

# 网络对抗原理与技术

## ——扫描与嗅探

—

尹钰 yinyu@xidian.edu.cn

# 从信息收集开始

目标长啥样

域名

站点

IP

接口

OS

业务

端口

第三方

服务

数据



# 收集域名

意味着承载了业务

访问 + 抓包

Google、github

遍历 + 暴力猜测

利用 DNS 服务器配置缺陷

从公司那直接得到

---

# 收集 IP

目标的基本粒度  
之一

域名

IP 段反推

特征

从公司那直接得到

---

# 端口扫描

缺陷的基本粒度  
之一

端口号

类型

服务版本

---

# 以 nmap 为例

如何面对工具

```
[kussa@Kussa ~]$ nmap -n 192.168.1.1
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-03-02 11:36 CST  
Nmap scan report for 192.168.1.1
```

```
Host is up (0.0093s latency).
```

```
Not shown: 996 closed ports
```

```
PORT      STATE SERVICE
```

```
23/tcp    open  telnet
```

```
80/tcp    open  http
```

```
1723/tcp  open  pptp
```

```
1900/tcp  open  upnp
```

```
Nmap done: 1 IP address (1 host up) scanned in 1.35 seconds
```

```
[kussa@Kussa ~]$ █
```

```
[root@Kussa kussa]# nmap -n 192.168.1.2
```

---

# 扫描的过程

发、收、判断特征、结论

准备工作

目标调度

IP 存活检查

端口探测

版本探测

OS 探测

进一步探测

结果整理

---

# 目标描述

发、收、判断特征、结论

IP

192.168.1.0/24

192.168.1-23.1,2

Port

U:53,111,137

T:21-25,80,139

T:80,22,U:53

---



# 主机存活

ping ?

Ping	-PE
Other ICMP	-PP -PM
TCP SYN	-PS
TCP ACK	-PA
UDP	-PU
不判断	-P0 -Pn

---

# 端口扫描

端口状态

open

close

filtered

组合

---

# 端口扫描

开放情况

TCP Connect      -sT

TCP SYN            -sS

TCP ACK            -sA

UDP                 -sU

.....

---

# 时间参数

卡啦？ OK！

--min-rtt-timeout

--max-rtt-timeout

--initial-rtt-timeout

--max-retries

--host-timeout

--scan-delay

---

# 服务 / 版本探测

获取更多的信息

-sV

--version-intensity 0-9 7

--version-light 2

--version-all 9

---

# OS 探测

获取更多的信息

-0

似乎没啥好说的

用的其实比想象中的少

---

# NSE

从工具到框架的升华

不用考虑

目标调度

存活性

端口开放情况

服务情况

只要关心

发什么 / 怎么分析

---

# 结果输出

选择合适的格式

-oN

-oX

---



# 以 nmap 为例

如何面对工具

```
[kussa@Kussa ~]$ nmap -n 192.168.1.1
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-03-02 11:36 CST  
Nmap scan report for 192.168.1.1
```

```
Host is up (0.0093s latency).
```

```
Not shown: 996 closed ports
```

```
PORT      STATE SERVICE
```

```
23/tcp    open  telnet
```

```
80/tcp    open  http
```

```
1723/tcp  open  pptp
```

```
1900/tcp  open  upnp
```

```
Nmap done: 1 IP address (1 host up) scanned in 1.35 seconds
```

```
[kussa@Kussa ~]$ █
```

```
[root@Kussa kussa]# nmap -n 192.168.1.2
```

---

# 其他工具

用在合适的场合

ping

-c -i -s -t

telnet

hping

---

# 狭义嗅探

libpcap 衍生

混杂模式

libpcap

wireshark

tcpdump

winpcap

---

# 应用场景

典型？ 普遍

嗅探未加密的敏感信息

被动探测 / 收集信息

逆向

排查问题

---

# BPF 语法

Berkeley Packet Filter

host 1.2.3.4 or net 10.0.0.0/8

ether src host 11:22:33:44:55:66

tcp src port 80 or udp port 53

ip[6] & 0x40 != 0

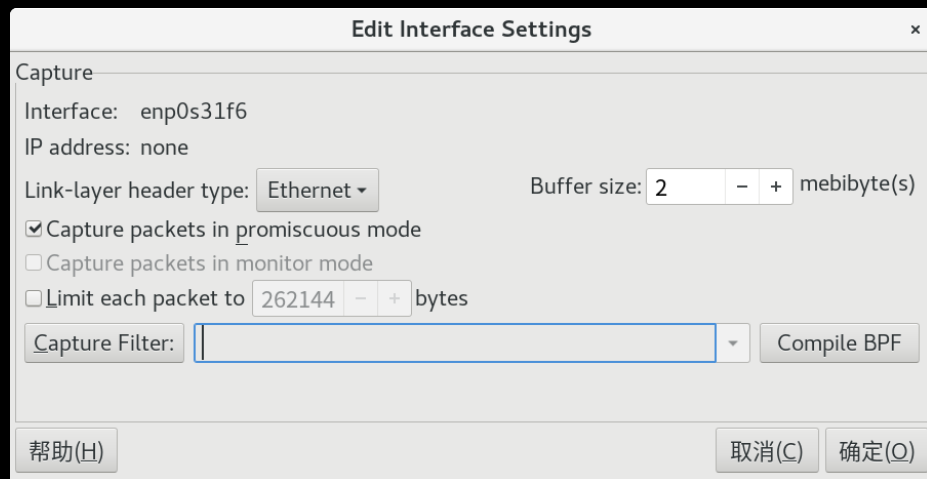
tcp[13] & 0x03 != 0

tcp[0:2] < 0x0400

# wireshark

## GUI 抓包 / 分析工具

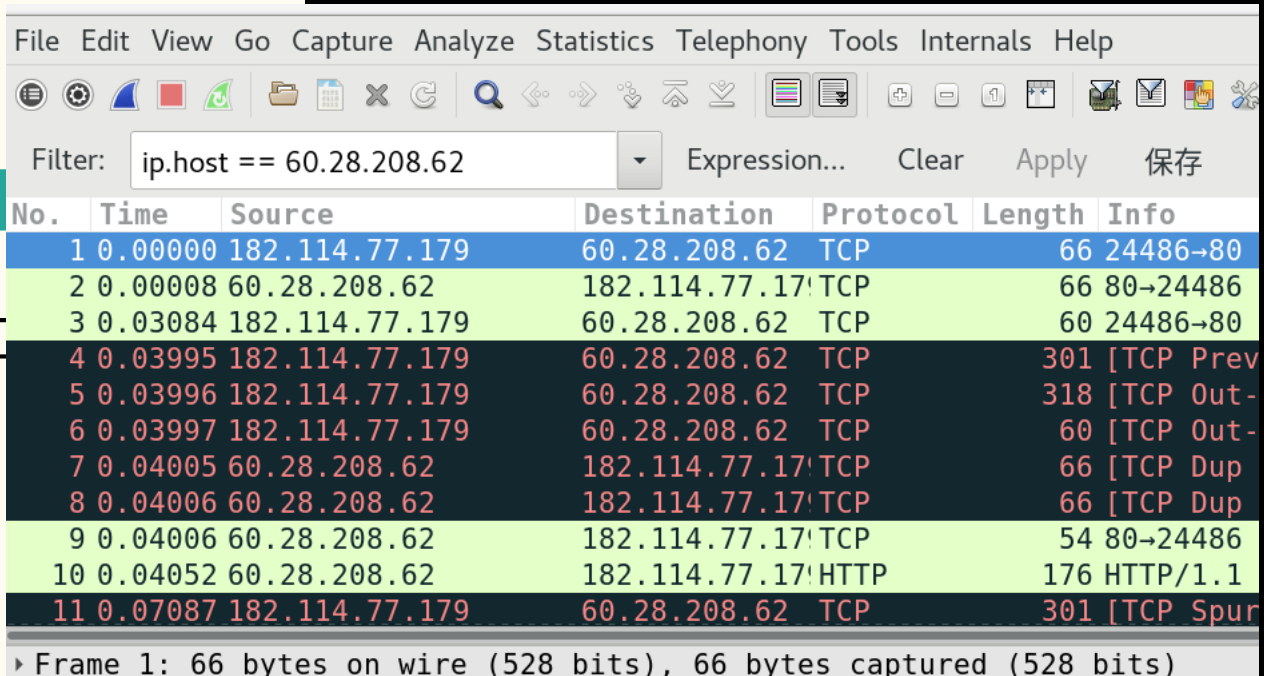
### 本地



# wireshark

## GUI 抓包 / 分析工具

本地



Wireshark interface showing a list of captured packets. The filter is set to `ip.host == 60.28.208.62`. The table displays packet details including No., Time, Source, Destination, Protocol, Length, and Info.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.00000	182.114.77.179	60.28.208.62	TCP	66	24486→80
2	0.00008	60.28.208.62	182.114.77.179	TCP	66	80→24486
3	0.03084	182.114.77.179	60.28.208.62	TCP	60	24486→80
4	0.03995	182.114.77.179	60.28.208.62	TCP	301	[TCP Prev
5	0.03996	182.114.77.179	60.28.208.62	TCP	318	[TCP Out-
6	0.03997	182.114.77.179	60.28.208.62	TCP	60	[TCP Out-
7	0.04005	60.28.208.62	182.114.77.179	TCP	66	[TCP Dup
8	0.04006	60.28.208.62	182.114.77.179	TCP	66	[TCP Dup
9	0.04006	60.28.208.62	182.114.77.179	TCP	54	80→24486
10	0.04052	60.28.208.62	182.114.77.179	HTTP	176	HTTP/1.1
11	0.07087	182.114.77.179	60.28.208.62	TCP	301	[TCP Spur

▶ Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)

## Wireshark: Preferences - Profile: Default

SRVLOC  
SSCOP  
SSH  
SSL  
STANAG 5066 DTS  
STANAG 5066 SIS  
StarTeam  
STP  
STT  
SUA  
SV  
SYNCHROPHASOR  
T.38  
TACACS+  
TALI  
TCAP  
TCP

帮助(H)

Show TCP summary in protocol tree: ☒

Validate the TCP checksum if possible: ☐

Allow subdissector to reassemble TCP streams: ☒

Analyze TCP sequence numbers: ☒

Relative sequence numbers: ☐

Scaling factor to use when not available from capture: Not known ▾

Track number of bytes in flight: ☒

Calculate conversation timestamps: ☐

Try heuristic sub-dissectors first: ☐

Ignore TCP Timestamps in summary: ☐

Do not call subdissectors for error packets: ☒

TCP Experimental Options with a Magic Number: ☒

Display process information via IPFIX: ☐

应用(A)

取消(C)

确定(O)



## Wireshark: Preferences - Profile: Default

H263P  
H264  
HART\_IP  
HAZELCAST  
HCI\_ACL  
HCI\_CMD  
HCI\_EVT  
HCI\_MON  
HCI\_USB  
HCrt  
HDFS  
HDFSDATA  
HiSLIP  
HNBAP  
HP\_ERM  
HPFEEDS  
HTTP

Reassemble HTTP headers spanning multiple TCP segments: ☒

Reassemble HTTP bodies spanning multiple TCP segments: ☒

Reassemble chunked transfer-coded bodies: ☒

Uncompress entity bodies: ☒

TCP Ports:

SCTP Ports:

SSL/TLS Ports:

Custom HTTP header fields:

帮助(H)

应用(A)

取消(C)

确定(O)

# tcpdump

CLI 抓包工具

目标服务器上

-i      -e      -c      -w      -p

-n      -nn

-v      -vv      -vvv

-X      -XX

---

# 广义抓包

更上层，更多细节，更多限制

tcpflow

代理抓包

Paros、ZAP、Fiddler

浏览器抓包工具 / 插件

FireBug、Chrome、HTTP Watch

手机抓包

真机配代理

模拟器配代理

——AP 抓包

# 流还原

发包 ++

tcpreplay

tcpprep

tcprewrite

tcpdump

---