CS305 Lab2 Introduction to Python & Wireshark

Dept. Computer Science and Engineering
Southern University of Science and Technology

SUSTech
Southern University
of Science and Technology

Thanks to HHQ. Zhang

Part. A

Introduction to Python

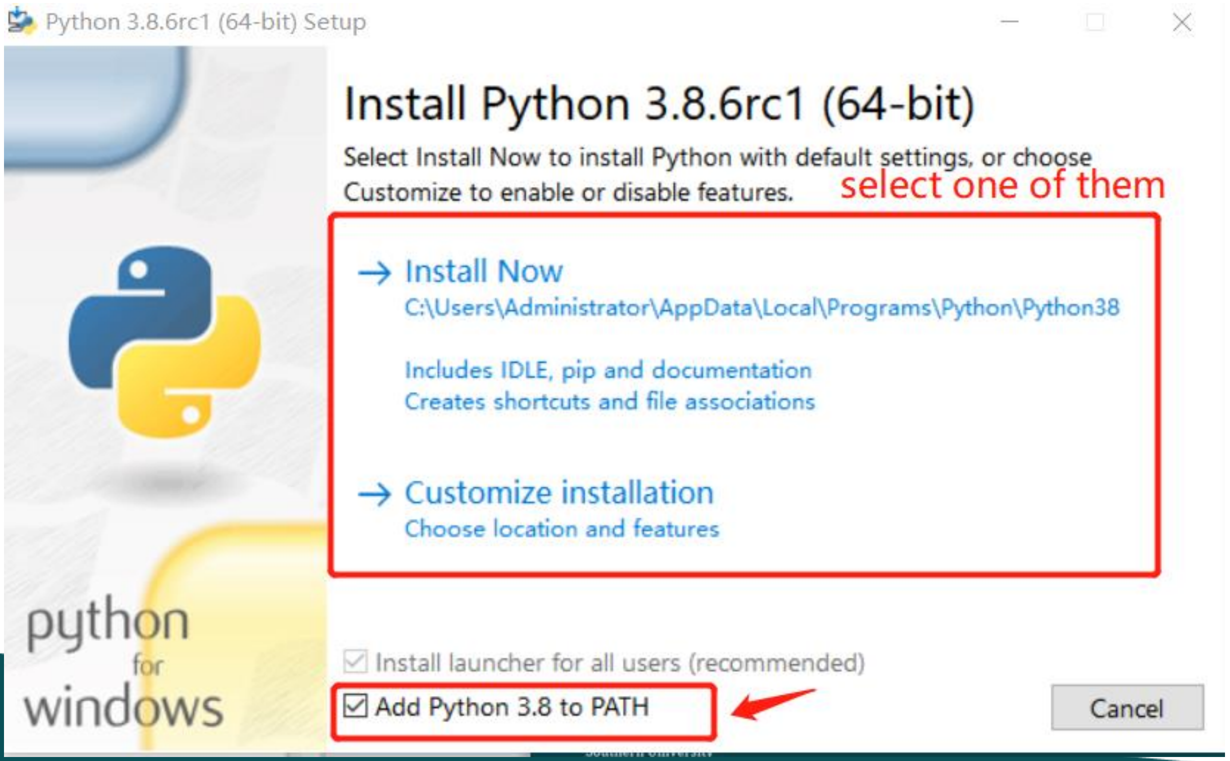
SUSTech
Southern University
of Science and Technology

Python

- Python is an interpreted high-level object-oriented programming language.
- First release in 1991.
- Official Tutorial: https://docs.python.org/3/tutorial/

Install python(1)

- The Installation package can be got from **https://www.python.org/downloads/**
- You can choose install it by **default settings** or **customize installation**.
- It is highly recommend that choose 'Add Python xx to PATH', or you need to set PATH by hand.



Python 3.8.6rc1 (64-bit) Setup

Install Python 3.8.6rc1 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

select one of them

→ Install Now
C:\Users\Administrator\AppData\Local\Programs\Python\Python38

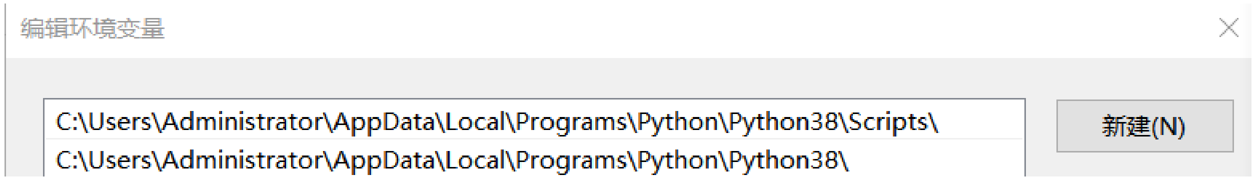
Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.8 to PATH

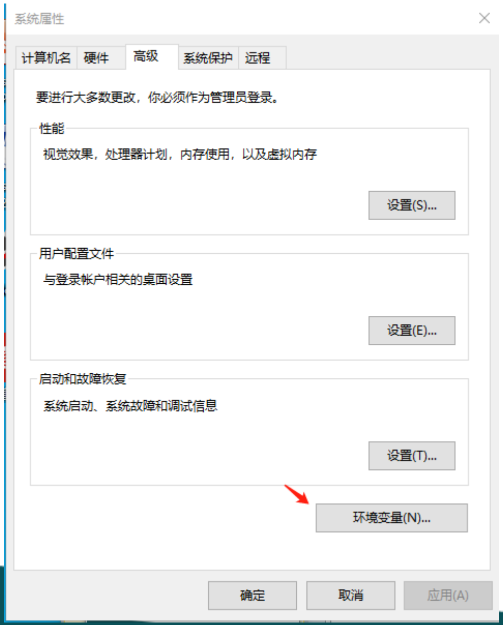
Install python(2)

- If the ‘Add Python xx to PATH’ is not set while installing, configure ‘Path’ manually according to the following steps after the installation.
  - Right click ‘my computer’ on the desktop
  - select ‘attribute’-> ‘advanced attribute’->environment variable
  - configure ‘Path’ with the path where python.exe belongs and its subdirectory ‘Scripts’



编辑环境变量

```
C:\Users\Administrator\AppData\Local\Programs\Python\Python38\Scripts\
C:\Users\Administrator\AppData\Local\Programs\Python\Python38\
```



系统属性

要进行大多数更改，你必须作为管理员登录。

性能

视觉效果，处理器计划，内存使用，以及虚拟内存

用户配置文件

与登录帐户相关的桌面设置

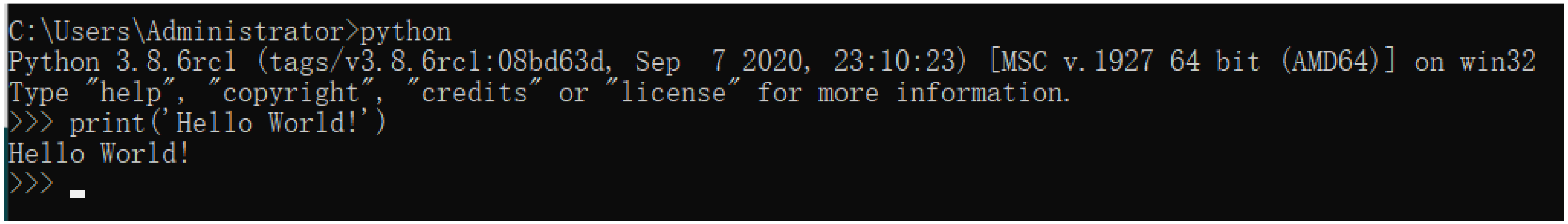
启动和故障恢复

系统启动、系统故障和调试信息

SUSTech Southern University of Science and Technology

Read-Eval-Print Loop

- Python has an REPL playground.
- Type and get feedback.



```
C:\Users\Administrator>python
Python 3.8.6rc1 (tags/v3.8.6rc1:08bd63d, Sep  7 2020, 23:10:23) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hello World!')
Hello World!
>>> 
```

SUSTech Southern University of Science and Technology

Basic Types and Operations

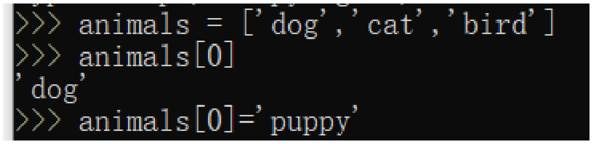
- The following standard types are built in the interpreter:
  - **Numeric** Types — int, float, complex
  - **Boolean** Type — True, False
  - **Text Sequence** Type — str
  - **Sequence** Types — list, tuple, range
  - **Set** Type & **Dict** Type
  - **Binary Sequence** Types — bytes, byte array
- There are predefined operations on each type

- Ref: https://docs.python.org/3/library/stdtypes.html

Sequence Types

- **List**

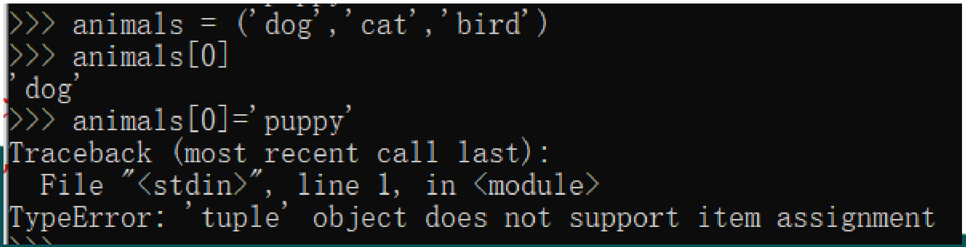
```
animals = ['dog', 'cat', 'bird']
animals[0] # => 'dog'
animals[0] = 'puppy'
```



```
>>> animals = ['dog','cat','bird']
>>> animals[0]
'dog'
>>> animals[0]='puppy'
```

- **Tuple**

```
animals = ('dog', 'cat', 'bird')
animals[0] # => 'dog'
animals[0] = 'puppy'
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```



```
>>> animals = ('dog','cat','bird')
>>> animals[0]
'dog'
>>> animals[0]='puppy'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

SUSTech
Southern University
of Science and Technology

Unpacking from Sequence Types

- **List**

  foo, bar = **[**'dog', 'cat'**]**

  foo # => 'dog'

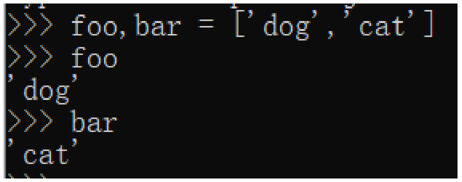
  bar # => 'cat'

- **Tuple**

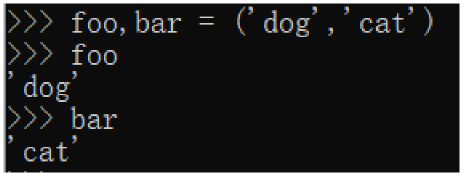
  foo, bar = **(**'dog', 'cat'**)**

  foo # => 'dog'

  bar # => 'cat'



```
>>> foo, bar = ['dog', 'cat']
>>> foo
'dog'
>>> bar
'cat'
```



```
>>> foo, bar = ('dog', 'cat')
>>> foo
'dog'
>>> bar
'cat'
```

SUSTech Southern University of Science and Technology

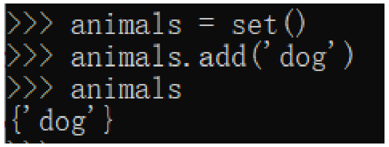
Set & Dict

- **Set**

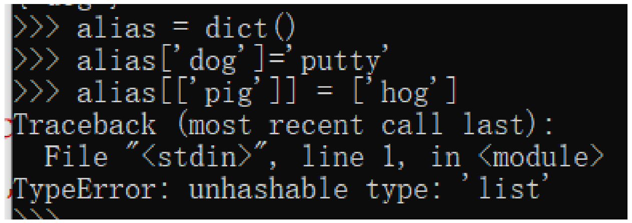
  ```
  animals = set()
  animals.add('dog')
  animals  # => {'dog'}
  ```

- **Dict**

  ```
  alias = dict()
  alias['dog'] = 'puppy'
  alias[['pig']] = ['hog']
  Traceback (most recent call last):
      File "<stdin>", line 1, in <module>
  TypeError: unhashable type: 'list'
  ```



```
>>> animals = set()
>>> animals.add('dog')
>>> animals
{'dog'}
```



```
>>> alias = dict()
>>> alias['dog']='putty'
>>> alias[['pig']] = ['hog']
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unhashable type: 'list'
```

Immutable & Mutable

- Mutable: it is possible to change its content
- **Immutable Type: Numeric, Boolean, str, tuple, bytes, etc.**
- **Mutable Type: list, dict, set, etc.**

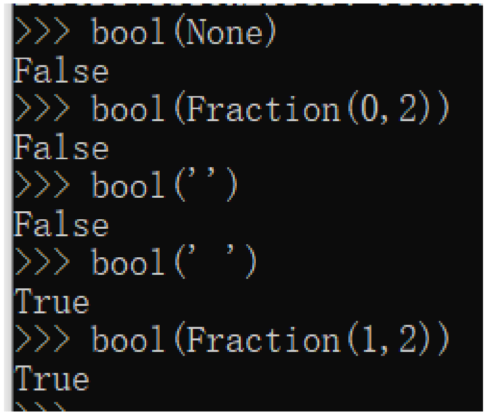
- Example:

```
>>> cubes = [1, 8, 27, 65, 125]    # cubes here is a list
>>> cubes[3] = 64                  # replace the item whose index is 3
>>> cubes
[1, 8, 27, 64, 125]
```

Boolean Values

- Following values are treated as **False**:
  - **None**, **False**
  - **0**, **0.0**, **0j**, **Decimal(0)**, **Fraction(0, 1)**
  - **''**, **()**, **[]**, **{}**, **set()**, **range(0)**

- Otherwise they are **True**



```
>>> bool(None)
False
>>> bool(Fraction(0,2))
False
>>> bool('')
False
>>> bool(' ')
True
>>> bool(Fraction(1,2))
True
```

SUSTech Southern University of Science and Technology

Flow Control — if

- Example:

```
foo = []
if foo:
    print(foo)
else:
    if foo == []:
        print('100% sure foo is empty')
    else:
        print('what hell?')
```

Flow Control — if

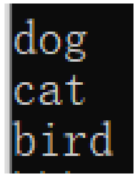
- Example:

```
foo = [1, 2, 3, 4]
if foo:
    print(foo)
else:
    if foo == []:
        print('100% sure foo is empty')
    else:
        print('what hell?')
```

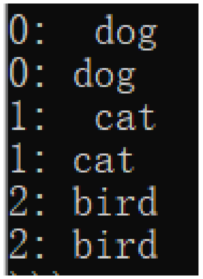
Flow Control — for

- Example:

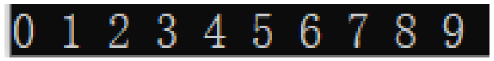
```
foo = ['dog', 'cat', 'bird']
for bar in foo:
    print(bar)
for index, value in enumerate(foo):
    print('%d: %4s' % (index, value))
    print('{0}: {1}'.format(index, value))
for i in range(10):
    print(i,end=" ")
```



```
dog
cat
bird
```



```
0:  dog
0: dog
1:  cat
1: cat
2: bird
2: bird
```

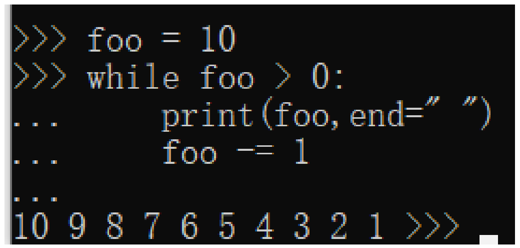


```
0 1 2 3 4 5 6 7 8 9
```

Flow Control — while

- Example:

```
foo = 10
while foo > 0:
    print(foo, end=" ")
    foo -= 1
```



```
>>> foo = 10
>>> while foo > 0:
...     print(foo,end=" ")
...     foo -= 1
...
10 9 8 7 6 5 4 3 2 1 >>>
```

Defining Functions

- Example:

```
def fib(n):  # write Fibonacci series up to n
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
    print()
```

Defining Functions

- Example:

```
def fib2(n): # return Fibonacci series up to n
    result = []
    a, b = 0, 1
    while a < n:
        result.append(a)    # see below
        a, b = b, a+b
    return result
```

Closure

- A closure is an inner function that has access to the outer (enclosing) function's variables.
- Example:

```
def add(x):
    def addX(y):
        return y + x
    return addX
foo = add(1)
print(foo(2)) # => 3
```

Practise:
After the defination of function "add", run the following test, what's the testing result?

```
foo = add(1)
print(foo(2))
print(foo(3))
goo = add(100)
print(goo(2))
print(foo(4))
```

Defining Classes

```
class Animal:
    def __init__(self, name):
        self.name = name

class Duck(Animal):
    def __init__(self, name):
        super(Duck, self).__init__(name)
    def quack(self):
        print(self.name, ' Quack')
```

Duck Type

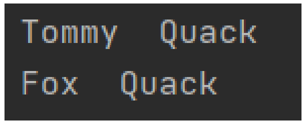
- "If it walks like a duck and it quacks like a duck, then it must be a duck"

```
class Dog(Animal):
    def __init__(self, name):
        super(Dog, self).__init__(name)
    def quack(self):
        print(self.name, ' Quack')
```

Duck Type

- "If it walks like a duck and it quacks like a duck, then it must be a duck"

```
def testDuck(duck):
    duck.quack()

duck = Duck('Tommy')
dog = Dog('Fox')
testDuck(duck)
testDuck(dog)
```



```
Tommy  Quack
Fox  Quack
```

Practise:
1. What's the testing result if run testDuck('duck')?
2. Modify the name of parameter of “testDuck” from “duck” to “x” , will the running result of “testDuck(duck)” and “testDuck(dog)” be changed?
3. If Duck and Dog don't share the same parents, will the running result of “testDuck(duck)” and “testDuck(dog)” be changed?

SUSTech Southern University of Science and Technology

Module

- Save our fib functions(fib and fib2) into fibs.py

```
import fibs
fibs.fib(5) # => 0 1 1 2 3
result = fibs.fib2(5) # => [0, 1, 1, 2, 3]
```

Practise:
Add two functions fib3 and fib4 to fibs.py, the parameter of these two function is the number of first items of fibonacci sequence. fib3 print the specified number of first items of fibonacci sequence, fib4 returen a list which includes the specified number of first items of fibonacci sequence.

for example:

```
fibs.fib3(10) # => 0 1 1 2 3 5 8 13 21 34
result = fibs.fib3(10) # => [0, 1, 1, 2, 3,5,8,13,21,34]
```

SUSTech
Southern University
of Science and Technology

Part. B

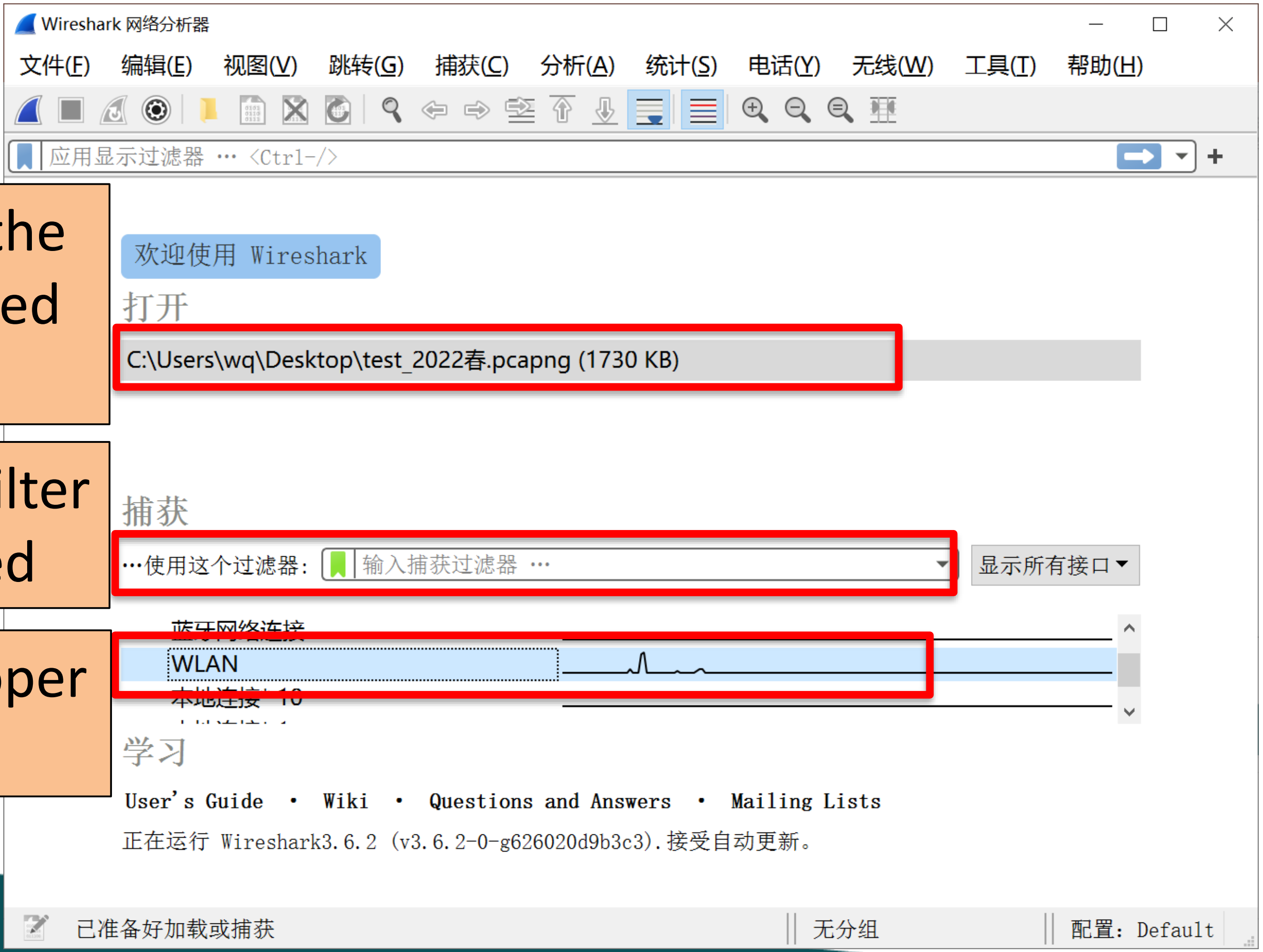
Packet Capture and Analysis

SUSTech
Southern University
of Science and Technology

Wireshark

- Wireshark is a free and open-source packet analyzer.
  It is used for network trouble shooting, analysis, software and communications protocol development, and education.
- Official Website: http://www.wireshark.org/
- Alternative utilities:
  - Tcpdump
  - Tshark
- Tip: new version of Wireshark uses Npcap instead of Winpcap.
- Wireshark User's Guide:
  - file:///C:/Program%20Files/Wireshark/Wireshark%20User's%20Guide/index.html
  - https://gitlab.com/wireshark/wireshark/-/wikis/home

Main Interface


Wireshark 网络分析器
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)
应用显示过滤器 … <Ctrl-/>
欢迎使用 Wireshark
打开
捕获
…使用这个过滤器：
输入捕获过滤器 …
显示所有接口
蓝牙网络连接
WLAN
本地连接* 10
学习
User's Guide · Wiki · Questions and Answers · Mailing Lists
已准备好加载或捕获
无分组
配置: Default

C:\Users\wq\Desktop\test_2022春.pcapng (1730 KB)

正在运行 Wireshark3.6.2 (v3.6.2-0-g626020d9b3c3). 接受自动更新。

We can open the recently-opened file directly

Enter capture filter here if needed

Choose the proper NIC

Capture Filter (1)

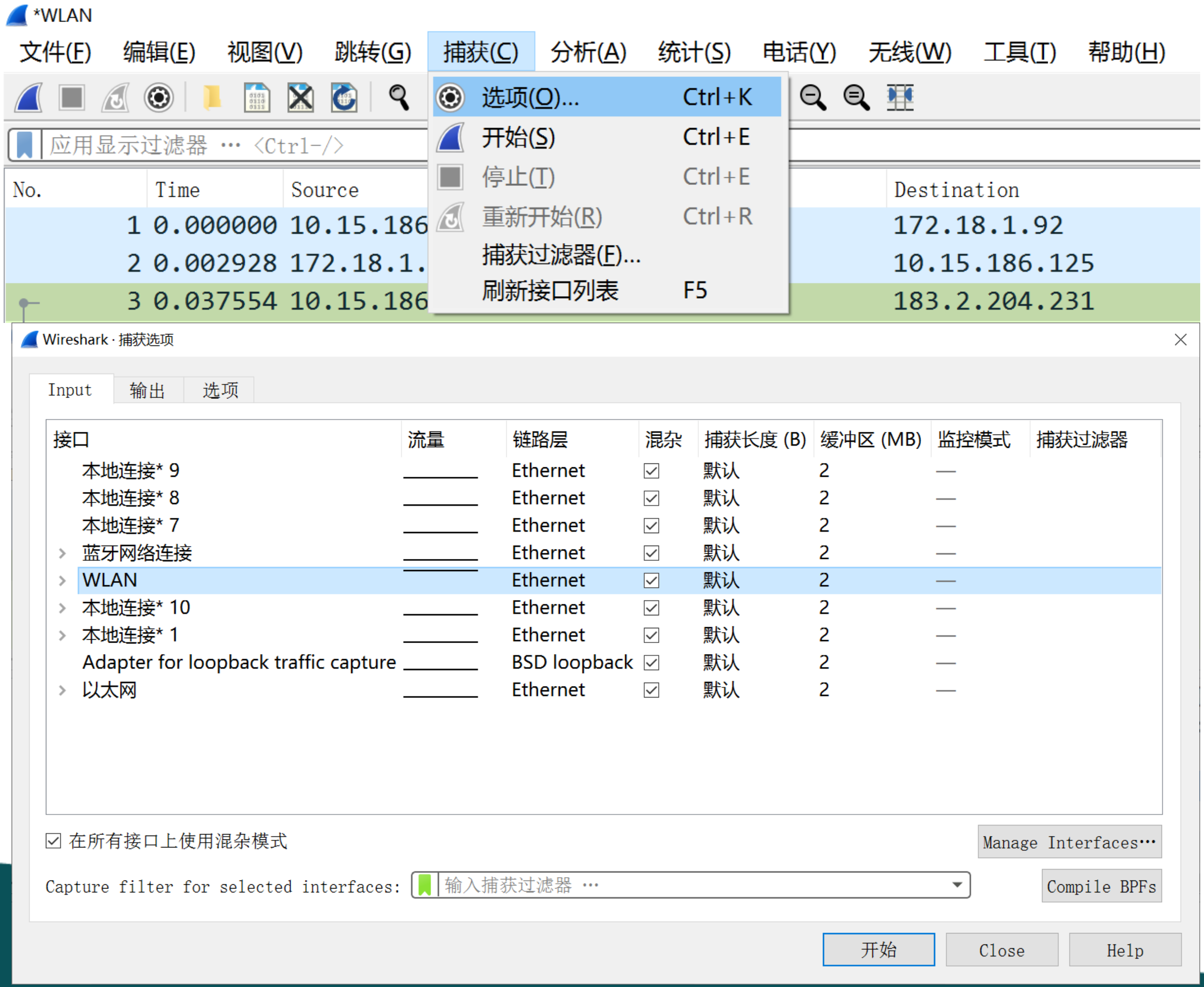
- Capture filter allows you to select the packets you want from all the packets captured by Wireshark.
- A proper capture filter can reduce the workload of Wireshark and the size of raw packets.
- Capture filter **is not** a display filter.
- Wireshark capture filters are written in libpcap filter language.
- Basic syntax: ***[not] primitive [and|or [not] primitive ...]***
  - Green: valid capture filter
  - Red: invalid capture filter

Capture Filter (2)

- Example:
  - host 172.18.5.4: This example captures traffic to and from the host 172.18.5.4
  - port 53: This example captures DNS traffic
  - tcp port 23 and host 10.0.0.5: This example captures telnet traffic to and from the host 10.0.0.5
  - dst net 192.168.0.0/24: This example captures traffic to a range of destination IP address from 192.168.0.0 to 192.168.0.255
- More syntax explanation and examples:
  - https://gitlab.com/wireshark/wireshark/-/wikis/CaptureFilters
  - http://www.tcpdump.org/manpages/pcap-filter.7.html

Capture Filter (3)

- Set capture filters after starting
- Capture -> Options



No.	Time	Source	Destination
1	0.000000	10.15.186	172.18.1.92
2	0.002928	172.18.1.	10.15.186.125
3	0.037554	10.15.186	183.2.204.231

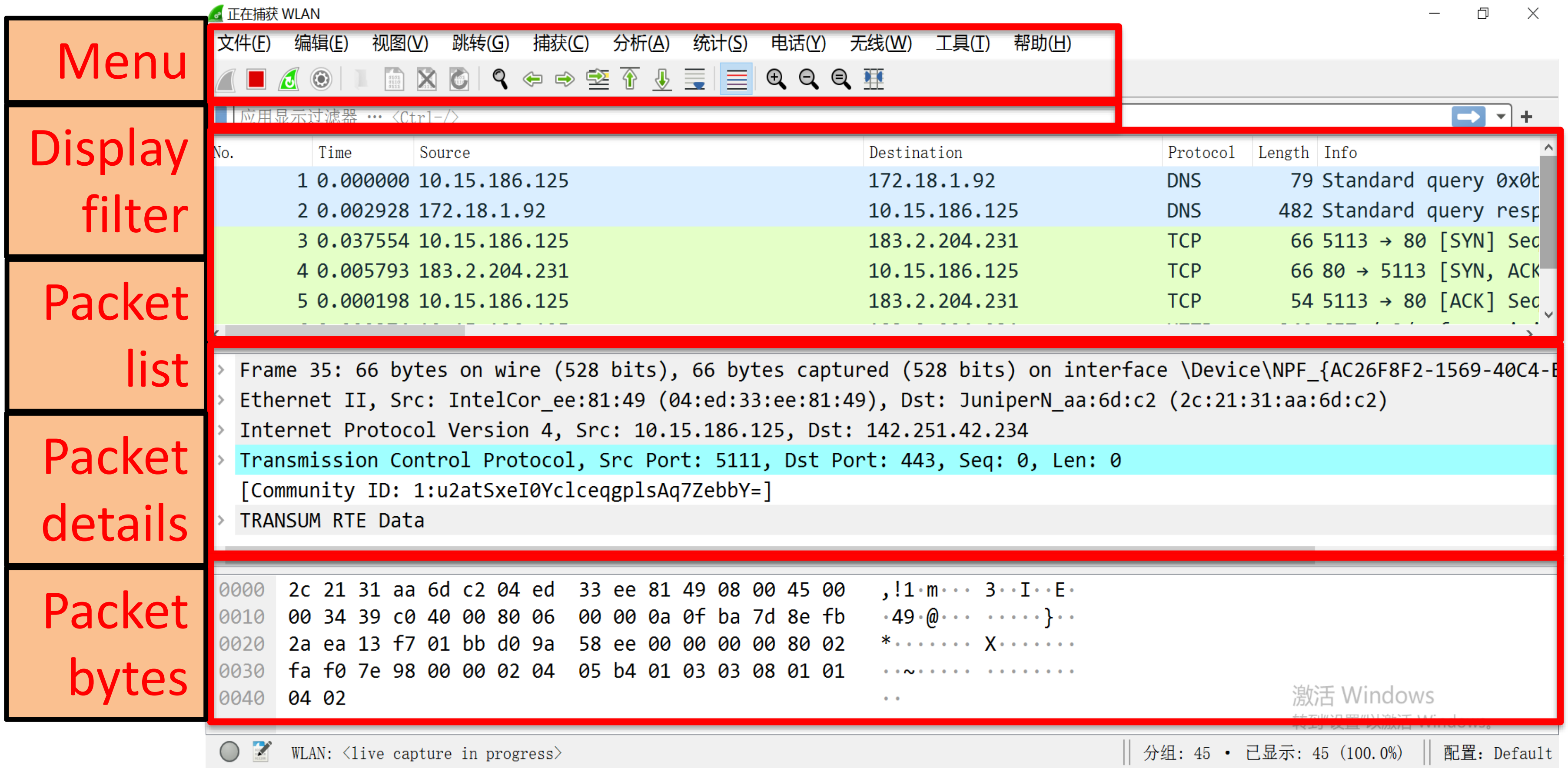
接口	流量	链路层	混杂	捕获长度 (B)	缓冲区 (MB)	监控模式	捕获过滤器
本地连接* 9		Ethernet	✓	默认	2	—	
本地连接* 8		Ethernet	✓	默认	2	—	
本地连接* 7		Ethernet	✓	默认	2	—	
蓝牙网络连接		Ethernet	✓	默认	2	—	
WLAN		Ethernet	✓	默认	2	—	
本地连接* 10		Ethernet	✓	默认	2	—	
本地连接* 1		Ethernet	✓	默认	2	—	
Adapter for loopback traffic capture		BSD loopback	✓	默认	2	—	
以太网		Ethernet	✓	默认	2	—	

☑ 在所有接口上使用混杂模式

Capture filter for selected interfaces: 输入捕获过滤器 …

SUSTech Southern University of Science and Technology

Display Interface


Menu
Display filter
Packet list
Packet details
Packet bytes
正在捕获 WLAN
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)
应用显示过滤器 … <Ctrl-/>

```
No.  Time     Source         Destination    Protocol Length Info
   1 0.000000 10.15.186.125  172.18.1.92    DNS          79 Standard query 0x0b
   2 0.002928 172.18.1.92    10.15.186.125  DNS         482 Standard query resp
   3 0.037554 10.15.186.125  183.2.204.231  TCP          66 5113 → 80 [SYN] Seq
   4 0.005793 183.2.204.231  10.15.186.125  TCP          66 80 → 5113 [SYN, ACK
   5 0.000198 10.15.186.125  183.2.204.231  TCP          54 5113 → 80 [ACK] Seq
```

```
> Frame 35: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{AC26F8F2-1569-40C4-E
> Ethernet II, Src: IntelCor_ee:81:49 (04:ed:33:ee:81:49), Dst: JuniperN_aa:6d:c2 (2c:21:31:aa:6d:c2)
> Internet Protocol Version 4, Src: 10.15.186.125, Dst: 142.251.42.234
> Transmission Control Protocol, Src Port: 5111, Dst Port: 443, Seq: 0, Len: 0
  [Community ID: 1:u2atSxeI0YclceqgplsAq7ZebbY=]
> TRANSUM RTE Data
```

```
0000  2c 21 31 aa 6d c2 04 ed  33 ee 81 49 08 00 45 00   ,!1·m··· 3··I··E·
0010  00 34 39 c0 40 00 80 06  00 00 0a 0f ba 7d 8e fb   ·49·@··· ·····}··
0020  2a ea 13 f7 01 bb d0 9a  58 ee 00 00 00 00 80 02   *······· X·······
0030  fa f0 7e 98 00 00 02 04  05 b4 01 03 03 08 01 01   ··~····· ········
0040  04 02                                              ··
```

WLAN: <live capture in progress>　分组: 45 · 已显示: 45 (100.0%)　配置: Default

Display Filter

- After the capture starts, the display filter can be set to accurately hide the packet you don’t care.
- Display filter can be change at anytime on the fly.
- Filters are evaluated against each individual packet.
- Boolean expressions dealing with packet properties.
- Supports regular expressions.
- Can either be manually constructed, composed via the expressions menu or composed based on a selected packet's properties.

Build Display Filter Expressions

- Enter regular expressions in filter text box
  - Green: valid filter
  - Red: invalid filter
  - Yellow: may produce unexpected results
- Packet based filter
  - Filters can be constructed on the basis of individual packets by right clicking on a packet and selecting either:
  - Prepare as filter: creates a filter
  - Apply as filter: creates a filter and applies it to the trace
  - Follow TCP Stream: creates a filter from a TCP packet's stream number and applies it to the trace.

Display Filter Expressions (1)

- Uses Perl regex syntax
- Comparing Values
- Compound Filters

Table 1. Display Filter Comparison Operators

English	C-like	Description
eq	==	Equal
ne	!=	Not equal
gt	>	Greater than
lt	<	Less than
ge	>=	Greater than or equal to
le	<=	Less than or equal to
contains		Protocol, field or slice contains a value
matches	~	Protocol or text field matches a Perl-compatible regular expression
Bitwise_and	&	Bitwise AND is non-zero

Table 2. Display Filter Logical Operations

English	C-like	Description
and	&&	Logical AND
or	||	Logical OR
xor	^^	Logical XOR
not	!	Logical NOT
[…]		Subsequence
in		Set Membership

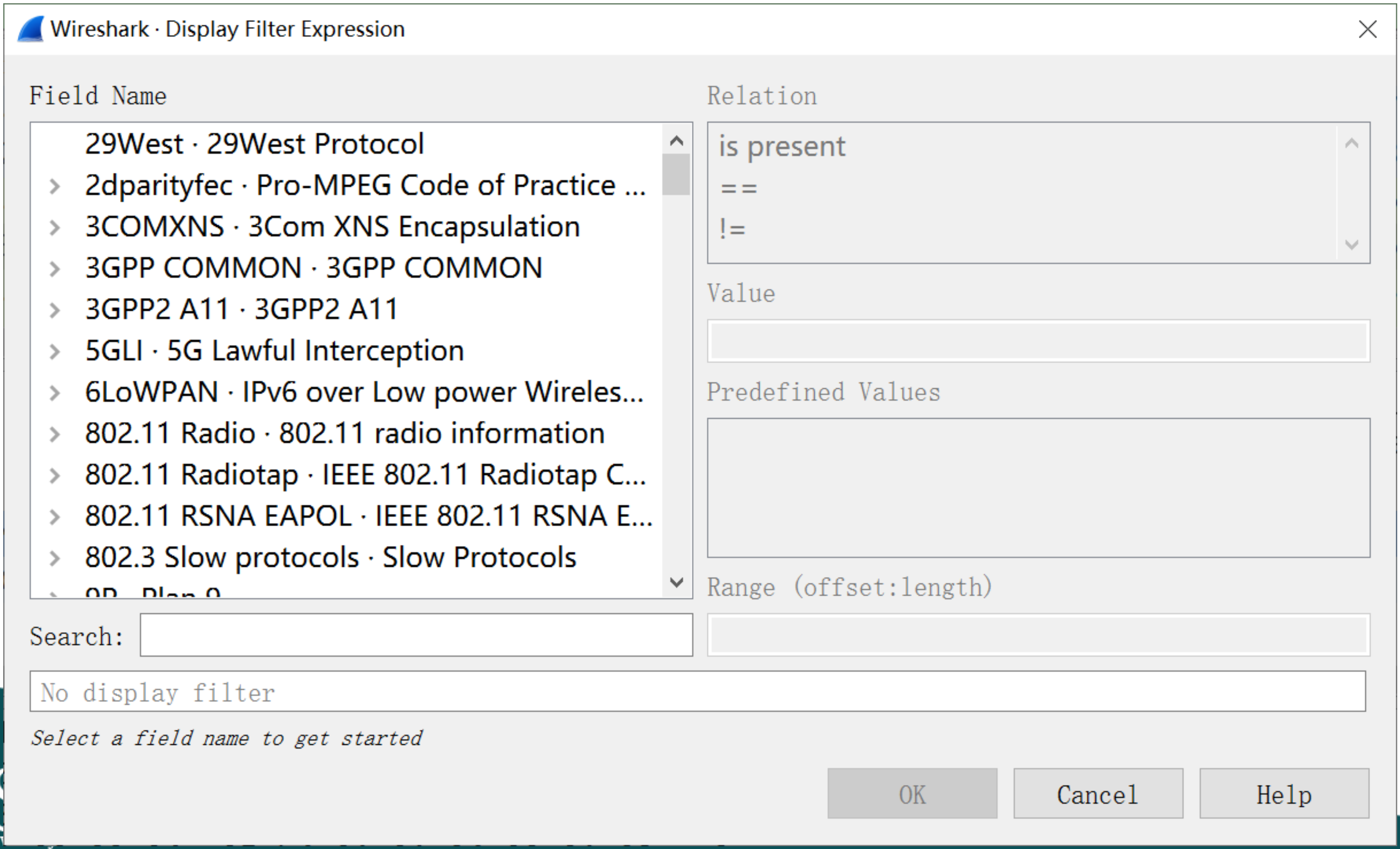
SUSTech Southern University of Science and Technology

Display Filter Expressions (2)

- Examples:
  - tcp.port eq 25 or icmp: Shows only SMTP (port 25) and ICMP traffic.
  - ip.len le 1500: Shows the IP packets whose length field is less than or equal to 1500 bytes.
  - ip.src != xxx.xxx.xxx.xxx && ip.dst != xxx.xxx.xxx.xxx && sip : Filter by a protocol (e.g. SIP) and filter out unwanted IPs.
  - http.request.uri matches "(gif)$": Display all HTTP requests in which the uri ends with "gif".
- More examples:
  - file:///C:/Program%20Files/Wireshark/Wireshark%20User's%20Guide/ChWorkBuildDisplayFilterSection.html
  - https://gitlab.com/wireshark/wireshark/-/wikis/DisplayFilters

“Display Filter Expression” Dialog Box (1)

- Analyze -> Display filter expression…



Wireshark · Display Filter Expression

Field Name

29West · 29West Protocol
2dparityfec · Pro-MPEG Code of Practice ...
3COMXNS · 3Com XNS Encapsulation
3GPP COMMON · 3GPP COMMON
3GPP2 A11 · 3GPP2 A11
5GLI · 5G Lawful Interception
6LoWPAN · IPv6 over Low power Wireles...
802.11 Radio · 802.11 radio information
802.11 Radiotap · IEEE 802.11 Radiotap C...
802.11 RSNA EAPOL · IEEE 802.11 RSNA E...
802.3 Slow protocols · Slow Protocols

Relation

is present
==
!=

Value

Predefined Values

Range (offset:length)

Search:

No display filter

Select a field name to get started

OK Cancel Help

SUS Southern University of Science and Technology

“Display Filter Expression” Dialog Box (2)

- Field name: selects the packet property.
  - Every protocol with filterable fields is listed at the top level.
  - You can search for a particular protocol entry by entering the first few letters of the protocol name.
  - By expanding a protocol name you can get a list of the field names available for filtering for that protocol.
- Relation: selects the boolean test.
- Value: Arbitrary Textual or Numeric value against which the selected packet property is tested.
- Predefined values: common values against which the selected packet property is tested.
- Search
- Range

Capture Filter VS. Display Filter

- Usage
  - Capture filters are much more limited and are used to reduce the size of a raw packet capture.
  - Display filters are used to hide some packets from the packet list.
- Synax
  - tcp port 80
  - tcp.port == 80
- Setting
  - Capture filters are set before starting a packet capture and cannot be modified during the capture.
  - Display filters on the other hand do not have this limitation and you can change them on the fly.

Packet List Pane (1)

- Displays all of the packets in the trace in the order they were recorded.
- Columns
  - Time: the timestamp at which the packet crossed the interface.
  - Source: the originating host of the packet.
  - Destination: the host to which the packet was sent.
  - Protocol: the highest level protocol that Wireshark can detect.
  - Length: the lenght in bytes of the packet on the wire.
  - Info: an informational message pertaining to the protocol in the protocol column.

Packet List Pane (2)

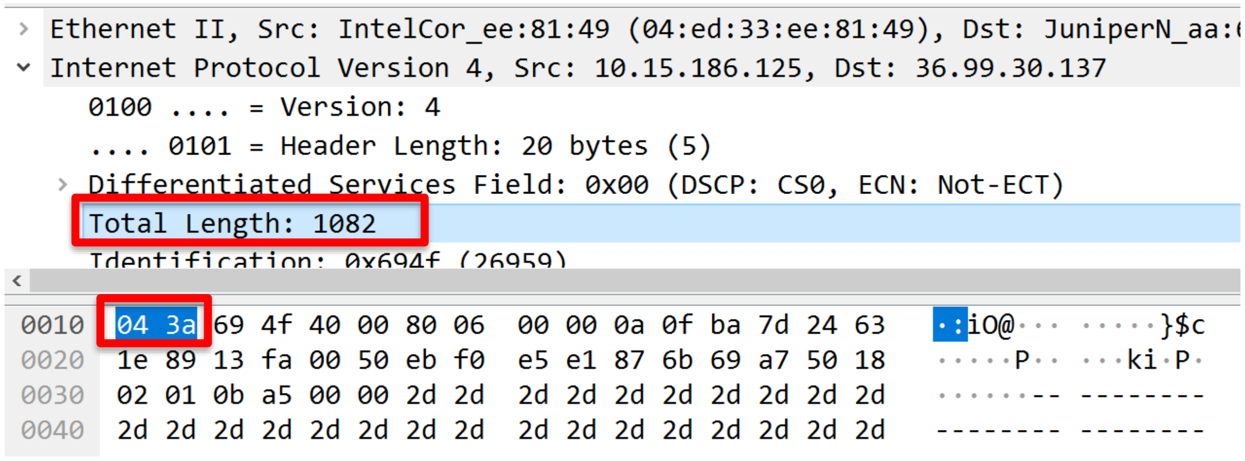
- Default Coloring
  - Gray: TCP packets
  - Black with red letters: TCP Packets with errors
  - Green: HTTP Packets
  - Light Blue: UDP Packets
  - Pale Blue: ARP Packets
  - Lavender: ICMP Packets
  - Black with green letters: ICMP Packets with errors
- Colorings can be changed under View -> Coloring Rules

Packet Details Pane

- Display detailed information about the currently selected packet.
- All packet layers are displayed in the tree menu.
- Any portion of any layer can be exported via a right click and selecting Export Selected Packet Bytes.

Packet Bytes Pane

- Displays the raw packet bytes.
- The selected packet layer is highlighted.
- Network byte order verification
  - The high byte data is at the low address.
  - The low byte data is at the high address.
  - The large end mode.



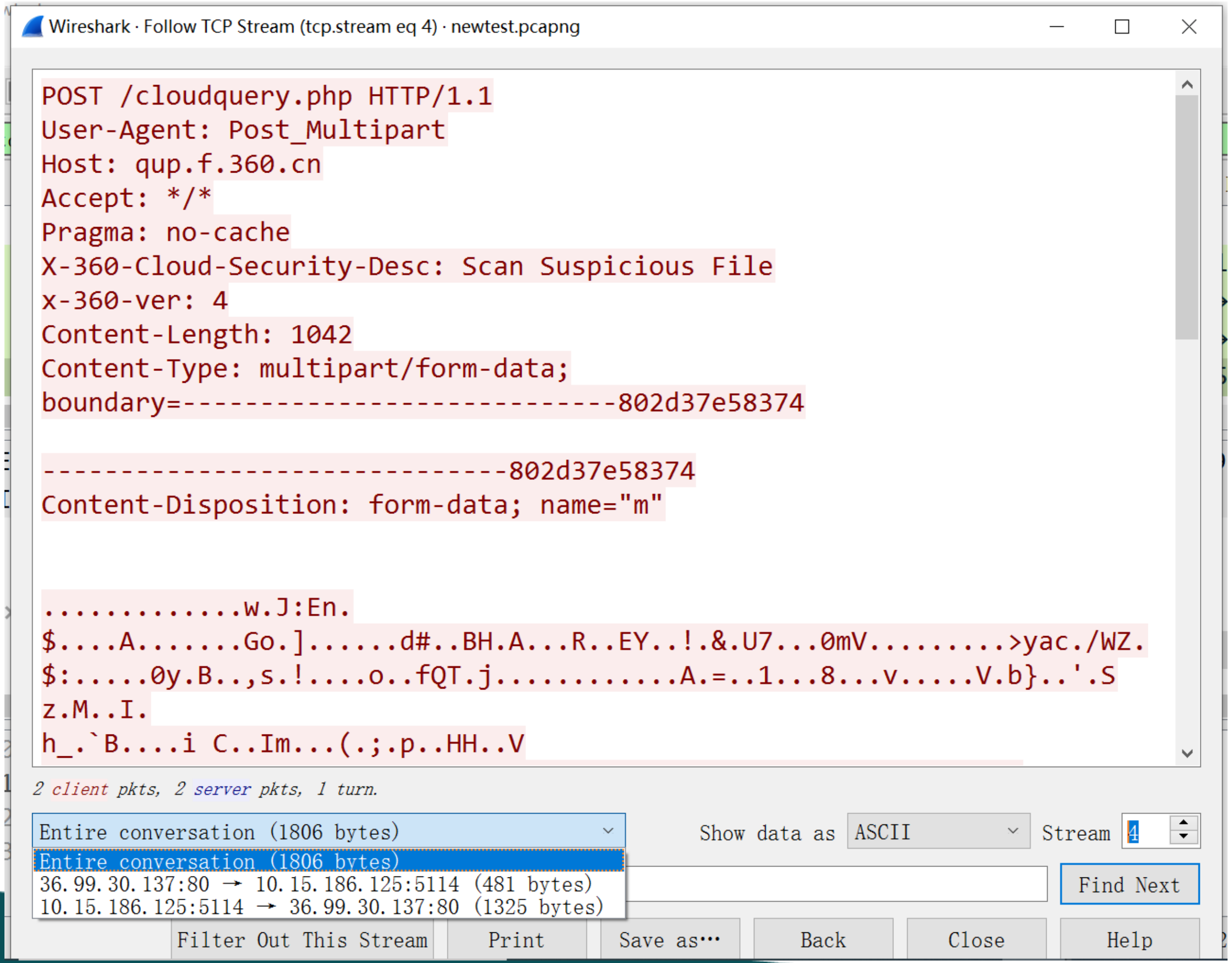
```
> Ethernet II, Src: IntelCor_ee:81:49 (04:ed:33:ee:81:49), Dst: JuniperN_aa:
v Internet Protocol Version 4, Src: 10.15.186.125, Dst: 36.99.30.137
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1082
    Identification: 0x694f (26959)

0010  04 3a 69 4f 40 00 80 06  00 00 0a 0f ba 7d 24 63   ·:iO@··· ·····}$c
0020  1e 89 13 fa 00 50 eb f0  e5 e1 87 6b 69 a7 50 18   ·····P·· ···ki·P·
0030  02 01 0b a5 00 00 2d 2d  2d 2d 2d 2d 2d 2d 2d 2d   ······-- --------
0040  2d 2d 2d 2d 2d 2d 2d 2d  2d 2d 2d 2d 2d 2d 2d 2d   -------- --------
```

SUSTech Southern University of Science and Technology

Trace Analysis (1)

- Display packets belong to the same stream.
- Dialog box
  - Analyze -> Follow -> ** Stream
  - Right click the specified packet -> Follow -> **Stream
- Useful for debugging or analyzing any TCP based application layer protocol.
- Protocols supported
  - TCP, UDP, DCCP, TLS, HTTP, HTTP/2, QUIC, SIP

Trace Analysis (2)


Wireshark · Follow TCP Stream (tcp.stream eq 4) · newtest.pcapng

```
POST /cloudquery.php HTTP/1.1
User-Agent: Post_Multipart
Host: qup.f.360.cn
Accept: */*
Pragma: no-cache
X-360-Cloud-Security-Desc: Scan Suspicious File
x-360-ver: 4
Content-Length: 1042
Content-Type: multipart/form-data;
boundary=----------------------------802d37e58374

------------------------------802d37e58374
Content-Disposition: form-data; name="m"


.............w.J:En.
$....A.......Go.]......d#..BH.A...R..EY..!.&.U7...0mV.........>yac./WZ.
$:.....0y.B..,s.!....o..fQT.j...........A.=..1...8...v.....V.b}..'.S
z.M..I.
h_.`B....i C..Im...(.;.p..HH..V
```

2 client pkts, 2 server pkts, 1 turn.

Entire conversation (1806 bytes)

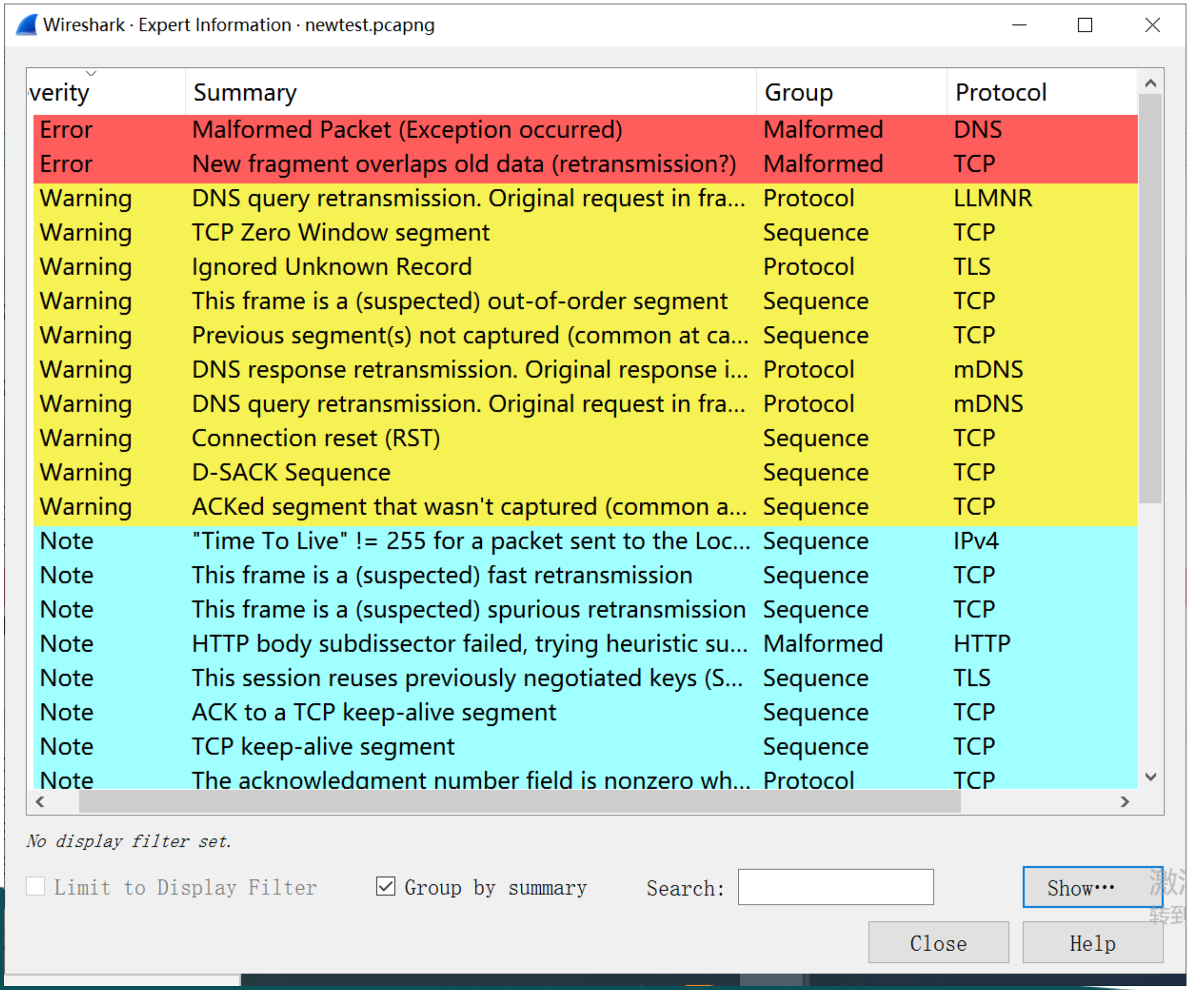
- Entire conversation (1806 bytes)
- 36.99.30.137:80 → 10.15.186.125:5114 (481 bytes)
- 10.15.186.125:5114 → 36.99.30.137:80 (1325 bytes)

Show data as ASCII   Stream 4

SUSTech Southern University of Science and Technology

Expert Information Dialog Box (1)

- Wireshark keeps track of any anomalies and other items of interest it finds in a capture file and shows them in the Expert Information dialog.
- Analysis -> Expert Infomation



Wireshark · Expert Information · newtest.pcapng

verity	Summary	Group	Protocol
Error	Malformed Packet (Exception occurred)	Malformed	DNS
Error	New fragment overlaps old data (retransmission?)	Malformed	TCP
Warning	DNS query retransmission. Original request in fra...	Protocol	LLMNR
Warning	TCP Zero Window segment	Sequence	TCP
Warning	Ignored Unknown Record	Protocol	TLS
Warning	This frame is a (suspected) out-of-order segment	Sequence	TCP
Warning	Previous segment(s) not captured (common at ca...	Sequence	TCP
Warning	DNS response retransmission. Original response i...	Protocol	mDNS
Warning	DNS query retransmission. Original request in fra...	Protocol	mDNS
Warning	Connection reset (RST)	Sequence	TCP
Warning	D-SACK Sequence	Sequence	TCP
Warning	ACKed segment that wasn't captured (common a...	Sequence	TCP
Note	"Time To Live" != 255 for a packet sent to the Loc...	Sequence	IPv4
Note	This frame is a (suspected) fast retransmission	Sequence	TCP
Note	This frame is a (suspected) spurious retransmission	Sequence	TCP
Note	HTTP body subdissector failed, trying heuristic su...	Malformed	HTTP
Note	This session reuses previously negotiated keys (S...	Sequence	TLS
Note	ACK to a TCP keep-alive segment	Sequence	TCP
Note	TCP keep-alive segment	Sequence	TCP
Note	The acknowledgment number field is nonzero wh...	Protocol	TCP

No display filter set.

☐ Limit to Display Filter ☑ Group by summary Search:

SUSTech Southern University of Science and Technology

Expert Information Dialog Box (2)

- Severity level using different colors
- Limit to display filter
  - Only show expert information items present in packets that match the current display filter.
- Group by summary
- Can be applied as display filter
- “Colorized” Protocol Details Tree

Statistics

- General statistics
  - Capture File Properties about the capture file
  - Protocol Hierarchy of the captured packets.
  - Conversations e.g. traffic between specific IP addresses.
  - Endpoints e.g. traffic to and from an IP addresses.
  - I/O Graphs visualizing the number of packets (or similar) in time.
- Protocol specific statistics
  - Service Response Time between request and response of some protocols.
  - Various other protocol specific statistics.

“Capture File Properties” Dialog

- General information about the current capture file.
- Information:
  - Details: Notable information about the capture file.
    - File: General information about the capture file.
    - Time: The timestamps of the first and the last packet in the file along with their difference.
    - Capture Information about the capture environment.
    - Interfaces Information about the capture interface or interfaces.
    - Statistics: A statistical summary of the capture file.
  - Capture file comments

"Protocol Hierarchy" Window

- This is a tree of all the protocols in the capture.
- Protocol hierarchy columns
  - Protocol: This protocol's name.
  - Percent Packets: The percentage of protocol packets relative to all packets in the capture.
  - Packets: The total number of packets of this protocol.
  - Percent Bytes
  - Bytes: The total number of bytes of this protocol.
  - Bits/s: The bandwidth of this protocol relative to the capture time.
  - End Packets: The absolute number of packets of this protocol where it was the highest protocol in the stack (last dissected).
  - End BytesEnd Bits/s: The bandwidth of this protocol relative to the capture time where was the highest protocol in the stack (last dissected).
- Useful for determining the types, amounts, and relative proportions of protocols within a trace.

“Endpoints” Window

- A network endpoint is the logical endpoint of separate protocol traffic of a specific protocol layer.
- Endpoint and Conversation types:
  - Bluetooth, Ethernet, Fibre Channel, IEEE 802.11, FDDI, IPv4, IPv6, IPX, JXTA, NCP, RSVP, SCTP, TCP, Token Ring, UDP, USB
- For each supported protocol, a tab is shown in this window.
- Each row in the list shows the statistical values for exactly one endpoint.
- Name resolution will be done if selected in the window and if it is active for the specific protocol layer.
- Limit to display filter will only show conversations matching the current display filter.
- “Endpoint Types” button lets you choose which traffic type tabs are shown.

“Conversation” Window

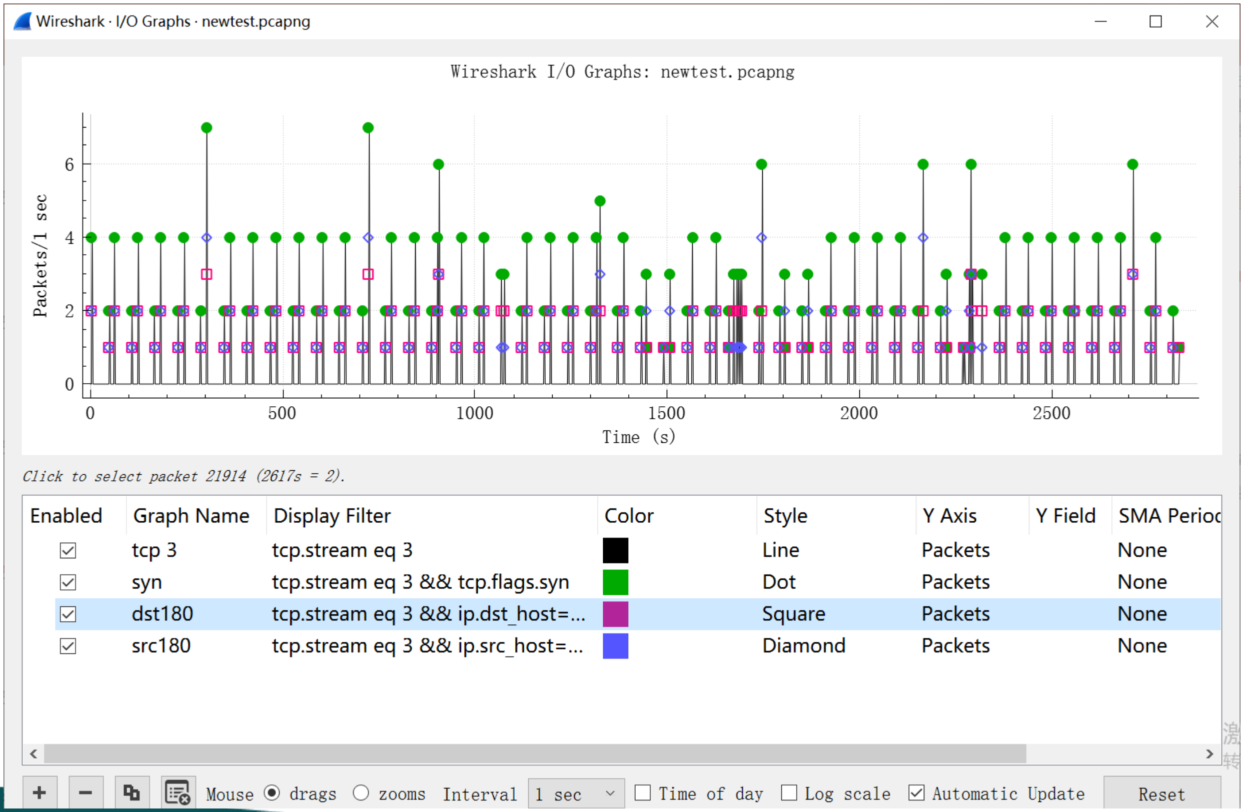
- A network conversation is the traffic between two specific endpoints.
- Each row in the list shows the statistical values for exactly one conversation.
- Compared to endpoints window, this one adds four columns:
  - Rel Start/ Abs Start: the start time of the conversation
  - the duration of the conversation in seconds
  - the average bits (not bytes) per second in each direction

“Packet Lengths” Window

- Shows the distribution of packet lengths and related information.
- Information is broken down by packet length ranges.
  - Packet Lengths
  - Count
  - Average
  - Min Val, Max Val: The minimum and maximum lengths in this range.
  - Rate (ms): The average packets per millisecond for the packets in this range.
  - Percent: The percentage of packets in this range, by count.
  - Burst Rate: Packet bursts are detected by counting the number of packets in a given time interval and comparing that count to the intervals across a window of time. Statistics for the interval with the maximum number of packets are shown. By default, bursts are detected across 5 millisecond intervals and intervals are compared across 100 millisecond windows.
  - Burst Start: The start time, in seconds from the beginning of the capture, for the interval with the maximum number of packets.

"I/O Graphs" Window

- Lets you plot packet and protocol data in a variety of ways.
- Double clicking to change the setting of columns.
- Use + and – buttons to add or remove a row.
- Do not forget to check "Enabled" list to show in the graph.



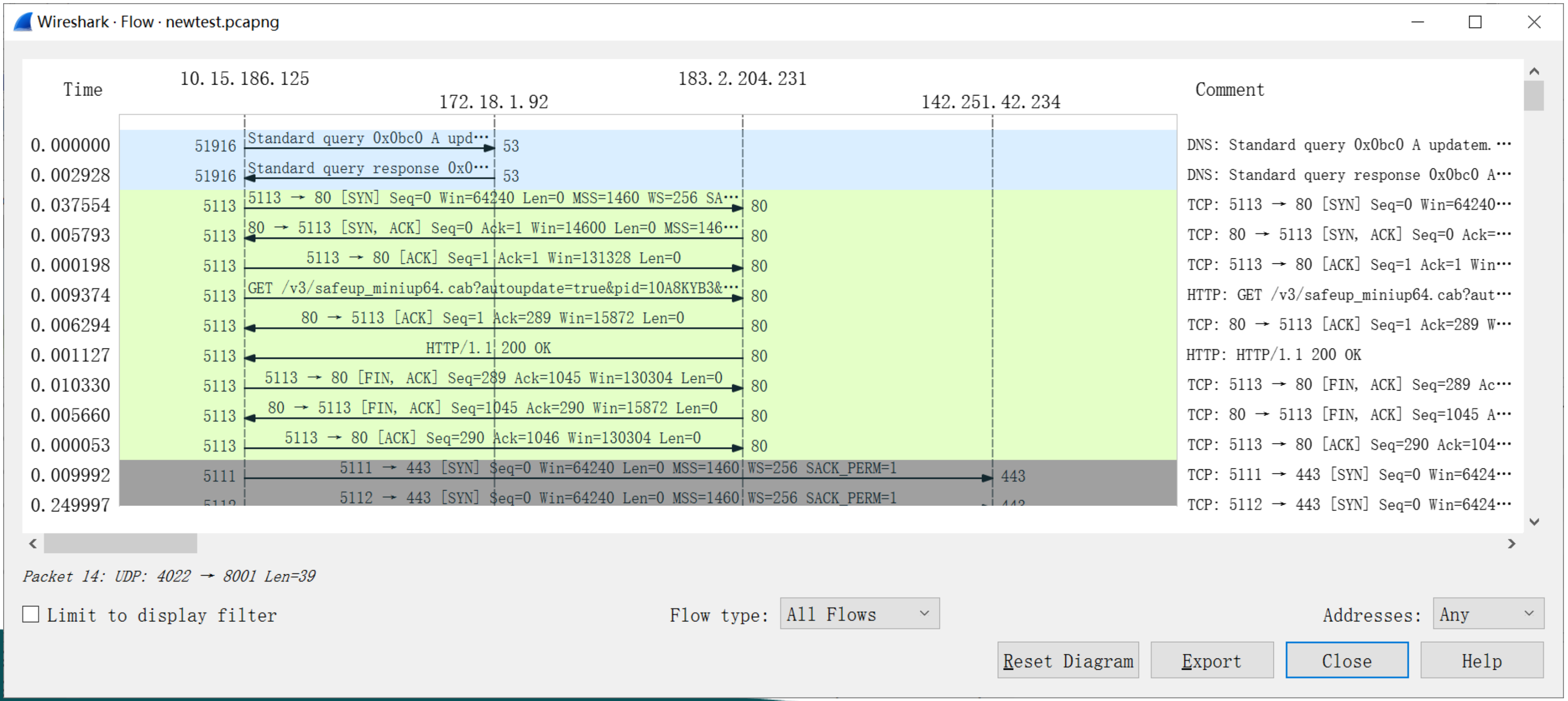
Wireshark · I/O Graphs · newtest.pcapng
Wireshark I/O Graphs: newtest.pcapng
Packets/1 sec
0
2
4
6
0
500
1000
1500
2000
2500
Time (s)

Click to select packet 21914 (2617s = 2).

Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period
✓	tcp 3	tcp.stream eq 3	⬛	Line	Packets		None
✓	syn	tcp.stream eq 3 && tcp.flags.syn	🟩	Dot	Packets		None
✓	dst180	tcp.stream eq 3 && ip.dst_host=...	🟪	Square	Packets		None
✓	src180	tcp.stream eq 3 && ip.src_host=...	🟦	Diamond	Packets		None

“Flow Graphs” Window

- Shows connections between hosts.
- Useful for understanding seq. and ack. calculations.
- Each vertical line represents the specific host.



Wireshark · Flow · newtest.pcapng

```
Time       10.15.186.125        172.18.1.92        183.2.204.231        142.251.42.234    Comment
0.000000   51916  Standard query 0x0bc0 A upd…  53                      DNS: Standard query 0x0bc0 A updatem.…
0.002928   51916  Standard query response 0x0…  53                      DNS: Standard query response 0x0bc0 A…
0.037554   5113  5113 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SA…  80      TCP: 5113 → 80 [SYN] Seq=0 Win=64240…
0.005793   5113  80 → 5113 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=146…  80      TCP: 80 → 5113 [SYN, ACK] Seq=0 Ack=…
0.000198   5113  5113 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=0  80                   TCP: 5113 → 80 [ACK] Seq=1 Ack=1 Win…
0.009374   5113  GET /v3/safeup_miniup64.cab?autoupdate=true&pid=10A8KYB3&…  80     HTTP: GET /v3/safeup_miniup64.cab?aut…
0.006294   5113  80 → 5113 [ACK] Seq=1 Ack=289 Win=15872 Len=0  80                 TCP: 80 → 5113 [ACK] Seq=1 Ack=289 W…
0.001127   5113  HTTP/1.1 200 OK  80                                            HTTP: HTTP/1.1 200 OK
0.010330   5113  5113 → 80 [FIN, ACK] Seq=289 Ack=1045 Win=130304 Len=0  80     TCP: 5113 → 80 [FIN, ACK] Seq=289 Ac…
0.005660   5113  80 → 5113 [FIN, ACK] Seq=1045 Ack=290 Win=15872 Len=0  80      TCP: 80 → 5113 [FIN, ACK] Seq=1045 A…
0.000053   5113  5113 → 80 [ACK] Seq=290 Ack=1046 Win=130304 Len=0  80          TCP: 5113 → 80 [ACK] Seq=290 Ack=104…
0.009992   5111  5111 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1  443   TCP: 5111 → 443 [SYN] Seq=0 Win=6424…
0.249997         5112 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1        TCP: 5112 → 443 [SYN] Seq=0 Win=6424…
```

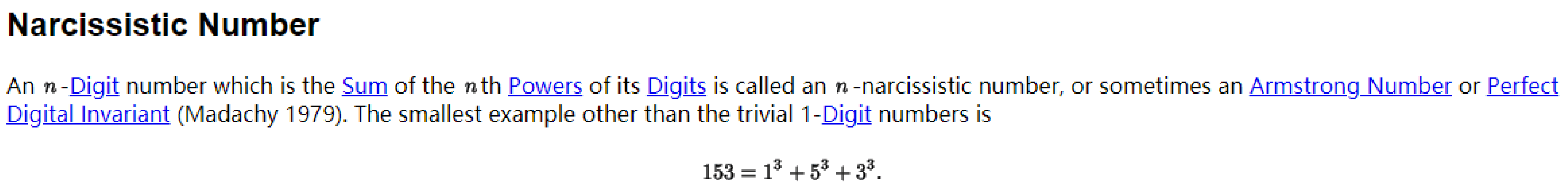
Packet 14: UDP: 4022 → 8001 Len=39

☐ Limit to display filter   Flow type: All Flows   Addresses: Any

SUSTech Southern University of Science and Technology

PRACTISE 1

- Find Narcissistic Number
  - filename: narcissistic_number.py
  - requirement:
    - implement a function to find all the narcissistic numbers in a range
  - function signature:
    - def find_narcissistic_number(start: int, end: int) -> list



Narcissistic Number

An n-Digit number which is the Sum of the nth Powers of its Digits is called an n-narcissistic number, or sometimes an Armstrong Number or Perfect Digital Invariant (Madachy 1979). The smallest example other than the trivial 1-Digit numbers is

$$153 = 1^3 + 5^3 + 3^3.$$

PRACTISE 2

Use Wireshark to capture packets and answer the questions with your screenshots:

1. Launch a http session between your host and "www.example.com"

   1-1. What's the filter used for the HTTP session between your host and "www.example.com"?

   1-2. Find a HTTP packet whose destination is your localhost in this http session, and find what are the decimal and hexdecimal representations of the src ip addr, src port, dst ip addr and dst port?

2. Launch a http session between your host and "www.baidu.com"

   2-1. Answer the question 1-2 based on the new http session between your host and "www.baidu.com"

   2-2. List the items which value is same in the answers of both question 1-2 and 2-1.

PRACTISE 3 (Optional)

Use ICMPv4 to trace route between your computer(source) and www.163.com (destination). Use a proper capture filter and display filter separately to show this session. And then answer the following questions with words and screenshots on both the execution result of command(DOS) and capture result of Wireshark:

1. How many 'time-to-live exceed' and 'echo reply' response messages are received? What's the source IP address of the 1st received 'time-to-live exceed' message, What's the source IP address of the 1st received 'echo reply' message?

2. Calculate the RTT (round-trip time) between your host and www.163.com based on the packets captured. Are they the same with RTT from command execution result?

3. Add the value of hops (between source and destination) and TTL value of ICMPv4 messages received by source (which sends ICMPv4 echo request). Is it the initial value of TTL from ICMPv4 message send by source or the ICMPv4 message send by destination? How to prove this conclusion?

SUSTech
Southern University
of Science and Technology