

Module 5

Computer Network Security

Prepared by Dr. Manal Mizher

Lecture 9

جامعة الجميع الذكية

تسعى لتكون الكيان الأول الذي يتبنى مبدأ المجتمع التعليمي
التعاوني ويقدم العلم مقابل العمل لتنمية وبناء المجتمعات.

Network programming

Course ID: CYB615



جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

1. Network Security

(Definition, Management, Types of attacks)

2. Deep packet inspection (DPI)

(How does DPI work?, Common uses of DPI, Limitations of DPI, DPI techniques, DPI vs. traditional packet filtering)

3. Injection packet-contents in Scapy

4. PyDivert

(Definition, Installation, Socket buffers, Capturing and Editing Packets examples)

■ Network security:

- Network security consists of the policies, processes and practices adopted to prevent, detect and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources.
- Network Security protects your network and data from breaches, intrusions, and other threats. This is a vast and overarching term that describes hardware and software solutions as well as processes or rules and configurations relating to network use, accessibility, and overall threat protection.



■ Security management:

- Security management for networks is different for all kinds of situations.
- A home or small office may only require basic security while large businesses may require high-maintenance and advanced software and hardware to prevent malicious attacks from hacking and spamming.
- In order to minimize susceptibility to malicious attacks from external threats to the network, corporations often employ tools that carry out network security verifications.

- Attacks can be from two categories:

- *"Passive"*, when a network intruder intercepts data traveling through the network.
- *"Active"*, in which an intruder initiates commands to deactivate (damage) the network's normal operation or to manner reconnaissance(exploration) and lateral (roundabout) movements to find and gain access to assets available via the network.





Key Differences between active & passive attacks

1. The active attack includes modification of the message. On the other hand, in passive attacks, the attacker doesn't commit any changes to the intercepted information.
2. The active attack causes a huge amount of harm to the system while the passive attack doesn't cause any harm to the system resources.
3. A passive attack is considered as a threat to data confidentiality. In contrast, an active attack is a threat to the integrity and availability of the data.
4. The attacked entity is aware of the attack in case of active attack. As against, the victim is unaware of the attack in the passive attack.
5. The active attack is accomplished by gaining the physical control over the communication link to capture and insert transmission. On the contrary, in a passive attack, the attacker just needs to observe the transmission.

■ Network passive attacks:

1. Wiretapping
2. Passive Port scanner
3. Idle scan
4. Encryption
5. Traffic analysis

- The *release of message content* can be expressed with an example, in which the sender wants to send a confidential message or email to the receiver. The sender doesn't want the contents of that message to be read by some interceptor.
- By using encryption a message could be masked in order to prevent the extraction of the information from the message, even if the message is captured. Though still attacker can analyze the traffic and observe the pattern to retrieve the information. This type of passive attack refers to as *traffic analysis*.

■ Network active attacks:

1. Virus
2. Eavesdropping
3. Data modification
4. Denial-of-service attack
5. Active Port scanner
6. DNS spoofing
7. Man in the middle
8. ARP poisoning
9. VLAN hopping
10. Smurf attack
11. Buffer overflow
12. Heap overflow
13. Format string attack
14. SQL injection
15. Phishing
16. Cross-site scripting
17. CSRF
18. Cyber-attack

- **Interruption** is known as a masquerade attack in which an unauthorized attacker tries to pose as another entity.
- **Modification** can be done using two ways replay attack and alteration. In the **replay** attack, a sequence of events or some data units is captured and resent by them. While **alteration** of the message involves some change to the original message, either one of them can cause alteration.
- **Fabrication** causes **Denial Of Service (DOS)** attacks in which attackers strive to prevent licit users from accessing some services, which they are permitted to (in simple words, the attacker gain access to the network and then locks the authorized user out).

■ Deep packet inspection (DPI):

- An advanced method of examining and managing network traffic.
- It is a form of packet filtering that locates, identifies, classifies and reroutes or blocks packets with specific data or code payloads that conventional packet filtering, which examines only packet headers, cannot detect. (*Remember QoS*)
- Usually performed as part of a firewall defense, deep packet inspection functions at the application layer of the Open Systems Interconnection (OSI) reference model.

How does deep packet inspection work?

- DPI examines the contents of packets passing through a given checkpoint and makes real-time decisions depending on what a packet contains and based on rules assigned by an enterprise, an internet service provider(ISP) or a network manager.
- Previous forms of packet filtering only looked at packet header information.
- Technological advancements have enabled DPI for **firewalls** to perform more advanced inspections so that it can check both packet headers and data.
- DPI can examine the contents of a message and identify the specific application or service that sent it. In addition, filters can be programmed to look for and reroute network traffic from a specific Internet Protocol address range or a certain online service, such as Facebook or Twitter.

1. DPI is mainly used by firewalls that include an intrusion detection system feature and by standalone (IDS)s that are intended to both detect attacks and protect the network.
2. It can be used for a good purposes as a network security tool to detect and intercept viruses, worms, spyware and other forms of malicious traffic and intrusion attempts. But it can also be used for worse activities, like eavesdropping and government -sponsored censorship.
3. DPI is also useful for network management and content policy enforcement to stop data leaks and to streamline or modify the flow of network traffic as per specific use cases. For example, a message tagged as high-priority can be routed to its destination ahead of less important or lower-priority messages or packets.

4. DPI can also be used to throttle data transfers to prevent peer-to-peer abuse and thus improve network performance.
5. Because DPI makes it possible to identify the originator or recipient of content containing specific packets, it has sparked concern among privacy advocates and opponents of net neutrality.

- It can create new vulnerabilities in the network, even as it provides protection against existing vulnerabilities. While effective against buffer overflow attacks, denial-of-service attacks and certain types of malware, DPI can also be exploited to facilitate attacks in those same categories.
 - DPI adds to the complexity and unwieldy nature of existing firewalls and other security-related software. And, to remain optimally effective, DPI requires periodic updates and revisions, which can increase the administrative burden for security teams.
 - DPI can reduce network speed and performance because it creates network bottlenecks and increases the burden on firewall processors for data decryption and inline inspection.
- Despite these limitations, many network administrators have embraced deep packet inspection technology to cope with the increase in the volume, complexity and frequency of internet-related threats.

Most organizations have either an IDS or an IPS, and many have both as part of their security information and event management framework.

	IDS	IPS
NAME	Intrusion detection system	Intrusion prevention system
DESCRIPTION	A system that monitors network traffic for suspicious activity and alerts users when such activity is discovered.	A system that monitors network traffic and alerts for suspicious activity, like an IDS, but also takes preventative action against suspicious activity.
LOCATION	A host-based intrusion detection system is installed on the client computer. A network-based intrusion detection system resides on the network.	Located between a company's firewall and the rest of its network.
USE	Warns of suspicious activity taking place, but it doesn't prevent it.	Warns of suspicious activity taking place and prevents it.
FALSE POSITIVE	IDS false positives are usually just a minor inconvenience. Although the IDS incorrectly labels legitimate traffic as malicious, it does not prevent the traffic from entering the network.	IPS false positives can be more serious. When an IPS mistakes legitimate traffic for a threat, it stops the legitimate traffic from entering the network, which could impact any part of the organization, not just the IT team.

- A false positive state is when the IDS identifies an activity as an attack but the activity is acceptable behavior. A false positive is a false alarm.
- A false negative state is the most serious and dangerous state. This is when the IDS identifies an activity as acceptable when the activity is actually an attack.

1. Pattern or signature matching
2. Protocol anomaly (*abnormality*)
3. Intrusion prevention system (IPS)

■ Pattern or signature matching:

- A firewall with IDS capability analyzes each packet against a database of known network attacks.
- It looks for specific patterns that are known to be malicious and blocks the traffic if it finds such a pattern.
- The disadvantage of this approach is that its effectiveness depends on the signatures being updated regularly. This method only works against known threats or attacks.
- As new threats are discovered daily, ongoing signature updates are critical to ensure that the firewall can detect the threats and continue to protect the network.

■ Protocol anomaly:

- The protocol anomaly method -- again, used by firewalls with an IDS -- doesn't have the inherent weakness of the pattern/signature matching method because it doesn't simply allow all content that doesn't match the signature database.
- Rather, it follows a default deny approach. The firewall determines which content/traffic should be allowed based on protocol definitions.
- Thus, unlike signature matching, this method also protects the network against unknown attacks.

■ Intrusion prevention system (IPS):

- IPS solutions can block detected attacks in real time by preventing malicious packets from being delivered based on their contents.
- Thus, if a particular packet represents a known security threat, the IPS will proactively deny network traffic based on a defined rule set.
- One drawback of IPS is that the cyber-threat database must be regularly updated with information about new threats.
- The risk of false positives is also high but can be mitigated by establishing proper baseline behaviors for network components, creating conservative policies and custom thresholds, and regularly reviewing alerts and logged incidents to improve monitoring and alerting.

- In a network, each data packet comes with a header that provides basic information about its sender, intended recipient and the time it was sent:
 - *Conventional packet filtering*: can only read this information. This is the traditional approach used by older firewalls since they were incapable of processing other types of data quickly enough to avoid adversely affecting network performance.
 - *With deep packet inspection*: firewalls can overcome those shortcomings for more comprehensive, real-time packet inspection. This enables them to extract or filter information beyond packet headers for more proactive and advanced network monitoring and protection. Within the constantly expanding cyber-threat landscape, DPI is a powerful aspect of the network security ecosystem.

- Inject contents at the BYTE LEVEL with Scapy.

- Ex1:

```
from scapy.all import *  
  
packet = IP(dst='129.132.2.21') / Raw(load= 'Everyone s Smart University')  
send(packet)
```

- EX2:

```
from scapy.all import *  
  
packet = IP(dst="127.0.0.1")/Raw("\xaa\xbb\x00\x02")  
send(packet)
```

■ Capturing and Editing Packets with PyDivert

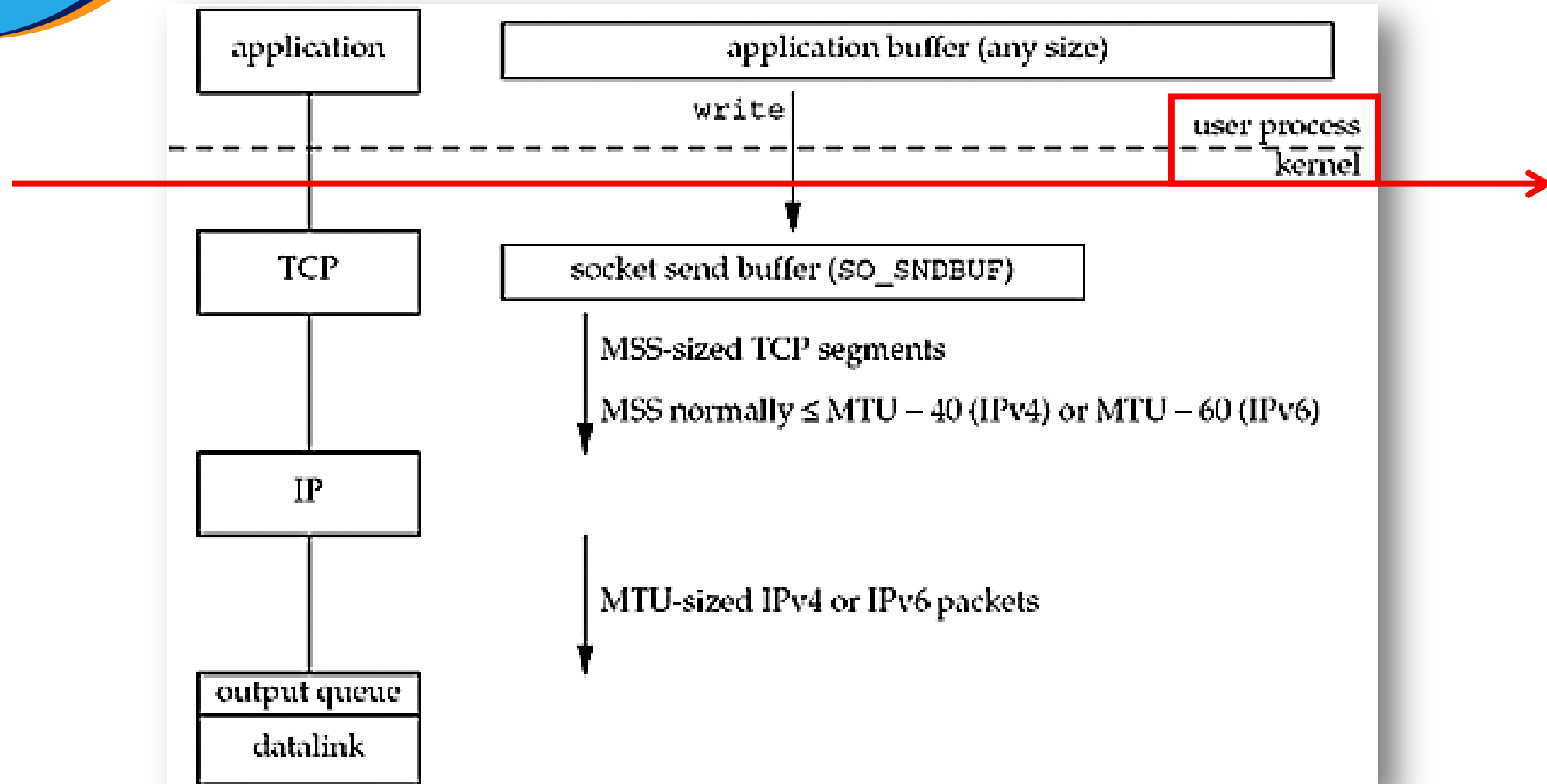
PyDivert is a python wrapper for a library called WinDivert that allows users to hook the Windows socket API (WinSock). Once hooked, the user has access to the packet send/receive buffer, and may filter or edit packets before sending them out. Below is a diagram of the packet receipt process, showing how packets are queued into the buffer at the socket layer.

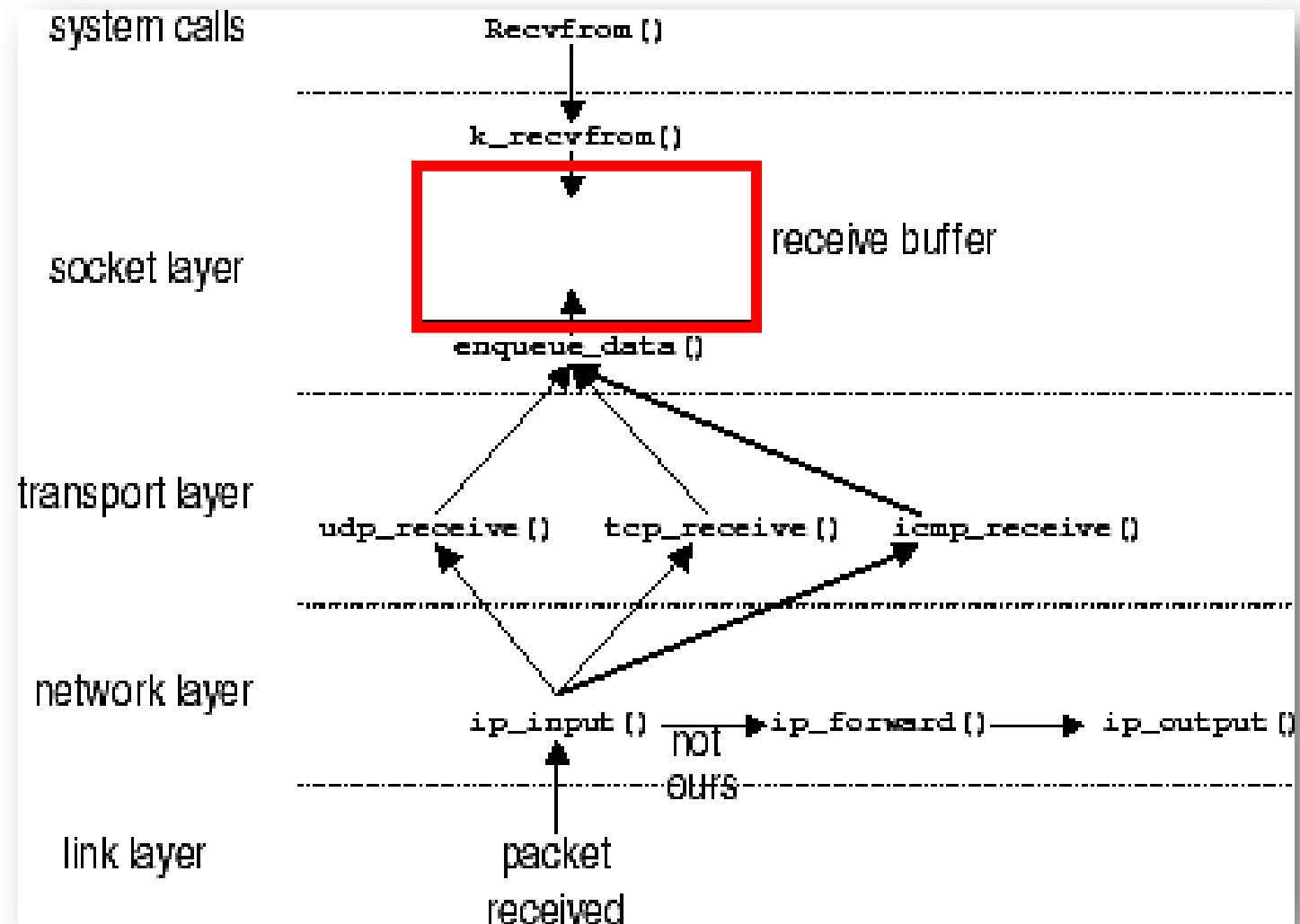
■ Socket buffers:

Are the short queues of packets the kernel holds on behalf of your app, as it's shuffling data between the **NIC** and your app's memory space.

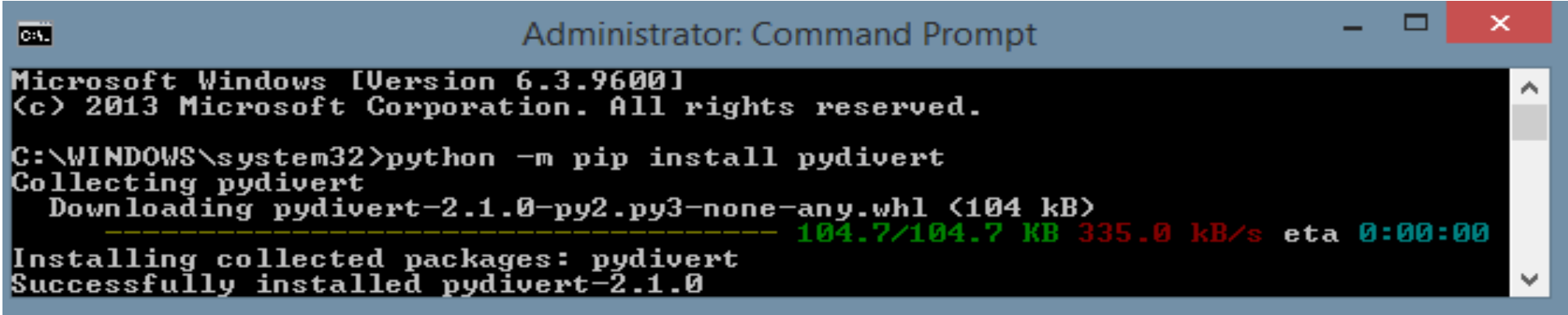


Network Interface Card





- You can install PyDiver by running: *Python -m pip install pydivert*



```
Administrator: Command Prompt

Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>python -m pip install pydivert
Collecting pydivert
  Downloading pydivert-2.1.0-py2.py3-none-any.whl (104 kB)
----- 104.7/104.7 KB 335.0 kB/s eta 0:00:00
Installing collected packages: pydivert
Successfully installed pydivert-2.1.0
```

- `class pydivert.WinDivert(filter='true', layer=<Layer.NETWORK:0>, priority=0, flags=0)`
- A WinDivert handle that can be used to capture packets.
- The main methods are `.open()` , `.recv()` , `.send()` and `.close()`.

- Here is the basic structure of a PyDivert program:

```
import pydivert

// Use WinDivert packet buffer
with pydivert.WinDivert() as buffer:
    // Iterate the buffer
    for packet in buffer:
        // Do something...
        // Send packet
        buffer.send(packet)
```

[illegible]

- **send**(*packet*, *recalculate_checksum=True*)
 - Injects a packet into the network stack. Recalculates the checksum before sending unless *recalculate_checksum=False* is passed. The return value is the number of bytes actually sent.
 - The injected packet may be one received from *recv()*, or a modified version, or a completely new packet. Injected packets can be captured and diverted again by other WinDivert handles with lower priorities.

- In sending, the network adapter actually calculates and appends the FCS as it's transmitting the frame, but the copy of the frame given to Wireshark happens from the in-memory version before it's transferred to the network adapter. (This is true with pretty much all modern NICs: the checksum is calculated in the adapter because it can use a hardware CRC generator that is way faster than software can do it.)

- `class pydivert.Packet(raw, interface, direction)`

A single packet, possibly including an IP header, a TCP/UDP header and a payload. Creation of packets is cheap, parsing is done on first attribute access.

- `recalculate_checksums(flags=0)`

(Re)calculates the checksum for any IPv4/ICMP/ICMPv6/TCP/UDP checksum present in the given packet. Individual checksum calculations may be disabled via the appropriate flag. Typically this function should be invoked on a modified packet before it is injected with `WinDivert.send()`. Returns the number of checksums calculated.

```
import pydivert

# Use WinDivert packet buffer
with pydivert.WinDivert() as buffer:
    # Iterate the buffer
    for packet in buffer:
        print('Packet BEFORE modification the buffer = ')
        print(packet.payload)
        packet.payload = bytes("Hello world!", 'utf-8')
        print('Packet AFTER modification the buffer = ')
        print(packet.payload)
    # Send packet
    buffer.send(packet)
```

```
import pydivert

with pydivert.WinDivert("outbound and tcp.PayloadLength > 0 and tcp.DstPort == 443") as w: # Try inbound
    print(w)
    for packet in w:
        print('packet BEFORE modification : ')
        print('Destination Address = ', packet.dst_addr)
        print('Destination Port = ', packet.dst_port)
        print('Payload BEFORE editing = ', packet.payload)

        # if packet.dst_port == 443:
        print("packet to the server >") # packet to the server
        packet.dst_port = 80
        packet.dst_addr = '161.69.12.13' # Rerouting to Google translator
        packet.payload = bytes("Hi, I am Here!", 'utf-8')

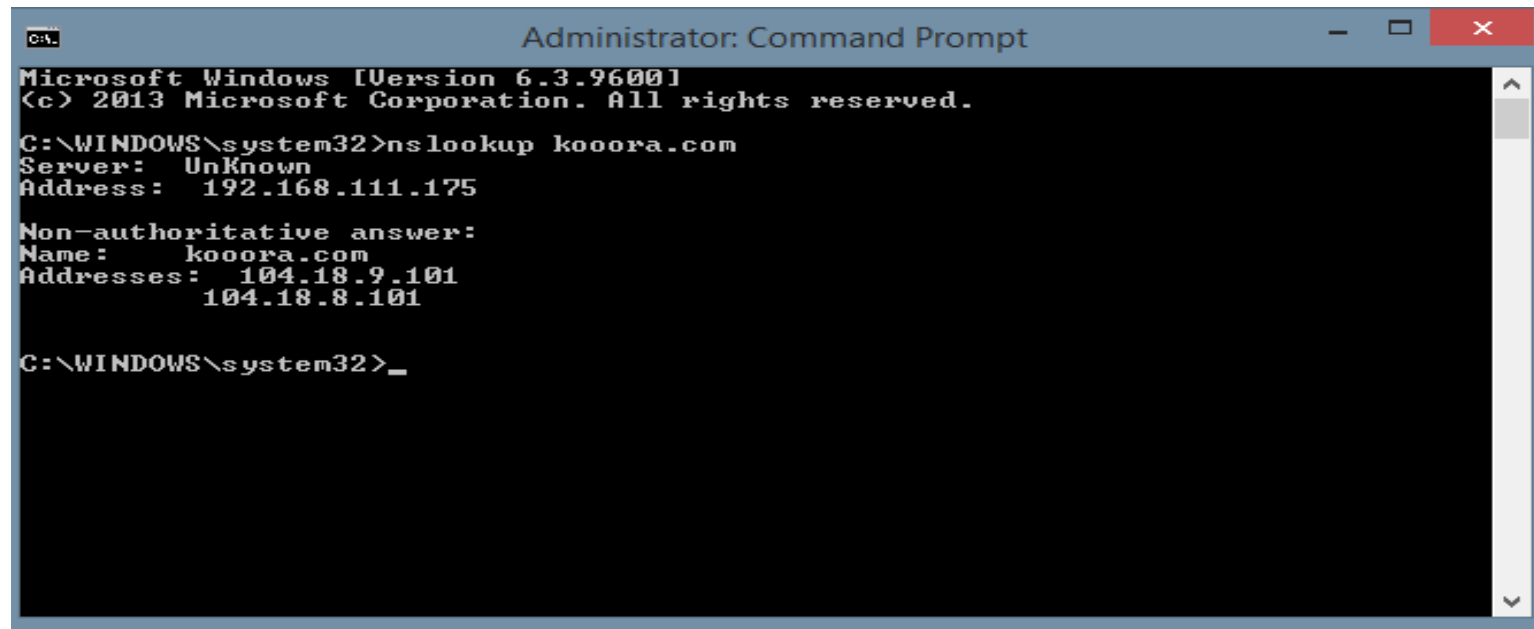
        w.send(packet)

        print('packet AFTER modification : ')
        print('Destination Address = ', packet.dst_addr)
        print('Destination Port = ', packet.dst_port)
        print('Payload AFTER editing = ', packet.payload)
        print('*****')
    # break # For one packet
```

```
for packet in packets:
    # print('packet BEFORE modification = ', packet)
    if packet.tcp:
        print('packet.tcp.payload....Before editing', packet.tcp.payload)
        packet.tcp.payload = bytes('Hello, network programming class', 'utf-8')
        print('packet.tcp.payload.... After editing', packet.tcp.payload)
        print('packet.dst_port BEFORE editing = ', packet.dst_port)
        packet.dst_port = packet.dst_port + 100
        print('packet.dst_port AFTER editing = ', packet.dst_port)
        # print("packet to the server >", packet) # packet to the server

    if bytes("class", "utf-8") in packet.tcp.payload:
        print('Found $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$')
```

- How to find the IP domain for a website?
 1. *nslookup* command in cmd
 2. By some special webpages, like: <https://www.whatismyip.com/dns-lookup/>



```
C:\WINDOWS\system32>nslookup kooora.com
Server: UnKnown
Address: 192.168.111.175

Non-authoritative answer:
Name: kooora.com
Addresses: 104.18.9.101
           104.18.8.101

C:\WINDOWS\system32>
```

PyDiver (Dropping Packets)

```
import pydivert
from pydivert import *

packets = pydivert.WinDivert()
if packets.is_open == False:
    packets.open()
pydivert.Packet(raw=1, interface='Qualcomm Atheros AR9485WB-EG Wireless Network Adapter', direction=0)
pydivert.Layer(0) # 0 for NETWORK , 1 for NETWORK_FORWARD
pydivert.Flag(1) # 1 for sniffing
for packet in packets:
    packet = packets.recv()
    print('packet.src_addr', packet.src_addr)
    print('packet.src_port = ', packet.src_port)
    print('packet.dst_addr', packet.dst_addr)
    print('packet.dst_port = ', packet.dst_port)
    if packet.dst_addr in ['104.18.9.101', '104.18.8.101']: # https://www.kooora.com/
                                                # Find Domain IP from https://www.whatismyip.com/dns-lookup/
        pydivert.Flag(2) # 2 for dropping
        print('illegal address')
        print('Dropped packet !!!!!!!!!!!!!!!')
        # elif bytes("HTTP", "utf-8") in packet.payload:
        # print('illegal content')
        # pydivert.Flag(2) # 2 for dropping
        # print('Dropped packet !!!!!!!!!!!!!!!')
    else:
        packets.send(packet)
        print('Forwarded packet !!!')
```

Q1: Find the IPv4 Address for www.bbc.com?

Q2: Then, block your computer from browsing this website? Use Dropping in pydivert for that.