

RDP MITM Setup



PyRDP RDP Interception Overview.pdf

PDF

i have three kali machines A B and C. i have rdp connection between A and B . now give completet step b y step procedure. refer the uploaded pdf



To set up and use PyRDP for intercepting RDP sessions between Kali Machine A (Server) and Kali Machine B (Client), follow these steps in detail:

Setting Up the RDP Server on Kali Machine A

1. Install xRDP:

```
bash

sudo apt update
sudo apt install xrdp -y
```

2. Start and Enable xRDP:

```
bash

sudo systemctl start xrdp
sudo systemctl enable xrdp
```

3. Check Status: Ensure it's running:

```
bash

sudo systemctl status xrdp
```

4. Allow RDP Port:

```
bash

sudo ufw allow 3389
```

5. Find Machine A IP Address:

```
bash

ip addr
```

Note the `inet` address (e.g., `192.168.1.100`).

Setting Up the RDP Client on Kali Machine B

1. Install Remmina:

```
bash

sudo apt update
sudo apt install remmina -y
sudo apt install remmina-plugin-rdp -y
```

2. Connect to the Server:

- Launch Remmina:

```
bash

remmina
```

- Configure:
 - Protocol: RDP
 - Server: `192.168.1.100` (Machine A IP)
 - Enter user credentials for Machine A.
 - Click **Connect**.
-

Setting Up Man-in-the-Middle (MITM) on Kali Machine C

1. Install Ettercap:

```
bash

sudo apt update
sudo apt install ettercap-graphical -y
```

2. Launch Ettercap GUI:

```
bash

sudo ettercap -G
```

- Select **Unified Sniffing** and your network interface (e.g., `eth0` or `wlan0`).

3. Scan for Hosts:

- Go to `Hosts → Scan for Hosts`.
- Add `Machine A` (Server) and `Machine B` (Client) to targets:
 - Target 1: `192.168.1.100`
 - Target 2: `192.168.1.101`

4. Start ARP Spoofing:

- Go to `MITM → ARP Poisoning`.
- Enable `Sniff remote connections` and click **OK**.

Setting Up PyRDP on Kali Machine C

1. Install PyRDP:

```
bash

git clone https://github.com/GoSecure/pyrdp.git
cd pyrdp
python3 -m venv env
source env/bin/activate
pip install -r requirements.txt
```

2. Generate Certificates:

```
bash

openssl req -newkey rsa:2048 -nodes -keyout pyrdp.key -x509 -days 365 -out pyrdp.crt
```

3. Start PyRDP MITM Listener:

```
bash

python3 pyrdp.py mitm --certificate pyrdp.crt --private-key pyrdp.key
```

- PyRDP will now intercept RDP traffic between Machine A and Machine B.

Interception and Analysis

1. **Monitor Traffic:** PyRDP logs RDP credentials and other session details in real time.
2. **Replay Session:**

```
bash

python3 pyrdp.py player <recording_file>
```

3. Inject Commands:

```
bash

python3 pyrdp.py inject --command "<your_command>"
```

Troubleshooting

1. Ensure IP Forwarding is Enabled on Machine C:

```
bash

echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
```

2. Verify ARP Spoofing: Use Ettercap to check if the spoofing is effective.

3. Firewall Issues: Make sure ports are open on all machines:

```
bash

sudo ufw status
```

This step-by-step procedure should enable a complete setup for RDP interception using PyRDP with MITM techniques. Let me know if you encounter issues at any step!



machine A is server rdp with ip address 192.168.1.10 and B is rdp client with ip 192.168.1.13



Given that:

- Machine A is the RDP **server** with IP: `192.168.1.10`
- Machine B is the RDP **client** with IP: `192.168.1.13`

Below is the updated, step-by-step guide to configure and intercept the RDP session between these machines using PyRDP and Ettercap on Machine C (MITM).

1. Configure RDP Server on Machine A

1. Install and Configure xRDP:

```
bash

sudo apt update
sudo apt install xrdp -y
sudo systemctl start xrdp
sudo systemctl enable xrdp
```

2. Allow RDP Port:

```
bash

sudo ufw allow 3389
```

3. Verify xRDP is Running:

```
bash

sudo systemctl status xrdp
```

4. Find and Confirm IP:

```
bash

ip addr
```

Confirm that the IP address is `192.168.1.10`.

2. Configure RDP Client on Machine B

1. Install Remmina:

```
bash

sudo apt update
sudo apt install remmina remmina-plugin-rdp -y
```

2. Connect to RDP Server:

- Launch Remmina:

```
bash

remmina
```

- Set the server to `192.168.1.10` (Machine A's IP).
- Enter the username and password of the account on Machine A.

- Click **Connect** to start the session.

3. Set Up MITM Interception on Machine C

3.1 Install Ettercap for ARP Spoofing

1. Install Ettercap:

```
bash

sudo apt update
sudo apt install ettercap-graphical -y
```

2. Enable IP Forwarding:

```
bash

echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
```

3. Launch Ettercap:

```
bash

sudo ettercap -G
```

- Select **Unified Sniffing** and choose your network interface (e.g., ``eth0`` or ``wlan0``).

4. Scan and Add Hosts:

- Go to ``Hosts → Scan for Hosts``.
- Add Machine A (``192.168.1.10``) as **Target 1**.
- Add Machine B (``192.168.1.13``) as **Target 2**.

5. Start ARP Poisoning:

- Go to ``Mitm → ARP Poisoning``.
- Enable **Sniff remote connections** and click **OK**.

4. Set Up PyRDP on Machine C

4.1 Install PyRDP

1. Clone PyRDP:

```
bash

git clone https://github.com/GoSecure/pyrdp.git
cd pyrdp
python3 -m venv env
```

```
source env/bin/activate
pip install -r requirements.txt
```

2. Generate TLS Certificates:

```
bash

openssl req -newkey rsa:2048 -nodes -keyout pyrdp.key -x509 -days 365 -out pyrdp.crt
```

4.2 Start PyRDP MITM Listener

Run PyRDP to intercept RDP traffic:

```
bash

python3 pyrdp.py mitm --certificate pyrdp.crt --private-key pyrdp.key
```

PyRDP will now log and display RDP session data as Machine B connects to Machine A.

5. Test and Monitor Interception

1. Intercepted Data:

- PyRDP will capture and display:
 - RDP credentials (username, password)
 - Session commands
 - Keystrokes
 - Screenshots

2. Record and Replay Sessions: To replay a captured session:

```
bash

python3 pyrdp.py player <recording_file>
```

3. Inject Commands: To inject commands into the intercepted session:

```
bash

python3 pyrdp.py inject --command "<your_command>"
```

6. Clean Up After Testing

1. Stop Ettercap:

- Go to `Mitm → Stop Mitm Attack` in the Ettercap GUI.

2. Disable IP Forwarding:

```
bash

echo 0 | sudo tee /proc/sys/net/ipv4/ip_forward
```

3. Deactivate PyRDP Environment:

```
bash

deactivate
```

4. Clear ARP Cache:

```
bash

sudo arp -d 192.168.1.10
sudo arp -d 192.168.1.13
```

This procedure ensures a seamless setup for intercepting and analyzing RDP traffic between Machines A and B using Machine C as the MITM. Let me know if you encounter any issues!



please mention directories also



Here is the updated step-by-step procedure for setting up the RDP server on Machine A, the RDP client on Machine B, and intercepting traffic using Machine C (MITM), with the directories where commands should be executed:

1. Configure RDP Server on Machine A (`192.168.1.10`)

1.1 Install and Configure xRDP

1. **Install xRDP:** Run the following commands from **any directory** (default: home directory `~`):

```
bash

sudo apt update
sudo apt install xrdp -y
```

2. Start and Enable xRDP:


```
bash

sudo systemctl start xrdp
sudo systemctl enable xrdp
```

3. Check Status:

```
bash

sudo systemctl status xrdp
```

4. Allow RDP Port:

```
bash

sudo ufw allow 3389
```

5. Verify the IP Address:

```
bash

ip addr
```

Note the IPv4 address (e.g., `192.168.1.10`).

2. Configure RDP Client on Machine B (`192.168.1.13`)

2.1 Install Remmina

1. **Install Remmina:** Run the following commands from **any directory** (default: `~`):

```
bash

sudo apt update
sudo apt install remmina remmina-plugin-rdp -y
```

2.2 Connect to the RDP Server

1. **Launch Remmina:**

```
bash

remmina
```

2. Set Up Connection:

- Server: ``192.168.1.10``
- Username: ``<username for Machine A>``
- Password: ``<password for Machine A>``
- Click **Connect** to start the session.

3. Set Up MITM Interception on Machine C

3.1 Install Ettercap

1. **Install Ettercap:** Run the following commands from **any directory** (default: ``~``):

```
bash

sudo apt update
sudo apt install ettercap-graphical -y
```

2. **Enable IP Forwarding:**

```
bash

echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
```

3.2 Launch Ettercap GUI

1. **Launch Ettercap:**

```
bash

sudo ettercap -G
```

2. **Set Up Unified Sniffing:**

- Select your network interface (e.g., ``eth0`` or ``wlan0``).

3. **Scan and Add Hosts:**

- Go to ``Hosts → Scan for Hosts``.
- Add:
 - Machine A (``192.168.1.10``) as **Target 1**.
 - Machine B (``192.168.1.13``) as **Target 2**.

4. **Start ARP Poisoning:**

- Go to ``Mitm → ARP Poisoning``.
- Enable **Sniff remote connections** and click **OK**.

4. Set Up PyRDP on Machine C

4.1 Clone PyRDP Repository

1. Navigate to the desired directory for PyRDP setup (e.g., `~/tools`):

```
bash

mkdir -p ~/tools
cd ~/tools
```

2. Clone PyRDP Repository:

```
bash

git clone https://github.com/GoSecure/pyrdp.git
cd pyrdp
```

4.2 Set Up PyRDP Environment

1. Create a Virtual Environment:

```
bash

python3 -m venv env
source env/bin/activate
```

2. Install Dependencies:

```
bash

pip install -r requirements.txt
```

3. Generate TLS Certificates:

```
bash

openssl req -newkey rsa:2048 -nodes -keyout pyrdp.key -x509 -days 365 -out pyrdp.crt
```

4.3 Start PyRDP MITM Listener

1. From the PyRDP directory (`~/tools/pyrdp`), run:

```
bash

python3 pyrdp.py mitm --certificate pyrdp.crt --private-key pyrdp.key
```

PyRDP will now intercept the RDP session between Machine A and Machine B.

5. Test and Monitor Interception

5.1 Intercepted Data

- PyRDP will display RDP credentials, keystrokes, and screenshots in real time.

5.2 Record and Replay Session

1. To replay a captured session:

```
bash

python3 pyrdp.py player <recording_file>
```

5.3 Inject Commands

1. To inject commands into the intercepted session:

```
bash

python3 pyrdp.py inject --command "<your_command>"
```

6. Clean Up After Testing

1. **Stop Ettercap:**

- Go to `Mitm → Stop Mitm Attack`.

2. **Disable IP Forwarding:**

```
bash

echo 0 | sudo tee /proc/sys/net/ipv4/ip_forward
```

3. **Deactivate PyRDP Environment:** From the PyRDP directory (`~/tools/pyrdp`), run:

```
bash

deactivate
```

4. **Clear ARP Cache:**

Directory Summary

This guide ensures you know the exact directory for every command, enabling a smooth workflow. Let me know if you need further assistance!

[illegible]

Email Address []:abc@gmail.com

```
└─(env)─(kaliⓀkali)-[~/pyrdp]
└─$
```



The error `python3: can't open file '/home/kali/pyrdp/pyrdp.py': [Errno 2] No such file or directory` occurs because the `pyrdp.py` file is not located in the `~/pyrdp` directory. This suggests that the correct way to start PyRDP is different from the command used.

Here's how to resolve this issue:

1. Verify the Correct Command to Start PyRDP

PyRDP does not use a single `pyrdp.py` script. Instead, it uses a command-line interface managed by `pyrdp` (installed during setup).

Run the following corrected command:

```
bash

python3 -m pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

This uses Python's module runner (`-m`) to invoke PyRDP's MITM listener.

2. Additional Steps to Verify Installation

If the above command doesn't work, verify the following:

2.1 Ensure Dependencies Are Installed

Confirm all required dependencies were installed correctly during the `pip install -r requirements.txt` step:

```
bash

pip install -r requirements.txt
```

2.2 Check PyRDP Installation

Ensure the PyRDP package is installed in your virtual environment:

```
bash

pip show pyrdp
```

If it's not installed, re-install PyRDP:

```
bash

pip install .
```

3. Correct Directory for Running PyRDP

Ensure you are in the directory where PyRDP was cloned (e.g., `~/pyrdp`) when you run the command.

4. Debugging Further

If the problem persists:

1. Check the directory contents to verify that the PyRDP repository was cloned correctly:

```
bash

ls ~/pyrdp
```

2. Ensure the virtual environment is activated:

```
bash

source env/bin/activate
```

3. Test the PyRDP installation by running:

```
bash

python3 -m pyrdp --help
```

This should display the help menu for PyRDP. If it doesn't, there may be an issue with your setup or installation.

Let me know if you encounter further issues!



```
(env)-(kali@kali)-[~/pyrdp]
└─$ pip show pyrdp
```

WARNING: Package(s) not found: pyrdp

```
(env)-(kali@kali)-[~/pyrdp]
└─$
```



The warning `Package(s) not found: pyrdp` indicates that the PyRDP package has not been properly installed in your virtual environment. Follow these steps to resolve this issue:

1. Install PyRDP Package

Ensure you are in the `~/pyrdp` directory where PyRDP was cloned and the virtual environment is activated.

1. Navigate to the PyRDP directory:

```
bash

cd ~/pyrdp
```

2. Activate the virtual environment:

```
bash

source env/bin/activate
```

3. Install PyRDP as a local package:

```
bash

pip install .
```

This installs PyRDP from the current directory into the virtual environment.

2. Verify the Installation

1. Check if PyRDP is now installed:

```
bash

pip show pyrdp
```

2. Test the PyRDP installation by running:


```
bash
```

```
python3 -m pyrdp --help
```

If the installation is successful, this will display the PyRDP help menu.

3. Run PyRDP MITM

Once the installation is verified, start the PyRDP MITM listener:

```
bash
```

```
python3 -m pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

4. Troubleshooting Further

1. **If Installation Fails:** Ensure all dependencies are installed:

```
bash
```

```
pip install -r requirements.txt
```

2. **Check for Errors During Installation:** If errors appear during the `pip install .` step, review the error messages and resolve any missing dependencies.
3. **Reclone the Repository:** If the problem persists, delete the existing directory and reclone PyRDP:

```
bash
```

```
cd ~
rm -rf pyrdp
git clone https://github.com/GoSecure/pyrdp.git
cd pyrdp
python3 -m venv env
source env/bin/activate
pip install -r requirements.txt
pip install .
```

After completing these steps, PyRDP should function correctly. Let me know if additional help is needed!



```
—(env)—(kali@kali)~[/pyrdp]
└─$ pip show pyrdp
```

WARNING: Package(s) not found: pyrdp

```

└─(env)─(kali@kali)─[~/pyrdp]
└─$ pip install .

```

Processing /home/kali/pyrdp

Installing build dependencies ... done

Getting requirements to build wheel ... done

Preparing metadata (pyproject.toml) ... done

Requirement already satisfied: appdirs<2,>=1 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (1.4.4)

Requirement already satisfied: cryptography<42,>=3.3.2 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (41.0.7)

Requirement already satisfied: namesgenerator<1,>=0.3 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (0.3)

Requirement already satisfied: progressbar2<5,>=3.20 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (4.3.2)

Requirement already satisfied: pyasn1<1,>=0 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (0.5.1)

Requirement already satisfied: pycryptodome<4,>=3.5 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (3.20.0)

Requirement already satisfied: pyopenssl<24,>=19 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (23.3.0)

Requirement already satisfied: pytz in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (2023.3.post1)

Requirement already satisfied: rsa<5,>=4 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (4.9)

Requirement already satisfied: scapy<3,>=2.4 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (2.5.0)

Requirement already satisfied: service_identity>=18 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (24.1.0)

Requirement already satisfied: twisted>=23 in ./env/lib/python3.12/site-packages (from pyrdp-mitm==2.1.1.dev0) (23.10.0)

Requirement already satisfied: cffi>=1.12 in ./env/lib/python3.12/site-packages (from cryptography<42,>=3.3.2->pyrdp-mitm==2.1.1.dev0) (1.16.0)

Requirement already satisfied: python-utils>=3.8.1 in ./env/lib/python3.12/site-packages (from progressbar2<5,>=3.20->pyrdp-mitm==2.1.1.dev0) (3.8.1)

Requirement already satisfied: attrs>=19.1.0 in ./env/lib/python3.12/site-packages (from service_identity>=18->pyrdp-mitm==2.1.1.dev0) (23.2.0)

Requirement already satisfied: pyasn1-modules in ./env/lib/python3.12/site-packages (from service_identity>=18->pyrdp-mitm==2.1.1.dev0) (0.3.0)

Requirement already satisfied: automat>=0.8.0 in ./env/lib/python3.12/site-packages (from twisted>=23->pyrdp-mitm==2.1.1.dev0) (22.10.0)

Requirement already satisfied: constantly>=15.1 in ./env/lib/python3.12/site-packages (from twisted>=23->pyrdp-mitm==2.1.1.dev0) (23.10.4)

Requirement already satisfied: hyperlink>=17.1.1 in ./env/lib/python3.12/site-packages (from twisted>=23->pyrdp-mitm==2.1.1.dev0) (21.0.0)

Requirement already satisfied: incremental>=22.10.0 in ./env/lib/python3.12/site-packages (from twisted>=23->pyrdp-mitm==2.1.1.dev0) (22.10.0)

Requirement already satisfied: typing-extensions>=4.2.0 in ./env/lib/python3.12/site-packages (from twisted>=23->pyrdp-mitm==2.1.1.dev0) (4.9.0)

Requirement already satisfied: zope-interface>=5 in ./env/lib/python3.12/site-packages (from

```
twisted>=23->pyrdp-mitm==2.1.1.dev0) (6.1)
Requirement already satisfied: six in ./env/lib/python3.12/site-packages (from automat>=0.8.0-
>twisted>=23->pyrdp-mitm==2.1.1.dev0) (1.16.0)
Requirement already satisfied: pycparser in ./env/lib/python3.12/site-packages (from cffi>=1.12-
>cryptography<42,>=3.3.2->pyrdp-mitm==2.1.1.dev0) (2.21)
Requirement already satisfied: idna>=2.5 in ./env/lib/python3.12/site-packages (from
hyperlink>=17.1.1->twisted>=23->pyrdp-mitm==2.1.1.dev0) (3.6)
Requirement already satisfied: setuptools in ./env/lib/python3.12/site-packages (from zope-
interface>=5->twisted>=23->pyrdp-mitm==2.1.1.dev0) (75.6.0)
Building wheels for collected packages: pyrdp-mitm
  Building wheel for pyrdp-mitm (pyproject.toml) ... done
  Created wheel for pyrdp-mitm: filename=pyrdp_mitm-2.1.1.dev0-cp312-cp312-linux_x86_64.whl
size=356491 sha256=0c019b09144e74eb0085d20977b420a1d3c6f7ea01f07594eb5bc647069c6cfc
  Stored in directory: /tmp/pip-ephem-wheel-cache-
gd5pq2mv/wheels/6d/5c/12/51e8d4ace01ee3951bcbd5a7d6b21b1c09d0639d74de98e16d
Successfully built pyrdp-mitm
Installing collected packages: pyrdp-mitm
Successfully installed pyrdp-mitm-2.1.1.dev0
```

```
(env)-(kali@kali)-[~/pyrdp]
└─$ pip show pyrdp
```

WARNING: Package(s) not found: pyrdp

```
(env)-(kali@kali)-[~/pyrdp]
└─$
```



The issue seems to be with the installation of PyRDP. Specifically, the package is being built and installed as ``pyrdp-mitm`` rather than ``pyrdp``. Here's how to proceed to ensure PyRDP is correctly installed and operational:

1. Verify Installed Packages

The package installed is named ``pyrdp-mitm``. To confirm this, check installed packages:

```
bash

pip list | grep pyrdp
```

You should see something like:

```
pyrdp-mitm      2.1.1.dev0
```

If this appears, PyRDP is installed but under a different name.

2. Run PyRDP

Instead of using the command `python3 -m pyrdp`, use the `pyrdp-mitm` module:

```
bash

python3 -m pyrdp_mitm mitm --certificate pyrdp.crt --private-key pyrdp.key
```

This explicitly calls the installed `pyrdp-mitm` module.

3. Verify Installation

To confirm the PyRDP installation is functional, run:

```
bash

python3 -m pyrdp_mitm --help
```

This should display PyRDP's help menu with available commands and options.

4. Clean Up (If Necessary)

If the above commands fail, clean up the current installation and try reinstalling:

1. Uninstall the current package:

```
bash

pip uninstall pyrdp-mitm
```

2. Reinstall PyRDP:

```
bash

pip install .
```

3. Verify the installation again using:

```
bash

python3 -m pyrdp_mitm --help
```

5. Debugging Further

If the issue persists, ensure that:

- You are in the virtual environment (``env``).
- Your Python version is compatible with PyRDP (Python 3.8 or higher).

Let me know if you encounter further issues!



```
pip show pyrdp
```

```
WARNING: Package(s) not found: pyrdp
```

```
(env)-(kali@kali)-[~/pyrdp]
└─$ pip list | grep pyrdp
```

```
pyrdp-mitm      2.1.1.dev0
```

```
(env)-(kali@kali)-[~/pyrdp]
└─$ python3 -m pyrdp_mitm mitm --certificate pyrdp.crt --private-key pyrdp.key
```

```
/home/kali/pyrdp/env/bin/python3: No module named pyrdp_mitm
```

```
(env)-(kali@kali)-[~/pyrdp]
└─$ python3 -m pyrdp-mitm mitm --certificate pyrdp.crt --private-key pyrdp.key
```

```
/home/kali/pyrdp/env/bin/python3: No module named pyrdp-mitm
```

```
(env)-(kali@kali)-[~/pyrdp]
└─$
```



It seems there's an issue with the module being recognized, even though it is installed. This could be due to the naming convention or an issue with how the package was installed. Here's how to troubleshoot and resolve it:

1. Verify Module Installation Path

List the installed site-packages to check where the ``pyrdp-mitm`` module is located:

```
bash
```

```
ls env/lib/python3.*/site-packages | grep pyrdp
```

If `pyrdp_mitm` or a similar name appears, the package is installed but might need adjustment in how it is called.

2. Test Running PyRDP Directly

The command-line interface might be available as a standalone script in the environment's `bin` directory. Try running:

```
bash

env/bin/pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

If this works, it indicates the module is installed correctly, but its CLI script needs to be explicitly referenced.

3. Use an Explicit Python Path

Specify the exact path to the Python interpreter in the virtual environment:

```
bash

env/bin/python3 -m pyrdp_mitm mitm --certificate pyrdp.crt --private-key pyrdp.key
```

4. Reinstall the Package (if Necessary)

If the above methods fail, clean the environment and reinstall:

1. Uninstall the Current Installation:

```
bash

pip uninstall pyrdp-mitm
```

2. Reinstall PyRDP:

```
bash

pip install .
```

3. Verify Installation: Check the module location again:

```
bash

ls env/lib/python3.*/site-packages | grep pyrdp
```

4. **Retry Running PyRDP:** Use either:

```
bash

env/bin/pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

Or:

```
bash

env/bin/python3 -m pyrdp_mitm mitm --certificate pyrdp.crt --private-key pyrdp.key
```

5. Debugging Further

If none of these steps work, verify your setup:

1. **Python Version:** Ensure you're using Python 3.8 or higher:

```
bash

python3 --version
```

2. **Environment Activation:** Confirm the virtual environment is activated:

```
bash

source env/bin/activate
```

3. **Check Installation Logs:** Reinstall and check for errors during the installation.

Let me know the outcome or any additional errors encountered!



```
└─(env)─(kali@kali)─[~/pyrdp]
└─$ ls env/lib/python3.*/site-packages | grep pyrdp

pyrdp
pyrdp_mitm-2.1.1.dev0.dist-info
```

```
(env)-(kali@kali)-[~/pyrdp]
└─$
```



The output confirms that both `pyrdp` and `pyrdp_mitm` are installed in the virtual environment's `site-packages`. However, the issue lies with how the module or command is being invoked.

1. Try Running PyRDP as a CLI Script

There should be a CLI script for PyRDP available in the virtual environment's `bin` directory. Run the following:

```
bash

env/bin/pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

2. Run PyRDP as a Python Module

If the CLI script doesn't work, explicitly call PyRDP as a Python module:

```
bash

python3 -m pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

3. Verify the `pyrdp` Module

To ensure the `pyrdp` module is functional, try running its help command:

```
bash

python3 -m pyrdp --help
```

This should display the help menu for PyRDP. If it works, retry the MITM command:

```
bash

python3 -m pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

4. Debug Path and Permissions

If none of the above work:

1. **Inspect the Module Path:** Check where the ``pyrdp`` and ``pyrdp_mitm`` modules are located:

```
bash

find env/lib/python3.*/site-packages -name "*pyrdp*"
```

2. **Verify Permissions:** Ensure all files have appropriate read and execute permissions:

```
bash

chmod -R u+rX env/lib/python3.*/site-packages/pyrdp*
```

5. Reinstallation as a Last Resort

If the above steps fail, completely uninstall and reinstall:

1. **Clean Environment:**

```
bash

deactivate
rm -rf env
python3 -m venv env
source env/bin/activate
```

2. **Reinstall PyRDP:**

```
bash

pip install -r requirements.txt
pip install .
```

3. **Retry Command:**

```
bash

env/bin/pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

Let me know the results of these steps!



```
(env)-(kali@kali)-[~/pyrdp]
└─$ python3 -m pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

/home/kali/pyrdp/env/bin/python3: No module named pyrdp.__main__; 'pyrdp' is a package and cannot be directly executed

```
(env)-(kali@kali)-[~/pyrdp]
└─$
```



The error `No module named pyrdp.__main__; 'pyrdp' is a package and cannot be directly executed` occurs because the `pyrdp` module does not include an executable `__main__.py` file and cannot be directly invoked as a script.

Here's how to resolve this and correctly run the MITM listener:

1. Run PyRDP Using the CLI Script

PyRDP provides a CLI script installed in the `env/bin` directory. To start the MITM listener, use the following command:

```
bash

env/bin/pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

This explicitly invokes the PyRDP CLI.

2. Verify CLI Installation

If the above command doesn't work, confirm that the `pyrdp` CLI script exists in the virtual environment:

```
bash

ls env/bin | grep pyrdp
```

You should see a file named `pyrdp`.

3. Debug and Reinstall if Necessary

If the `pyrdp` script is missing:

1. Uninstall Existing Package:

```
bash

pip uninstall pyrdp-mitm
```

2. Reinstall the Package:

```
bash

pip install .
```

3. Verify Installation:

```
bash

ls env/bin | grep pyrdp
```

4. Retry the MITM Command:

```
bash

env/bin/pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

4. Alternate Debugging

If the problem persists, check for logs or error messages during installation:

1. List Installed Packages:

```
bash

pip list
```

2. Check Installation Logs: Re-run the installation step and look for errors.

These steps should resolve the issue. Let me know the results!



```
python3 -m pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

```
/home/kali/pyrdp/env/bin/python3: No module named pyrdp.__main__; 'pyrdp' is a package and
cannot be directly executed
```

```
└─(env)─(kali@kali)─[~/pyrdp]
```

```
└─$ ls
CHANGELOG.adoc  README.md      pyproject.toml  requirements-slim.txt
CONTRIBUTING.md build          pyrdp          requirements.txt
Dockerfile     docker-compose.yml pyrdp.crt      setup.py
Dockerfile.slim docs          pyrdp.key      test
LICENSE        env          pyrdp_mitm.egg-info twisted
MANIFEST.in    ext          requirements-ci.txt
```



From the directory structure, it's clear that PyRDP is structured as a Python package with the main functionality handled via a CLI script, not as a directly executable module (`python3 -m pyrdp`).

Here's how to correctly run PyRDP:

1. Run the CLI Script Directly

The `pyrdp` CLI script is installed in the `env/bin` directory. Use the following command:

```
bash

env/bin/pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

This explicitly invokes the CLI script installed by the package.

2. Verify the `pyrdp` CLI Script Exists

If the above command fails, check if the `pyrdp` script exists in the virtual environment's `bin` directory:

```
bash

ls env/bin | grep pyrdp
```

You should see `pyrdp` listed. If not, reinstall the package as shown below.

3. Reinstall PyRDP if Necessary

1. Uninstall the existing installation:

```
bash

pip uninstall pyrdp-mitm
```

2. Reinstall the package from the current directory:

```
bash

pip install .
```

3. Verify that the `pyrdp` CLI script exists:

```
bash

ls env/bin | grep pyrdp
```

4. Test the Command

Once the CLI script is verified, run the MITM command again:

```
bash

env/bin/pyrdp mitm --certificate pyrdp.crt --private-key pyrdp.key
```

5. Check Permissions and Dependencies

If the problem persists:

1. Verify the Python version:

```
bash

python3 --version
```

Ensure it's Python 3.8 or higher.

2. Check for missing dependencies:

```
bash

pip install -r requirements.txt
```

3. Ensure all files in the virtual environment are readable and executable:

```
bash

chmod -R u+rX env/
```

Let me know the results or any further errors you encounter!