# 山东大学计算机网络实验——Protocol Layers



 PancrasBohemian
 于 2017-11-04 20:10:25 发布
 6843
 收藏 67

 分类专栏:
 <u>助教笔记</u> 文章标签:
 wireshark

 版权声明:
 本文为博主原创文章,遵循 <u>CC 4.0 BY-SA</u>版权协议,转载请附上原文出处链接和本声明。

 本文链接:
 <u>https://blog.csdn.net/PancrasBohemian/article/details/78445635</u>

 版权

	1				85.
	1			8	
1		1	-67	5	
			<u>.</u>		
۰.		c.		6.	
	1		-	20	ú.,

<u>助教笔记 专栏收录该内容</u>

5 篇文章 1 订阅 订阅专栏

# 山东大学计算机网络实验一 Protocol Layers

发现实验手册上的WireShark版本还停留在1.8、1.9,所以写一篇在Mac上进行实验的教程,希望能够帮到其他同学

### Step1

使用WireShark 抓包分析。 首先打开WireShark,看到这个界面

			<b>_</b> W	ireshark 🕅	网络分析	<u>98</u>							
	۲		6	۹ 🧼				垫			÷	Θ	»
■ 应用显示过滤器	器 < 米/>										表达式		+
		_											
欢迎	使用 Wiresh	hark											
捕获													
	_												
…使用〕	这个过滤器: 📙	输入捕获过	滤器				•	All in	terface	es shown	•		
											_		
N T	/i-Fi: en0	idao, bridao(		M	<i>۱</i> ــــــ	~	_^_						
n'	2n0	lage: briaget											
a	wdl0												
ut	tun0												
т	hunderbolt 1:	en1											
ut	tun1												
т	hunderbolt 2:	en2											
ut	tun2			-									
学习													
用户指	导 · Wiki ·	问题与解答	·邮件列表	Ę									
正在运	行 Wireshark2	.4.2 (v2.4.2-	0-gb6c63a	ae).									

你要确定你自己使用的网络是哪一个,因为Mac都没有网线插口,所以我们用网线的话都需要一个Adapter(适配器)。 我们打开系统偏好设置->网络

	网络	Q 搜索
位置:	自动	<b>C</b>
● Apple UAdapter 〈···〉 已连接 ● Wi-Fi 已连接	状态:	<b>已连接</b> "Apple USB Ethernet Adapter"当前是活跃的, 其 IP 地址为 211.87.226.144。
● 蓝牙 PAN <sub>未连接</sub>	配置 IPv4:	使用 DHCP 🗘
USB 10/00 LAN	IP 地址:	211.87.226.144
未连接	子网掩码:	255.255.255.192
● IPhone USB 未连接	路由器:	211.87.226.129
● Thunderbolt 网桥	DNS 服务器:	202.194.15.12、8.8.8.8、211.87.22
	搜索域:	qlsc.sdu.edu.cn
	IPv6 地址:	2001:da8:7001:30:63:ad58:fabb:1373
+ - *		高级 ?
		向导 复原 应用 http://blog.csdn.net/PancrasBohemian

我用的适配器是Apple官方的,所以名字叫Apple USB Ethernet Adapter,我们在WireShark中选择这个,你们在用的时候选择正确的即可。

捕获	
…使用这个过滤器: 📕 输入捕获过滤器	✓ All interfaces shown ▼
Thunderbolt 1: en1 utun1 Thunderbolt 2: en2 utun2	
Apple USB Ethernet Adapter: en5	Muma
Loopback: Io0 gif0 stf0 © Cisco remote capture: cisco	http://blog.csdp.pet/PapcrasBohemi

● ● ● ●																												
				0					×	(	5	C	2					$\geq$	1	Ś	Ł				Ð	Θ	2	>>
回应用	用显え	示过滤	器.	<9	€/>																		C	•	表达	式	+	•
No.		Time	e			S	ourc	е					De	stina	ation					Prot	locol	Le	ength	Info				
	1	0.0	0000	00		H	lang	zho	u_8e	:27	:00	)	Br	oad	dcas	st				AR	2		60	Who	has	211.	87	
	2	0.1	1029	84		H	lang	zho	u_8e	:27	:00	)	Br	oad	dcas	st				AR	2		60	Who	has	211.	87	Ш
	3	0.1	1818	24		2	11.	87.	226.	144	ŀ		20	2.1	194.	15.	12			DNS	5		87	Sta	ndar	d que	ry	U
	4	0.1	1827	55		2	02.	194	.15.	12			21	1.8	37.2	226.	14	4		DNS	5		152	Sta	ndar	d que	ry	
	5	0.2	2025	92		Н	lang	zho	u_8e	:27	:00	)	Br	oad	dcas	st				AR	2		60	Who	has	211.	87	
	6	0.3	8014	96		H	lang	zho	u_8e	:27	:00	)	Br	oad	dcas	st				AR	2		60	Who	has	211.	87	
	7	0.4	1006	62		Н	lang	zho	u_8e	:27	:00	)	Br	oad	dcas	st				AR	2		60	Who	has	211.	87	
	8	0.F	6018	50		Н	lano	zho	11 86	: 27	:00	1	Rr	nar	Icas	:+				ARI	2		60	Who	has	211.	87	_
► Fr ► Et ► Ad	ame heri dre:	1: net ss R	60 II, eso	Srolut:	es c: ion	on Han Pr	gzh oto	e (4 ou_8 col	180 3e:2 (re	gue	s), 0 ( st)	00:	Øf:	e2:	se:	27:	00)	a (4 ), I	480 Dst:	Bro	adca	n in ast	(ff	face	0 ff:ff	:ff:	ff)	
0000 0010 0020 0030	08 00 00	00	06 00 00	04 00 00	00 00 00	01 00 00	00 00 d3 00	0f 57 00	e2 e2 e2 00	8e 86 00	27 27 00 00	00 00 00	d3 00	00 57 00	e2 00	81 00				W .		W						
0	7	Apple	USE	8 Eth	erne	t Ad	apte	r: ent	5: <liv< td=""><td>e ca</td><td>pture</td><td>e in p</td><td>orogi</td><td>ess</td><td>&gt;</td><td></td><td>111</td><td>01</td><td>分组</td><td>: 26 ·</td><td>已显示</td><td>R: 26</td><td>(100.</td><td>0%)</td><td>配置文</td><td>件: Det</td><td>auit</td><td></td></liv<>	e ca	pture	e in p	orogi	ess	>		111	01	分组	: 26 ·	已显示	R: 26	(100.	0%)	配置文	件: Det	auit	

随后我们在上方的输入框中输入tcp.port == 80 不是手册上写的 tcp port 80! !!

_													
	tcp.po	rt == 80								X		表达式	+
No		Time		Source			Destinatio	n	Proto	col Length	Info		
Г	1	7 2.480	439	121.19	5.187.6	)	211.87.	226.144	TCP	66	80 →	50282	[FI
	18	3 2.480	503	211.87	.226.144	1	121.195	.187.60	TCP	66	5028	2 → 80	[AC
	19	2.480	592	211.87	.226.144	1	121.195	.187.60	TCP	66	5028	2 → 80	[FI
L	20	2.488	859	121.19	5.187.60	9	211.87.	226.144	TCP	66	80 →	50282	[AC
	10	3 19.19	1890	211.87	.226.144	1	121.195	.187.60	TCP	78	5028	3 → 80	[SY
	104	1 19.20	0714	121.19	5.187.60	)	211.87.	226.144	TCP	74	80 →	50283	[SY
	10	5 19.20	0791	211.87	.226.144	1	121.195	.187.60	TCP	66	5028	3 → 80	[AC
	10	5 19.20	1281	211.87	. 776. 144	1	121.19	. 187.60	тср	347	5028	R → 80	[PS
►	Frame	2 17: 6	6 bytes	s on wire	(528 bi	ts), 6	6 bytes	captured	(528 bits	) on inte	rface	0	
►	Ether	net II	, Src:	Hangzhou_	_8e:27:0	0 (00:	0f:e2:8e	:27:00),	Dst: Appl	e_ec:ff:5	(04:	69:f8:	ec:ff:
►	Inter	net Pr	otocol	Version 4	4, Src:	121.19	95.187.60	(121.19	5.187.60),	Dst: 211	87.22	6.144	(211.8
►	Trans	missio	n Conti	rol Proto	col, Sro	Port:	80, Dst	Port: 5	0282, Seq:	1, Ack: :	l, Len	: 0	miani

看到实验手册上要求check "enable network name resolution". 其实没那么玄乎,首先停止获取,然后使用cmd+K或者在上方菜单栏中选择捕获->选项

0 •		⊿ Wireshark · 捕获接口				
		输入 输出 选项				
接口	流量	链路层头	混杂	Snap长度 (E	缓存 (MB)	监控模
Thunderbolt 1: en1	-	Ethernet		默认	2	-
▶ utun1	-	BSD loopback		默认	2	-

	-	Ethernet	$\checkmark$	默认	2	
▶ utun2	-	BSD loopback	$\checkmark$	默认	2	
<ul> <li>Apple USB Ethernet Adapter: en5</li> </ul>		Ethernet	<b>V</b>	默认	2	—
Loopback: Io0		BSD loopback	$\checkmark$	默认	2	-
gif0	-	BSD loopback	$\checkmark$	默认	2	_
stf0		BSD loopback	$\checkmark$	默认	2	
Cisco remote capture: cisco	-	Remote capture dependent DLT	-	_	_	_
Random packet generator: randpkt	-	Generator dependent DLT	-	_	-	_
2 在所有接口上使用混杂模式					管理	里接口…
2 在所有接口上使用混杂模式 f选择接口的捕获过滤器: 输入捕获过滤器	전 전 1			•	管理	里接口 译BPFs

#### 在这里确保你选择了正确的接口之后,点上方的选项

	✓ Wireshark · 捕获接口
	输入 输出 选项
	<ul> <li>显示选项 解析名称</li> <li>✓ 实时更新分组列表 ✓ MAC地址解析</li> <li>✓ 实时捕获时自动滚屏</li> <li>✓ 显示额外的捕获信息对话框</li> <li>✓ 解析传输层名称</li> </ul>
	自动停止捕获,在经过
	1 \$ 文件
	□ 1
Help	http://blog.csdn.net/Pan <mark>Close</mark> Bo <mark>、开始,</mark>

发现已经选好了……费了好多劲找到,那我们那就点击start开始吧。 开始之后我们继续按照手册上来,在命令行中输入curl http://www.baidu.com 然后我们发现了海量的包,哪些是我们需要的呢?就是那些绿色底的包

blo		Time	Courses	Destination	Drotocol	Length	lafa	_
THU	2272	115,085000	211.87.226.144	www.a.shifen.com	TCP	78	50450 + 80 [SYN] Sec.e0 Win=65535 Lec.e0 MSS=1460 WS=32 TSval=364488978 TSecre0	
Γ	2273	116 007321	way a shifen com	211 87 226 144	TCP	78	88 - 50458 [SVN ACK] Seq Ack=1 Win=8192 Len=0 MSS=1452 WS=32 SACK PERM=1	
	2274	116 007400	211 97 226 144	Last a chifon com	TCP	5.4	50450 - 90 [Sitk] Son 1 Act 1 Win 261144 Lone	
-	22/4	110.007409	211.07.220.144	www.d.Shiren.com	TCP	34	3030 - 30 [ACK] 364-1 ACK-1 W11-202144 LEII-0	-
+	2275	116.007500	211.87.226.144	www.a.shifen.com	HTTP	131	GET / HTTP/1.1	6
	2276	116.029689	www.a.shifen.com	211.87.226.144	TCP	60	0 80 → 50450 [ACK] Seq=1 Ack=78 Win=24704 Len=0	2
	2277	116.031877	www.a.shifen.com	211.87.226.144	TCP	454	80 → 50450 [PSH, ACK] Seq=1 Ack=78 Win=24704 Len=400 [TCP segment of a reassem	8
	2278	116.031904	www.a.shifen.com	211.87.226.144	TCP	1094	80 → 50450 [PSH, ACK] Seq=401 Ack=78 Win=24704 Len=1040 [TCP segment of a reas	8
	2279	116.031991	211.87.226.144	www.a.shifen.com	TCP	54	50450 → 80 [ACK] Seq=78 Ack=401 Win=261728 Len=0	8.
	2280	116.031994	211.87.226.144	www.a.shifen.com	TCP	54	50450 → 80 [ACK] Seg=78 Ack=1441 Win=260704 Len=0	8
-	2281	116,033733	www.a.shifen.com	211.87.226.144	HTTP	1395	HTTP/1.1 200 0K (text/html)	<b>a</b> .,
	2282	116,033735	www.a.shifen.com	211.87.226.144	TCP	60	80 → 50450 [FIN_ACK] Sec=2782 Ack=78 Win=24704 Len=0	-0
	2202	*** ******	344 03 336 414		100			-
►	Frame	2275: 131 byt	es on wire (1048 bits	), 131 bytes captured	(1048 b	its) on	n interface 0	
►	Ether	net II, Src: A	pple_ec:ff:5c (04:69:	f8:ec:ff:5c), Dst: Ha	ngzhou_8	e:27:00	0 (00:0f:e2:8e:27:00)	
►	Inter	net Protocol V	ersion 4, Src: 211.87	.226.144 (211.87.226.	144), Ds	t: www.	.a.shifen.com (119.75.216.20)	
•	Trans	mission Contro	l Protocol, Src Port:	50450, Dst Port: 80,	Seq: 1,	Ack: 1	1, Len: 77	
v	Hyper	text Transfer	Protocol					
	► GET	/ HTTP/1.1\r	\n					
	Hos	t: www.baidu.	com\r\n					
	Use	er-Agent: curl	/7.52.1\r\n					- 1
	ACC	ept: */*\r\n						- 1
	111	n					http://blog_csdn_net/PancrasBohemia	
	[Fi	ill request UR	I: http://www.baidu.co	[/mo			http://brog.countree/renerabonemia	1

## Step 2

这一步主要是根据上一步来,通过对不同包(这个定义不准确,一组信息在链路层被称作帧,在网络层被称作包,在传输层称作 段,在应用层称作消息)的拆分等操作,了解里面的信息。 首先找到一个协议为HTTP的包

[SYN] Seq=0 Win=65535 Len=0 MSS=1466
[CVN ACK] Sog-0 Ack-1 Win-0102 Lon-
[SIN, ACK] SEY-0 ACK-1 WIN-0192 LEN-
[ACK] Seq=1 Ack=1 Win=262144 Len=0
2/1.1
[ACK] Seq=1 Ack=78 Win=24704 Len=0
[PSH, ACK] Seq=1 Ack=78 Win=24704 Le
[PSH, ACK] Seq=401 Ack=78 Win=24704
[ACK] Seq=78 Ack=401 Win=261728 Len=
[ACK] Seq=78 Ack=1441 Win=260704 Ler
200 OK (text/html)
50 50 50 50 50 50 50 50 50 50 50 50 50 5

这个包有着GET方法,就是我们要第一步找到的包,也就是No.2275包,这个包是从我们的电脑发送到服务器上的。

▶ Frame 2275: 131 bytes on wire (1048 bits), 131 bytes captured (1048 bits) on interface 0

Ethernet II, Src: Apple\_ec:ff:5c (04:69:f8:ec:ff:5c), Dst: Hangzhou\_8e:27:00 (00:0f:e2:8e:27:00)

Internet Protocol Version 4, Src: 211.87.226.144 (211.87.226.144), Dst: www.a.shifen.com (119.75.216.20)

- Transmission Control Protocol, Src Port: 50450, Dst Port: 80, Seq: 1, Ack: 1, Len: 77
- Hypertext Transfer Protocol

我们从应用中间的那一层可以看到这个包的一些细节。 第一部分是Frame,这不是一个协议,只是一个包的概括性信息的集合。 第二部分是Ethernet,这个就开始和下图匹配了



# Figure 4: Protocol stack for a web fetch

http://blog.csdn.net/PancrasBohemian

随后就是经典的IP、TCP、HTTP三个协议,我们注意到,这个顺序是从栈底到栈顶依次上升的,这是因为包是从顶到底依次传递的,低层次协议的头部信息被加到高层次协议信息的前头。具体可以见课本(中文版P25)。

这一步结束之后,我们找到另一个HTTP包

也就是2281号包,我们注意到在No这一行,两个HTTP包都带着一个箭头,指明方向,我们可以明白,这个包就是从服务器到我们的电脑的。与上一个HTTP包相比,这个包里面包含着"200 OK"的信息

# Hypertext Transfer Protocol ► HTTP/1.1 200 OK\r\n

这个信息说明我们的获取是成功的。 我们同时发现,这个面板中多出来了两个块,也就是下图中没有标蓝的两块

- [3 Reassembled TCP Segments (2781 bytes): #2277(400), #2278(1040), #2281(1341)]
- Hypertext Transfer Protocol
- Line-based text data: text/html

上面那一块描述了不仅仅包含这个包的信息。在最可能的情况下,网络请求被以一系列包的方式通过网络传输,并最终在电脑上 被组装成信息。标有HTTP的包是网络请求中的最后一个包,并且能列出可以获得完整网络请求的所有的包。 下面那一块描述了网页获取的内容。在本例中为text/html

#### **Step3 Packet Structure**

要求画一下GET包的结构,我们可以看到

```
Ethernet II, Src: Apple_ec:ff:5c (04:69:f8:ec:ff:5c), Dst: Hangzhou_8e:27:00 (00:0f:e2:8e:27:00)
```

- ▶ Internet Protocol Version 4, Src: 211.87.226.144 (211.87.226.144), Dst: www.a.shifen.com (119.75.216.20)
- ▶ Transmission Control Protocol, Src Port: 50450, Dst Port: 80, Seq: 1, Ack: 1, Len: 77

Hypertext Transfer Protocol

0000	00	Øf	e2	8e	27	00	04	69	f8	ec	ff	5c	08	00	45	00	'i\E.
0010	00	75	fb	28	40	00	40	06	3a	12	d3	57	e2	90	77	4b	.u.(@.@. :WwK
0020	d8	14	c5	12	00	50	00	31	a0	43	b5	77	34	e2	50	18	P.1 .C.w4.P.
0030	20	00	06	15	00	00	47	45	54	20	2f	20	48	54	54	50	GE T / HTTP
0040	2f	31	2e	31	Ød	0a	48	6f	73	74	3a	20	77	77	77	2e	/1.1Ho st: www.
0050	62	61	69	64	75	2e	63	6f	6d	Ød	0a	55	73	65	72	2d	baidu.co mUser-
0060	41	67	65	6e	74	3a	20	63	75	72	6c	2f	37	2e	35	32	Agent: c url/7.52
0070	2e	31	Ød	0a	41	63	63	65	70	74	3a	20	2a	2f	2a	Ød	.1Acce pt: */*.
0080	0a	Ød	0a														
0070 0080	2e Øa	31 Ød	0d 0a	0a	41	63	63	65	70	74	3a	20	2a	2f	2a	Ød	.1Acce pt: */*.

http://blog.csdn.net/PancrasBohemian

随着你鼠标滑过中间那些不同的协议,下面会有不同的内容被高亮,依据这个,我们可以画出图来。 而大小可以在程序的最下面找到

画出Ethernet的头部和有效载荷(payload,指除去协议首部之外实际传输的数据)



## Step4 Protocol Overhead

Protocol Overhead指的是协议开销,由于协议本身的报头等内容也需要占用一定的空间,用来标识该种协议、报文内各个字段的含义等信息,这种内容就是协议开销了。

下载的包从一个Info中带有SYN,ACK信号的包开始,到下面第一个遇到的HTTP包后面的TCP包为止。

	2121	87.783417	www.a.shifen.com	211.87.226.144	TCP	60	- 80° → 50448 [ACK] Seq=1 Ack=78 Win=247
/	2120	87.757777	211.87.226.144	www.a.shifen.com	HTTP	131	GET / HTTP/1.1
	2119	87.757649	211.87.226.144	www.a.shifen.com	TCP	54	50448 → 80 [ACK] Seq=1 Ack=1 Win=2621
	2118	87.757569	www.a.shifen.com	211.87.226.144	TCP	78	80 → 50448 [SYN, ACK, ECN, CWR] Seq=0

其中协议开销一共有78+54+54+60=246字节

HTTP有效信息一共有77字节,占23.84%可以说协议开销占比较大。也较为重要,因为标识各个字段的含义等信息是必不可少的。不过要是开销能小一些会更好

### **Step5 Demultiplexing Keys**

分别对应0x08 00(16进制) 0x6 (十六进制)

Which Ethernet header field is the demultiplexing key that tells it the next higher layer is IP? What value is used in this field to indicate "IP"?

上面红色框,对应的是0x08 00

Which IP header field is the demultiplexing key that tells it the next higher layer is TCP? What value is used in this field to indicate "TCP"?

下面红色框对应的是0x6



<u>创作打卡挑战赛</u> 赢取流量/现金/CSDN周边激励大奖