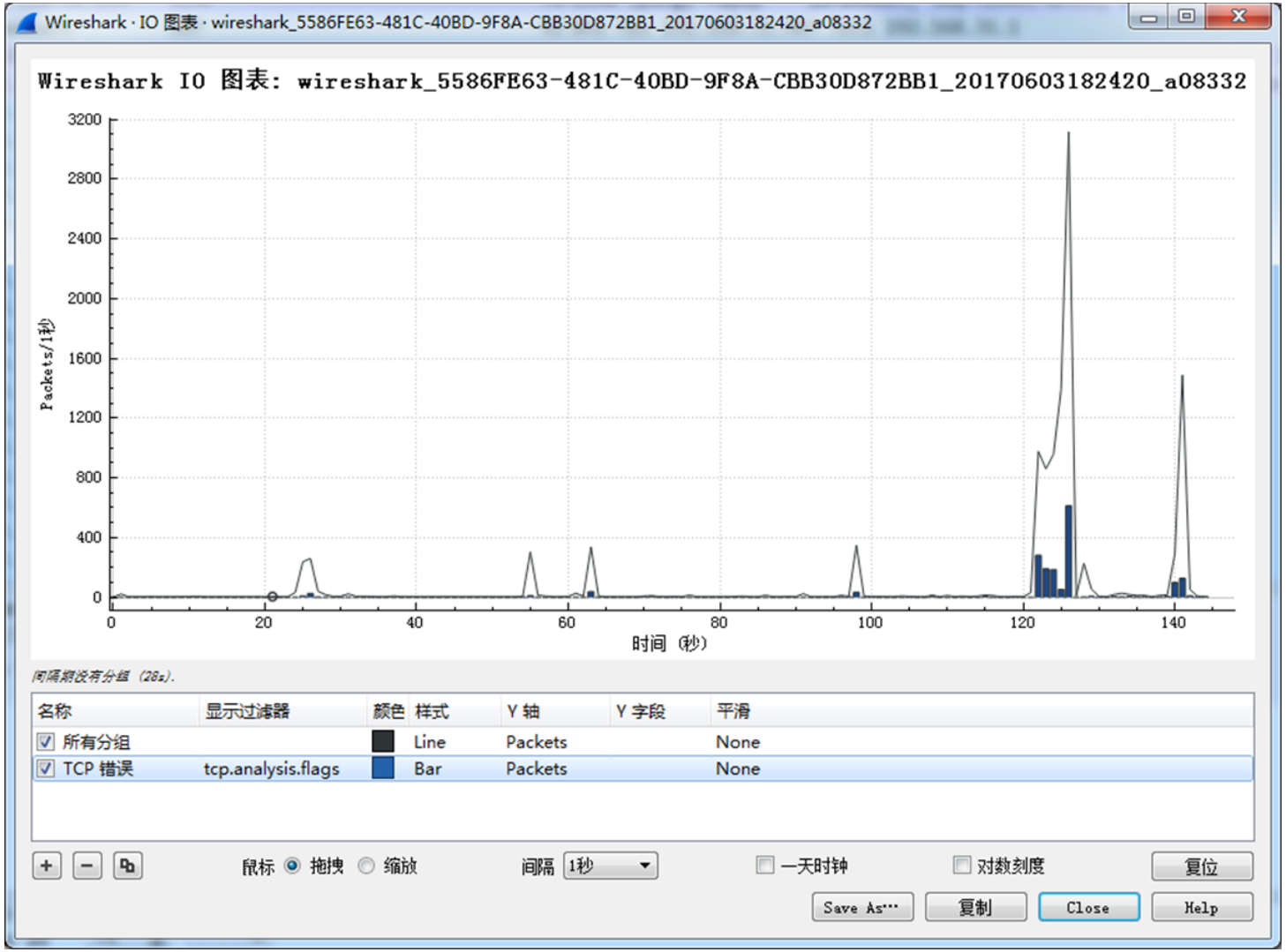
tcp contains "http"	显示payload中包含"http"字符串的tcp封包。

http.request.uri contains "online"	显示请求的uri包含"online"的http封包。

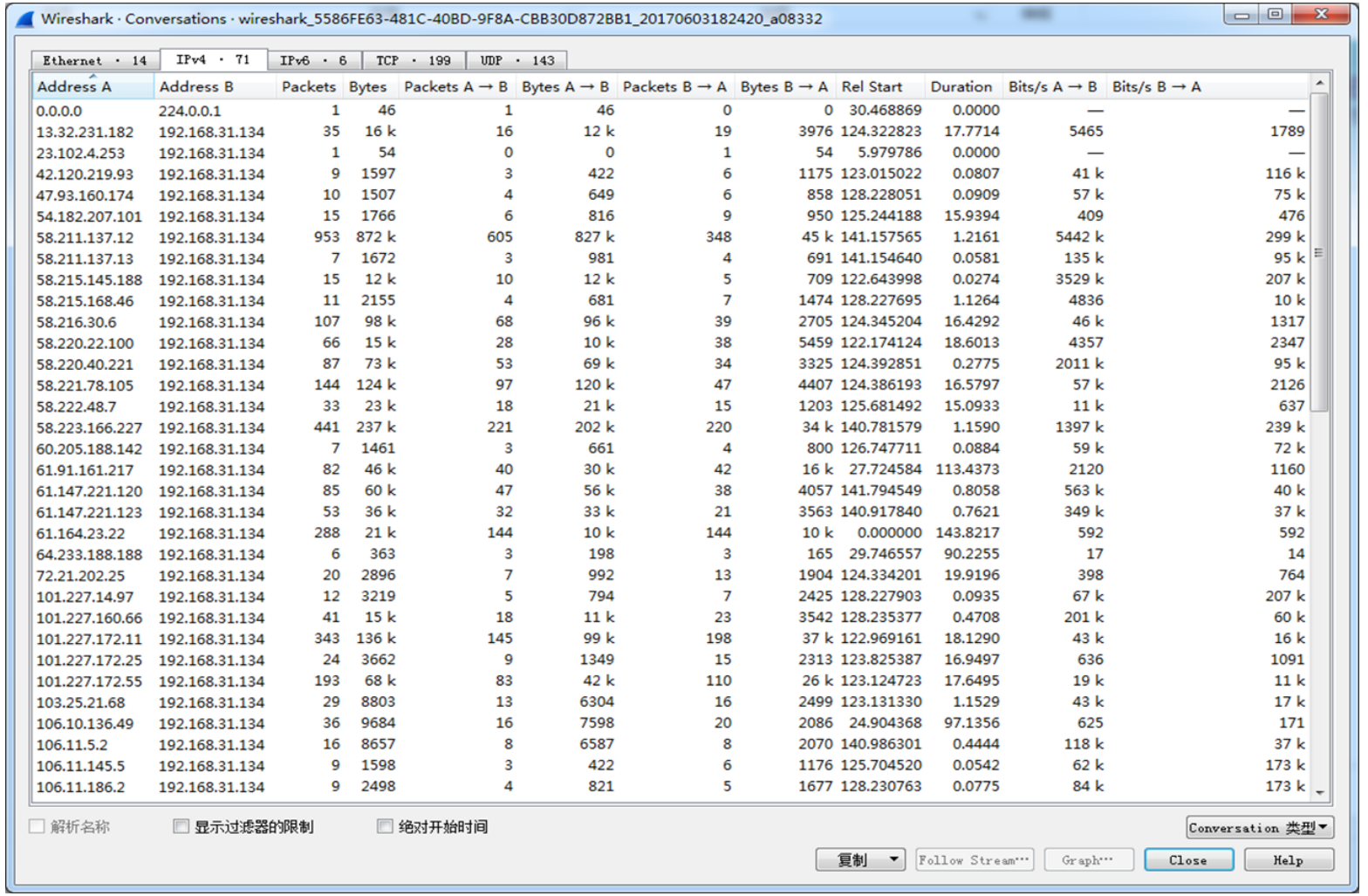
Matches：判断一个协议或者字符串匹配一个给定的Perl表达式。允许一个过滤器使用与Perl兼容的正则表达式（PCRE），但是只能应用于协议或者字符串类型的协议字段

其他

IO图形工具

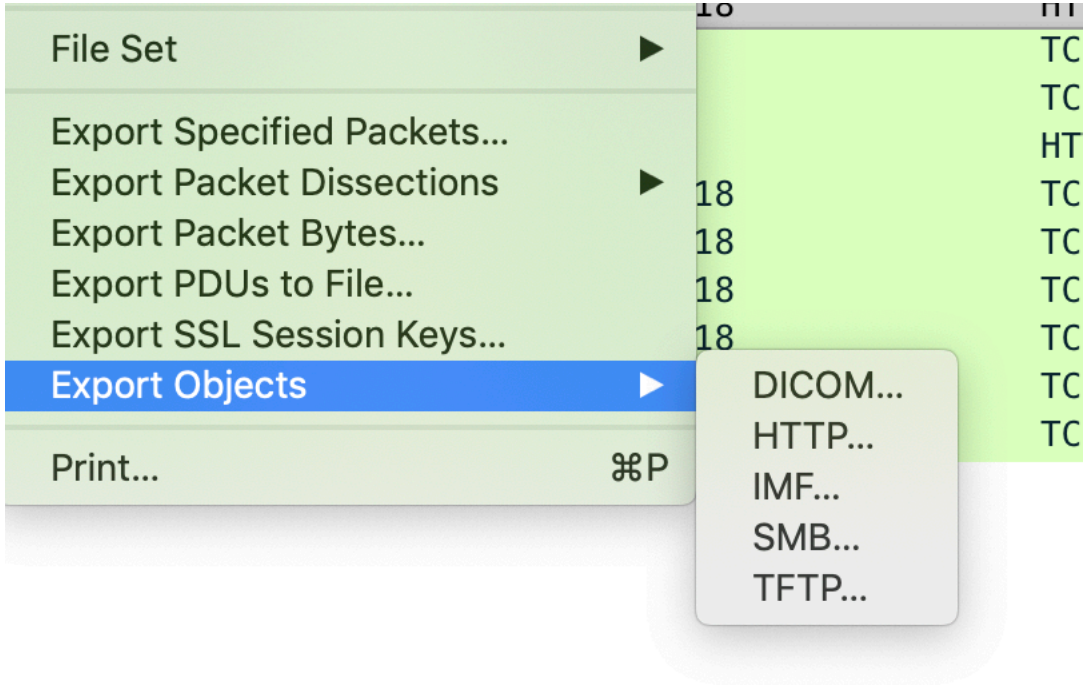


Statistics统计



Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
0.0.0.0	224.0.0.1	1	46	1	46	0	0	30.468869	0.0000	—	—
13.32.231.182	192.168.31.134	35	16 k	16	12 k	19	3976	124.322823	17.7714	5465	1789
23.102.4.253	192.168.31.134	1	54	0	0	1	54	5.979786	0.0000	—	—
42.120.219.93	192.168.31.134	9	1597	3	422	6	1175	123.015022	0.0807	41 k	116 k
47.93.160.174	192.168.31.134	10	1507	4	649	6	858	128.228051	0.0909	57 k	75 k
54.182.207.101	192.168.31.134	15	1766	6	816	9	950	125.244188	15.9394	409	476
58.211.137.12	192.168.31.134	953	872 k	605	827 k	348	45 k	141.157565	1.2161	5442 k	299 k
58.211.137.13	192.168.31.134	7	1672	3	981	4	691	141.154640	0.0581	135 k	95 k
58.215.145.188	192.168.31.134	15	12 k	10	12 k	5	709	122.643998	0.0274	3529 k	207 k
58.215.168.46	192.168.31.134	11	2155	4	681	7	1474	128.227695	1.1264	4836	10 k
58.216.30.6	192.168.31.134	107	98 k	68	96 k	39	2705	124.345204	16.4292	46 k	1317
58.220.22.100	192.168.31.134	66	15 k	28	10 k	38	5459	122.174124	18.6013	4357	2347
58.220.40.221	192.168.31.134	87	73 k	53	69 k	34	3325	124.392851	0.2775	2011 k	95 k
58.221.78.105	192.168.31.134	144	124 k	97	120 k	47	4407	124.386193	16.5797	57 k	2126
58.222.48.7	192.168.31.134	33	23 k	18	21 k	15	1203	125.681492	15.0933	11 k	637
58.223.166.227	192.168.31.134	441	237 k	221	202 k	220	34 k	140.781579	1.1590	1397 k	239 k
60.205.188.142	192.168.31.134	7	1461	3	661	4	800	126.747711	0.0884	59 k	72 k
61.91.161.217	192.168.31.134	82	46 k	40	30 k	42	16 k	27.724584	113.4373	2120	1160
61.147.221.120	192.168.31.134	85	60 k	47	56 k	38	4057	141.794549	0.8058	563 k	40 k
61.147.221.123	192.168.31.134	53	36 k	32	33 k	21	3563	140.917840	0.7621	349 k	37 k
61.164.23.22	192.168.31.134	288	21 k	144	10 k	144	10 k	0.000000	143.8217	592	592
64.233.188.188	192.168.31.134	6	363	3	198	3	165	29.746557	90.2255	17	14
72.21.202.25	192.168.31.134	20	2896	7	992	13	1904	124.334201	19.9196	398	764
101.227.14.97	192.168.31.134	12	3219	5	794	7	2425	128.227903	0.0935	67 k	207 k
101.227.160.66	192.168.31.134	41	15 k	18	11 k	23	3542	128.235377	0.4708	201 k	60 k
101.227.172.11	192.168.31.134	343	136 k	145	99 k	198	37 k	122.969161	18.1290	43 k	16 k
101.227.172.25	192.168.31.134	24	3662	9	1349	15	2313	123.825387	16.9497	636	1091
101.227.172.55	192.168.31.134	193	68 k	83	42 k	110	26 k	123.124723	17.6495	19 k	11 k
103.25.21.68	192.168.31.134	29	8803	13	6304	16	2499	123.131330	1.1529	43 k	17 k
106.10.136.49	192.168.31.134	36	9684	16	7598	20	2086	24.904368	97.1356	625	171
106.11.5.2	192.168.31.134	16	8657	8	6587	8	2070	140.986301	0.4444	118 k	37 k
106.11.145.5	192.168.31.134	9	1598	3	422	6	1176	125.704520	0.0542	62 k	173 k
106.11.186.2	192.168.31.134	9	2498	4	821	5	1677	128.230763	0.0775	84 k	173 k

文件提取


File Set
Export Specified Packets...
Export Packet Dissections
Export Packet Bytes...
Export PDUs to File...
Export SSL Session Keys...
Export Objects
Print...
⌘P
DICOM...
HTTP...
IMF...
SMB...
TFTP...

Example

http://106.12.37.37/misc1.tar

Tshark的使用

官方文档
https://www.wireshark.org/docs/man-pages/tshark.html

常用参数：
-r: -r <infile> 设置读取本地文件
-R: -R <read filter>,包的读取过滤器，可以在**wireshark**的**filter**语法上查看；在**wireshark**的视图**->**过滤器视图，在这一栏点击表达式，就会列出来对所有协议的支持。
-Y: -Y <display filter>,使用读取过滤器的语法，在单次分析中可以代替**-R**选项;

-T: -T pdml|ps|text|fields|psml,设置解码结果输出的格式，包括**text,ps,psml**和**pdml**，默认为**text**
-e: 如果**-T fields**选项指定，**-e**用来指定输出哪些字段;

键盘流量

USB协议的数据部分在Leftover Capture Data域之中

10	1.574247	host	1.10.1	USI
11	1.582183	1.10.1	host	USI
12	1.582235	host	1.10.1	USI
13	1.590187	1.10.1	host	USI
14	1.590238	host	1.10.1	USI
15	1.598183	1.10.1	host	USI
16	1.598234	host	1.10.1	USI
17	1.606184	1.10.1	host	USI
18	1.606236	host	1.10.1	USI
19	1.612183	1.10.1	host	USI
20	1.612235	host	1.10.1	USI
21	1.620184	1.10.1	host	USI
22	1.620236	host	1.10.1	USI
23	1.628182	1.10.1	host	USI

```
▶ Frame 13: 72 bytes on wire (576 bits), 72 bytes captured (576 bits)
▶ USB URB
  Leftover Capture Data: 0000010000000000
```

数据长度为八个字节

USB流量分为键盘流量和鼠标流量。
键盘数据包的数据长度为8个字节，击键信息集中在第3个字节

映射关系
https://usb.org/sites/default/files/documents/hut1_12v2.pdf

Table 12: Keyboard/Keypad Page

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
0	00	Reserved (no event indicated)[9]	N/A	√	√	√	4/101/104
1	01	Keyboard ErrorRollOver[9]	N/A	√	√	√	4/101/104
2	02	Keyboard POSTFail[9]	N/A	√	√	√	4/101/104
3	03	Keyboard ErrorUndefined[9]	N/A	√	√	√	4/101/104
4	04	Keyboard a and A[4]	31	√	√	√	4/101/104
5	05	Keyboard b and B	50	√	√	√	4/101/104
6	06	Keyboard c and C[4]	48	√	√	√	4/101/104
7	07	Keyboard d and D	33	√	√	√	4/101/104
8	08	Keyboard e and E	19	√	√	√	4/101/104
9	09	Keyboard f and F	34	√	√	√	4/101/104
10	0A	Keyboard g and G	35	√	√	√	4/101/104
11	0B	Keyboard h and H	36	√	√	√	4/101/104
12	0C	Keyboard i and I	24	√	√	√	4/101/104
13	0D	Keyboard j and J	37	√	√	√	4/101/104
14	0E	Keyboard k and K	38	√	√	√	4/101/104
15	0F	Keyboard l and L	39	√	√	√	4/101/104
16	10	Keyboard m and M[4]	52	√	√	√	4/101/104
17	11	Keyboard n and N	51	√	√	√	4/101/104
18	12	Keyboard o and O[4]	25	√	√	√	4/101/104
19	13	Keyboard p and P[4]	26	√	√	√	4/101/104
20	14	Keyboard q and Q[4]	17	√	√	√	4/101/104

可以用tshark命令可以将 leftover capture data进行提取

tshark -r 流量包 -T fields -e usb.capdata > usbdata.txt

提取出来后根据映射关系还原即可

```
mappings = {
  0x04:"A",0x05:"B",0x06:"C",0x07:"D",0x08:"E",0x09:"F",0x0A:"G",0x0B:"H",0x0C:"I",
  0x0D:"J",0x0E:"K",0x0F:"L",0x10:"M",0x11:"N",0x12:"O",0x13:"P",0x14:"Q",0x15:"R",
  0x16:"S",0x17:"T",0x18:"U",0x19:"V",0x1A:"W",0x1B:"X",0x1C:"Y",0x1D:"Z",0x1E:"1",
  0x1F:"2",0x20:"3",0x21:"4",0x22:"5",0x23:"6",0x24:"7",0x25:"8",0x26:"9",0x27:"0",
  0x28:"\n",0x2a:"[DEL]",0X2B:"    ",0x2C:" ",0x2D:"-",0x2E:"=",0x2F:"[",0x30:"]",
  0x31:"\\",0x32:"~",0x33:";",0x34:"'",0x36:",",0x37:"."
}
```

Thanks!

谷安天下

国内领先的信息安全与IT风险管理服务提供商

T +86 01 51626887
F +86 01 51626887-816
E market@gooann.com

A-806.Digital Tower,No.2,South Street ZhongGuanCun,
Haidian District,Beijing

北京海淀区中关村南2大街号
数码大厦A座806

安全牛课堂：**www.aqniukt.com**