Wireshark例题-CTF

原创

 H3rmesk1t

 于 2021-05-14 23:40:53 发布

 1173 か 收藏 6

 分类专栏:

 <u>安全学习</u>文章标签: wireshark ctf 网络安全

 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:

 https://blog.csdn.net/LYJ20010728/article/details/116804672

版权

C

安全学习 专栏收录该内容

21 篇文章 4 订阅 订阅专栏

Wireshark例题-CTF

搜索
文件提取
例题一
例题二
信息提取

搜索

题目文件: key.pcapng

题目描述: flag被盗, 赶紧溯源!

题目题解:

①可以只将这个数据包当做文本文件打开,比如用一些notepad++编辑器,然后直接搜索 ②用Wireshark自带的搜索功能找尝试查找一些关键词(比如key、flag、shell、pass等),然后跟进可疑的数据包,根据数据包特征,很明 显看出这是一个菜刀连接一句话木马的数据包,然后往下找,即可看到读取的flag

	147	Content-Length: 49			_	
1	148	Content-Type: text/html; char	查找			
1	149		杏状 苯格 立件本代 持过			_
1	150	X@Yflag{This_is_a_fl0g}	单场 省操 文件单扬 称吃			
1	151	[S]	李华日标(5), 角。			
1	152	/var/www/html		*	重扱下一下	
1	153	[E]			计称(1)	
1	154	X@YNULFSSOHNULNULACKNULNUI			PI \$2(1)	
		NULNUL (@w@NUL@ACK晶括xE4SOF			大当箭立件市委带	
		NUL 801 XFC * NUL NUL NUL * NUL NUL			1111月又日中亘找	
		NULNULNULNULNUL括xE4 STX NUL	□反向查找			
		xFFxFFxFFxFFNULPVxC0NULBS			查找所有打开文件(O)	
		NULNULACKINULNULNULX8CISOHINI				
		@NUL@ACK疎括xE4SOH括鋰烷NUL			取消	
1	155	X-Forwarded-For: 44.146.238.1	☑循环查找®			
1	156	Referer: http://192.168.228.1	李诒博士			
1	157	Content-Type: application/x-w		<u> </u>		
1	158	User-Agent: Mozilla/5.0 (comp	. ● 普通(12)		④失去焦点后	
1	159	Host: 192.168.228.135	◯扩展⊠ (\n, \r, \t, \0, \x)		○始终	
1	160	Content-Length: 785	○正则表达式(G) □.匹配新行			
1	161	Cache-Control: no-cache				
1	162				csdn:net/EYJ20010728	
1.1	1.00	COCCOMMENT AND A COMMENT AND A DATE AND A DESCRIPTION OF A				

	分组字节流 ~	宽窄	🗸 🗌 区分大小型	字符串	∨ flag				查找		取消
¥.~	Timo	Samea	Πe	-tination	Prot	and I andth	Trfo				
>	Frame 105: 251 b	oytes on wire	(2008 bits), 2	51 bytes ca	ptured (2008	3 bits) on	interface	\Device\NP	F_{02EC3	SCA0-I	E377-4309
>	Ethernet II, Sro	: VMware_35:f	9:e5 (00:0c:29	:35:f9:e5),	Dst: VMware	e_c0:00:08	(00:50:56	:c0:00:08)			
>	Internet Protoco	ol Version 4,	Src: 192.168.2	28.135, Dst	: 192.168.22	28.1					
>	Transmission Con	ntrol Protocol	, Src Port: 80	, Dst Port:	52713, Seq:	248, Ack:	2201, Le	n: 197			
>	Hypertext Transf	Fer Protocol									
~	Line-based text	data: text/ht	ml (5 lines)								
	X@Yflag{This_	is_a_f10g}\n									
	[S]\n										
	/var/www/html	\n									
	[E]\n										
	X@Y										

文件提取

例题一

题目文件: caidao.pcapng

题目描述: 有人偷偷下载了文件!

题目题解:

根据题意可能数据包中存在文件传输,尝试直接导出,选择File(文件)->Export Objexts(导出对象),然后可以看到一些协议,比如选 中http就可以看到通过http传输的一些文件,在右下角有导出按钮,可生生成相应的文件。但是本题中无法用此方法直接看到被下载的文 件,因为有些文件是直接通过tcp或udp协议传输的,http协议只能看到的访问的链接,但不会看到传输的内容(比如你去访问放一个链接 download.php?file=test.rar,通过上述导出对象的方式看不出来下载的文件的内容的)

文件(F) 编辑(E) 视图(V)	跳转(G) 捕获(C) 分析(A)	统计(S) 电	话(Y) 无线(W)	工具(T) 帮	助(H)		
Open	Ctrl+O	🖹 🛉 🛓		ର୍ ବ୍ 🏨				
Open Recent	+							+ 🗸 🛋
合并(M)			Destinat	on	Protocol	Length Info		
从 Hex 转储导入(I)		61	192.16	3.1.145	HTTP	828 POST	/3.php HTTP/1.1	(application/x-w
Close	Ctrl+W	145	10.211	55.61	HTTP	340 HTTP/	1.1 200 OK (tex	t/html)
/尼方(C)	Ctolus	61	192.16	3.1.145	HTTP	766 POST	/3.php HTTP/1.1	(application/x-w
10011(5)	Cui+5	145	10.211	55.61	HTTP	256 HTTP/	'1.1 200 OK (tex	t/html)
另存为(A)	Ctrl+Shift+S	61	192.16	3.1.145	HTTP	826 POST	/3.php HTTP/1.1	(application/x-w
文件集合	•	145	10.211	55.61	HTTP	433 HTTP/	1.1 200 OK (tex	t/html)
导出特定分组								
导出分组解析结果	+							
导出分组字节流(B)	Ctrl+Shift+X							
导出 PDU 到文件								
导出 TLS 会话密钥								
导出对象	•	DICON	1					
+T(D(D)	Chillip D	HTTP						
31C1(P)	Ctri+P	IMF						
Quit	Ctrl+Q	SMB						
		TFTP						
1								

No.	Time	Source	Destination	Protocol I	Length Info
	5 0.184621	10.211.55.61	192.168.1.145	HTTP	828 POST /3.php HTTP/1.1 (application/x-w
-	9 0.576743	192.168.1.145	10.211.55.61	HTTP	340 HTTP/1.1 200 OK (text/html)
	18 21.139025	10.211.55.61	192.168.1.145	HTTP	766 POST /3.php HTTP/1.1 (application/x-w
	20 24.225688	192.168.1.145	10.211.55.61	HTTP	256 HTTP/1.1 200 OK (text/html)
	30 48.763038	10.211.55.61	192.168.1.145	HTTP	826 POST /3.php HTTP/1.1 (application/x-w
	32 49.117671	192.168.1.145	10.211.55.61	HTTP	433 HTTP/1.1 200 OK (text/html)

这个时候就需要找到那个执行下载的数据包,找到数据传输的部分再导出,比如下面这个数据包 大概是一个菜刀下载的过程,在最后一个包可以看到下载的文件,直接右键点击"导出分组字节流",然后保存为.tar.gz文件

No.	Time	Source	Destination	Protocol L	ength	Info
	5 0.184621	10.211.55.61	192.168.1.145	HTTP	828	POST /3.php HTTP/1.1 (application/x-w
-	9 0.576743	192.168.1.145	10.211.55.61	HTTP	340	HTTP/1.1 200 OK (text/html)
	18 21.139025	10.211.55.61	192.168.1.145	HTTP	766	POST /3.php HTTP/1.1 (application/x-w
	20 24.225688	192.168.1.145	10.211.55.61	HTTP	256	HTTP/1.1 200 OK (text/html)
	30 48.763038	10.211.55.61	192.168.1.145	HTTP	826	POST /3.php HTTP/1.1 (application/x-w
	32 49.117671	192.168.1.145	10.211.55.61	HTTP	433	HTTP/1.1 200 OK (text/html)

MSSQLpri/\t2015-06-03 11:53:31\t0\t077\n MysqlUDFpri/\t2015-06-15 07:37:22\t0\t077\n sql/\t2015-06-08 00:45:44\t0\t077\n sqli/\t2015-12-30 08:06:54\t0\t077\n up/\t2015-06-08 00:36:02\t0\t077\n uploads/\t2015-06-08 00:36:11\t0\t077\n wcms/\t2016-01-17 05:56:46\t0\t0777\n webshop5/\t2015-06-08 08:04:59\t0\t077\n XiaoCms_20140710/\t2015-06-03 11:53:31\t0\t077\n xss/\t2016-01-28 06:48:07\t0\t077\n Z/\t2015-06-03 11:53:31\t0\t077\n 1.php\t2016-01-28 08:54:46\t1740\t0666\n 3.php\t2016-06-01 03:36:25\t27\t0666\n flag.tar.gz\t2016-06-27 08:45:38\t203\t0666\n log.txt\t2015-06-03 12:18:46\t1502\t0666\n news.asp\t2014-06-27 03:44:24\t365\t0666\n SaveFile.asp\t2014-06-27 05:45:08\t822\t0666\n

	32 49.117671	192.168.1.145	10.211.55.61	HTTP	433 HTTP/1.1 200 OK	(text/html)
--	--------------	---------------	--------------	------	---------------------	-------------

Frame 32: 433 bytes on wire (3464 bits), 433 bytes captured (3464 bits) on interface \Device\NPF_{32D3637B-25B5-48CD-

- > Ethernet II, Src: Parallel_00:00:18 (00:1c:42:00:00:18), Dst: Parallel_f4:84:6c (00:1c:42:f4:84:6c)
- > Internet Protocol Version 4, Src: 192.168.1.145, Dst: 10.211.55.61
- > Transmission Control Protocol, Src Port: 80, Dst Port: 49368, Seq: 1, Ack: 773, Len: 379

ransier Prococor		
taut data: taut/html /	(A lines)	
Expand Subtrees		
折叠子树		
全部展开		ÎûûDû+û>ûû∨ûûûû=ûµûûûiû[9ûbYûûzûGû/o\023ûnNûGû\021rûû:\r
全部折 器		137NR\030*RmoDRis\\$\\$RRRERERERERERERERERERE
王的川豆		
应用为列	Ctrl+Shift+I	
作为过滤器应用	•	
Prepare as Filter	•	
对话过滤器	•	
用过滤器着色	•	
追踪流	•	
12+000		
复制	•	
显示分组字节	Ctrl+Shift+O	
导出分组字节流(B)	Ctrl+Shift+X	nttps://biog.csdn.net/LYJ20010/28
	the second secon	taut datat taut/(html (/ lines)) Expand Subtrees 折叠子树 全部展开 全部振叠 位用为列 Ctrl+Shift+I 作为过滤器应用 > Prepare as Filter > 对话过滤器 > 月过滤器着色 > 道踪流 > 复制 > 显示分组字节 Ctrl+Shift+O 导出分组字节流(B) Ctrl+Shift+X

本题中最后还要使用16进制编辑器去除开头和结尾的X@Y字符,这个是菜刀的特征符号,不是文件内容

	flag.txt	2	016/6/27 16:10	文本文档	1 KB
	/ flag.txt	记事本			
	文件(F) 编辑	員(E) 格式(O) 查看(V)	帮助(H)		
	key{8769f	e393f2b998fa6a11	1afe2bfcd65e}		
had that					
例题二					

题目题解: 首先用HTTP条件过滤一下; 右键第一个包,追踪流;

	http						\times	-) +	e
No.		Time	Source	Destination	Protocol	Length	Info		^
>	13	0.925023	192.168.1.101	14.17.42.24	HTTP	210	POST /cgi-bin/uploadunite?func=Cr		
-	18	1.108355	14.17.42.24	192.168.1.101	HTTP/J	1147	HTTP/1.1 200 OK , JavaScript Obje		
	Wiresha	。 小、追踪 HTTD 法 (+	cn stream eq. 2), misc fly pca	222					

Accept-Language: zh-CN,zh;q=0.8

Cookie: ssuid=9979081647; ptui_loginuin=81101652; o_cookie=81101652; pgv_pvid=3703132940; newpt=2; ptcz=4d9c0097882e7b9300db96ff8272c602d390465e21f8cc52d4a4719b7a73dfff; pt2gguin=o0081101652; uin=o0081101652; skey=@zT p_skey=SMR2Xte-Sd3SZj-LIjK01P0C1XsLYf6J5WR7DYT1x1s_; pt4_token=VqHcOafacF-cGmaNbpHVyg__; wimrefreshrun=0&; qm_flag=0; sid=81101652&18c4549e039b41d8d5e73949a54d969a,qU01SM1h0ZS1TZDNTWmotTE1qS08xUE9DbFhzTF1mNko1V1I3RF1UbHgxc18.; qm_userna qm_sid=18c4549e039b41d8d5e73949a54d969a,qU01SM1h0ZS1TZDNTWmotTE1qS08xUE9DbFhzTF1mNko1V1I3RF1UbHgxc18.; qm_domain=http: qm_ptsk=81101652&@zTisQEDyk; foxacc=81101652&0; ssl_edition=mail.qq.com; edition=mail.qq.com; username=81101652&811016

{"path":"fly.rar","appid":"","size":525701,"md5":"e023afa4f6579db5becda8fe7861c2d3","sha":"ecccba7aea1d482684374b22e2e

可以看到一些基本信息,首先这是一个POST数据包,发送了一些文件相关信息,包括名称(fly.rar)和大小(525701)等。接下来应该就 是文件实际上传的数据包,将过滤条件改为: http.request.method=="POST";

	http.req	uest.method = "PO	ST″					*
No.		Time	Source	Destination	Protocol	Length	Info	
	13	0.925023	192.168.1.101	14.17.42.24	HTTP	210	POST	/cgi-bin/uploadunite?func=CreateF
	163	1.864990	192.168.1.101	59.37.116.102	HTTP	110	POST	/ftn_handler/0b126a291df43b53f99c
	289	2.068360	192.168.1.101	59.37.116.102	HTTP	610	POST	/ftn_handler/acbfc77208240d03e6af…
+	431	2.232611	192.168.1.101	59.37.116.102	HTTP	918	POST	/ftn_handler/146b038670952f51f18d
	577	2.364839	192.168.1.101	59.37.116.102	HTTP	782	POST	/ftn_handler/f6c7d6eef80795e03206
+	729	3.102710	192.168.1.101	59.37.116.102	HTTP	391	POST	/ftn_handler/1ffd8670a499bfb6e90c
	738	3.394152	192.168.1.101	14.17.42.24	HTTP	499	POST	<pre>/cgi-bin/uploadunite?func=CheckFi</pre>
	767	5.751789	192.168.1.101	183.60.15.162	HTTP	867	POST	/cgi-bin/getinvestigate?sid=x508Z
	781	6.103926	192.168.1.101	14.17.42.24	HTTP	801	POST	<pre>/cgi-bin/compose_send?sid=x508ZuW</pre>
	1051	7.403270	192.168.1.101	183.60.15.162	HTTP	1042	POST	/cgi-bin/getinvestigate?sid=x508Z

从数据包的结构上看应该就是第二至第六个数据包是数据传输的过程。点开第二个可以看到MediaType的长度为131436=;

> [96 Reassembled TCP Segments (132317 bytes): #38(881), #39(1440), #41(1440), #42(1440), #43(1440), #44(1440), #46(144)
> Hypertext Transfer Protocol

✓ Data (131436 bytes)

Data: abcd9876000003ef000000000000002015c01307cb0f705d9067792b5023cb028ba442e6533... [Length: 131436]

第二到第五个都是一样的长度,第六个为1777,应该是剩余的最后一部分数据。但是131436*4+1777=527521!=525701,再看下第一个数据包:

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F		^
00000030	C4	06	0B	3E	75	6C	14	21	2E	F3	7E	7F	87	6F	7C	81	Ä>ul.!.ó~.‡o .	
00000040	5A	F7	FC	04	63	DF	85	F9	21	A6	3E	FB	E4	В4	Fl	4E	Z÷ü.cß…ù!¦>ûä´ñN	
00000050	E1	21	93	DE	81	СВ	69	BE	1B	8B	4B	50	10	ЗF	68	AF	á!"Þ.Ëi¾.∢KP.?h	
00000060	BD	39	58	A0	DF	19	BC	39	FB	52	F4	35	FD	4E	9D	DO	ы9X ß.⊷9ûRô5ýN.Đ	
00000070	Α6	70	53	02	A 2	5E	F7	10	09	B 8	73	48	94	7F	07	2E	¦pS.¢^÷,sH″	
00000080	8A	91	A 7	48	8F	B6	2B	Α9	2C	C4	9C	4D	50	8E	Α4	87	Š`§H.¶+©,ÄœMPޤ‡	
00000090	03	0D	7A	0D	C0	6E	A1	17	C4	20	41	\mathbf{FC}	55	0A	1F	3B	z.Àn;.Ä AüU;	
000000A0	C5	D2	8D	63	20	65	59	FA	F2	A3	BA	A 7	B9	\mathbf{FC}	88	50	ÅÒ.c eYúò£°§¹ü^P	
000000B0	C1	2A	49	86	A 7	B2	81	92	DO	78	56	DE	86	AB	51	F3	Á*I†§°.′ÐxVÞ†«Qó	
00000000	1E	23	F9	E9	C8	ЗD	26	26	E0	9F	38	0B	F9	6E	60	DD	.#ùéÈ=&&àŸ8.ùn`Ý	
00000D0	F2	39	2A	84	43	C7	88	DF	D2	C8	CC	76	63	28	89	04	ò9*"CÇ^ßÒÈÌ∨c(‰.	
000000E0	17	ED	1D	E7	AA	AA	57	7B	CB	9D	1C	FB	FA	1E	FF	76	.í.窪W{Ëûú.ÿ∨	
000000F0	39	AC	E4	4B	BB	32	B8	B0	DB	D9	A3	5B	06	07	47	FD	9⊣äK≫2,°ÛÙ£[Gý	
00000100	5D	10	79	C0	CA	5C	9D	37	1F	E3	50	B8	F4	3C	7B	7A].yÀÊ\.7.ãP_ô<{z	
00000110	8E	07	FA	1F	80	03	32	68	69	76	8E	FA	1E	77	02	1E	Ž.ú.€.2hi⊽Žú.w	
00000120	2D	88	11	FO	\mathbf{FC}	24	D7	13	ЗD	48	F3	30	1B	A1	57	40	-^.ðü\$×.=Hó0.;₩@	
00000130	3B	8B	BA	В9	12	64	C5	C1	E3	Ε4	60	07	CB	4D	ЗE	8A	;<°`.dĂÁãä`.ËM>Š	
00000140	57	35	00	14	EC	CC	BA	7A	EA	1D	48	26	84	37	4B	22	W5ḭ̀zê.H&"7K"	
00000150	E2	E7	AB	AD	2B	A 8	67	49	00	08	05	85	00	00	00	00	âç«.+"gI	
00000160	00	02	00	00	00	00	00	00	00	00	00	00	52	61	72	21	Rar!	

都知道rar文件头应该是Rar,但是选中的数据部分前面却多出了很多,简单计算一下一共多出了364,且364*5+525701=527521,所以多出的也许是某种校验数据,在导出的时候将其忽略;

每个包都做同样的操作即可得出5个文件,再将这个文件按顺序拼接即可。拼接的话可以使用16进制编辑器手动拼接,也可以使用linux下cat 命令,比如"cat 1 2 3 4 5 > fly.rar"; '这道题还设置了伪加密,需要修改加密位,将0x84位置改为0x80即可;

	OF	0E	0D	0C	0B	0A	09	08	07	06	05	04	03	02	01	00	Offset(h)
Rar!Ï.s	00	00	00	0D	00	00	73	90	CF	00	07	1A	21	72	61	52	00000000
ù.t <mark>8</mark> =†	86	00	08	05	ЗD	00	2D	90	80	74	81	F9	00	00	00	00	00000010
"%-íñ.\YF.5	00	08	35	1D	46	59	5C	09	F1	ED	2D	25	02	00	0F	22	00000020
flag.txt.ðy.	03	79	FO	00	74	78	74	2E	67	61	6C	66	00	00	00	20	00000030
L%.Üï£rK.	90	4B	72	A3	EF	1D	16	DC	15	89	0C	15	15	1E	18	4C	00000040
°\$:R#.".ÈAÄ",@	40	82	84	C4	41	C8	02	22	05	23	52	ЗA	24	08	90	B0	00000050
:ì\$‡D:X;′o`	60	6F	92	81	08	18	A1	58	ЗA	44	87	24	EC	ЗA	04	9D	00000060

解压出来后是一个exe可执行文件,里面隐藏了一个png图片,是个二维码,扫描即可得到flag



信息提取

题目文件: sqlmap.pcap

题目描述:数据库中的flag被偷走了,好在全过程我们都有记录

题目题解: 数据包记录的是sqlmap获取flag的过程,使用http && http contains"flag"过滤一下

http and http contains"flag" +																																			
6	No.		Tim	e			S	ouro	e					De	stin	ati o:	n			Pr	otocol	Le	ength	Info	,										^
		776	11.	.967	929)	1	0.0	.0.1	101				10	.0.	0.2	01			HT	ТР		552	GET	· /ı	nessae	ze.p	hp?i	d=1	620A	ND%	200RI) 🗋		۷
Γ	> Tr	ansm	iss	ion	Cor	ntro	51 F	rot	toco	1, 9	Src	Po	rt:	429	902,	Ds	st F	ort:	80,	Sec	q: 1,	Ac	k: 1	l, Le	en:	521									^
ŀ	~ Ну	pert	ext	Tra	anst	Fer	Pro	oto	ol	ĺ.									, i																
	>	GET	/me	essa	ge.	php	?id	=-3	413%	620U	NIO	N%2	ØAL	L%2	ØSE	LEC	T%2	ØNUL	L%2C0	CONC	AT%28	0x7	7173	6363	71%	62CIFN	ULL	%28C	AST?	628C(оилт	%28%	60v		¥
Γ	0040	b9	f4	47	45	54	20	2f	6d	65	73	73	61	67	65	2e	70		GET	/m (essag	e.p)											_	^
	0050	68	70	3f	69	64	3d	2d	33	34	31	33	25	32	30	55	4e	hp	?id=	-3 4	413%2	0UN													
	0060	ð <mark>4</mark> 9	4f	4e	25	32	30	41	4c	4c	25	32	30	53	45	4c	45	I)N%20	AL	L%20S	ELE													
L	0070	ð <mark>4</mark> 3	54	25	32	30	4e	55	4c	4c	25	32	43	43	4f	4e	43	CT	1 <mark>%2</mark> 0N	IUL I	L%2CC	onc													
L	0080	ð <mark>41</mark>	54	25	32	38	30	78	37	31	37	33	36	33	36	33	37	A	° <mark>%2</mark> 80	x7 :	17363	637													
	0090	31	25	32	43	49	46	4e	55	4c	4c	25	32	38	43	41	53	19	62CIF	NU	LL%28	CAS													
L	00a0	ð <mark>5</mark> 4	25	32	38	43	4f	55	4e	54	25	32	38	25	36	30	76	т۶	628C0	UN 1	T%28%	60v													
	00b0	ð <mark>61</mark>	6c	75	65	25	36	30	25	32	39	25	32	30	41	53	25	al	lue%6	0% :	29%20	AS%													
	00c0	32	30	43	48	41	52	25	32	39	25	32	43	30	78	32	30	20)CHAR	%2 S	9%2CØ:	x20													
	00d0	ð <mark>25</mark>	32	39	25	32	43	30	78	37	31	36	66	37	35	37	33	%2	29%20	0x 1	716f7	573													
	00e6	37	31	25	32	39	25	32	30	46	52	4f	4d	25	32	30	69	71	.%29%	20	FROM%	20i													
	00f0	3 73	67	2e	66	6c	61	67	73	25	32	33	20	48	54	54	50	se	g.fla	igs 5	%23 H	TTP													2
L	0100) 2f	31	2e	31	0d	0a	41	63	63	65	70	74	2d	4c	61	6e	/1	.1	Ac o	cept-	Lan													2

将其payload解码一下是这样的,判断其ascii码是否大于64

id=1 AND ORD(MID((SELECTIFNULL(CAST(value AS CHAR),0x20) FROM isg.flags ORDER BY value LIMIT0,1),1,1))>64 然后一直到836个包判断第一位ascii码值大于72,然后开始从高到低递减,判断其ascii码不大于73,则第一位的ascii码值是73,对应的字符为I。以此类推,其flag为ISG{BLind_SQI_InJEcTiON_DeTEcTEd}。本题需要一定的耐心和SQL注入基础。但是这么做可能有些繁琐,其实 pcap数据包可以直接用文本编辑器打开,就可以看到其中的http请求



所以可以使用字符串搜索的方式直接去查找其中的语句,然后判断flag,首先将原数据包中的http请求导出来,另存为sqli.pcap

× flag.py	flag.py ×
	1 import re
	2 import urllib
	3
	4 flag = ""
	5 key = 1
	<pre>6 pcap = open("sqli.pcap","rb")</pre>
	<pre>7 lines = pcap.readlines()</pre>
	8 for line in lines:
	<pre>9 line = line.strip("\n")</pre>
1	<pre>0 line = urllib.unquote(line)</pre>
1	<pre>sql = re.search(r".*LIMIT 0,1\),(\d+),1\)\)>*(\d*)",line)</pre>
1	<pre>2 response = re.search(r"Content-Length: (\d*)",line)</pre>
1	3 if response:
1	<pre>4 Content_Length = response.groups()[0]</pre>
1	5 if sql:
1	<pre>count,asc = sql.groups()</pre>
1	<pre>if str(key) == count:</pre>
1	asc_2 = asc
1	9 else:
2	<pre>if int(Content_Length)>150:</pre>
2	1 flag += chr(int(asc_2)+1)
2	2 else:
2	$flag += chr(int(asc_2))$
2	key += 1
2	print flag
2	5
ISG{BLind_SQl_I [Finished in 0.4	https://blog.csdn.net/LYJ20010728