游戏抓包工具制作（一）—— 最简单转发代理的实现

原创 阿亮 游戏小测试 2024-04-08 19:07 广东

微信扫一扫
关注该公众号

上篇文章讲了我的游戏抓包工具的设计思路，今天这篇文章开始就逐步把它实现出来。

我们先写一个最简单的agent，绑定一个端口并对端口进行监听，以便响应客户端发过来的请求并创建与服务器的连接。这个类一开始会比较简单，只能实现协议转发，让游戏能够正常通过代理登录，后面的功能我们慢慢再完善。

先定义一个agent类，并编写它的初始化方法，我会尽量把注释写得很清楚，方便大家理解：

```
ass Agent(object):
  def __init__(self, agent_host, agent_port, server_host, server_port):
      """
      代理初始化一个实例
      :param agent_host: 代理绑定的ip
      :param agent_port: 代理绑定的端口
      :param server_host: 游戏服务器的ip
      :param server_port: 游戏服务器的端口
      """
      # 设置一个socket server，以便后续监听客户端请求
      self.socket_service = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
      # 将socket server绑定指定的ip和端口号
      self.socket_service.bind((agent_host, agent_port))
      self.socket_service.listen(5)
      # 设置客户端socket属性，先把它设置成一个空对象，后续与客户端请求过来之后再更新
      self.client_socket = None
      # 设置服务器socket属性，主动一个socket对象
      self.server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
      # 将server_socket连接到游戏服务器
      self.server_socket.connect((server_host, server_port))
      # 最代理设置一个表示存活的状态，用于在遇到多组异步的时候终止转发线程
      self.alive = True
```

然后我们开始写两个转发方法，一个是在客户端处等待客户端发送过来的协议，将其转发给服务器，另一个是在服务器处等待服务器返回的协议，并将其转发给客户端，以完成客户端与服务器之间的通信。

```
      def client_to_server(self):
          """
          转发客户端发送给服务器的包
          1.获取客户端发起的字节流
          2.通过server socket进行转发
          3.循环此步骤
          :return:
          """
          while self.alive:
              # 从客户端处接收客户端发送的协议内容
              data = self.client_socket.recv(65535)
              if data:
                  print(data)
                  # 将客户端的协议通过server_socket转发给服务器
                  self.server_socket.send(data)
              else:
                  print(f'收到了空字节流，socket连接可能断开')
                  # 收到空字节流，说明socket已断开，将alive设置成false，停止转发
                  # 还有其他的异常，后续会慢慢补充
                  self.alive = False
              time.sleep(0.01)

      def server_to_client(self):
          """
          转发服务器返回给客户端的包
          1.获取服务器返回回的字节流
          2.调用client socket，将服务器返回的字节流直接转发给客户端
          3.循环此步骤
          :return:
          """
          while self.alive:
              # 从服务器处等待返回的协议
              data = self.server_socket.recv(65535)
              if data:
                  print(data)
                  # 将服务器返回的协议通过client_socket转发给客户端
                  self.client_socket.send(data)
              else:
                  print(f'收到了空字节流，socket连接已经断开')
                  self.alive = False
              time.sleep(0.01)
```

由于socket的接收方法是阻塞式的，在没收到协议的时候会一直等待，而我们有两个socket转发方法，其中一个socket等待的时候，无法让另一个socket工作，这样无法做到客户端与服务器之间的实时通信。所以我们启动两个线程，让协议转发通过线程的方式来执行。又因为协议的转发需要在客户端连接过来的同时就要开始，所以我们把它们写在agent的start方法里。

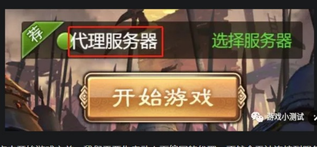
```
  start(self):
      """
      代理启动方法
      :return:
      """
      # 等待客户端连接
      self.client_socket, addr = self.socket_service.accept()
      # 启动转发协议，这里用的是Timer定时器，它也是破解自thread线程类，所以会单独起两个线程
      Timer(0, self.client_to_server).start()
      Timer(0, self.server_to_client).start()
```

最后就是给agent编写一个stop方法，让agent主动停止代理工作。虽然在最简单的模型里面可能不需要stop方法，但后面会用到，所以先简单的写一下。

```
      def stop(self):
          """
          代理结束方法
          :return:
          """
          # 这代理停止运行时，关闭客户端，服务器与代理之间的socket连接
          self.server_socket.shutdown(2)
          self.server_socket.close()
          self.socket_service.shutdown(2)
          self.socket_service.close()
```

这样一个简单的转发代理就完成了，接下来我们试一下它能不能正常工作。

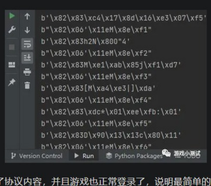
因为需要客户端主动连接到代理这里，最简单的方法就是在客户端中配置一个指向代理的服务器，代理绑定的ip和端口是多少，客户端就配置成多少（如127.0.0.1：10000），如下图这个服务器，我配置的是127.0.0.1:10000



在点击开始游戏之前，我们需要先启动上面编写的代理，不然会无法连接到服务器的，我们写个main函数，在刚刚写的文件中运行代码来启动它（请根据自己项目的实际情况填写ip）：

```
if __name__ == '__main__':
    agent = Agent("127.0.0.1", 10000, "192.168.50.50", 8011)
    agent.start()
```

好了，这个时候再点击开始游戏按钮，观察一下是否有打印协议，以及游戏是否可以正常登录



OK，成功打印了协议内容，并且游戏也正常登录了，说明最简单的代理功能，我们已经实现了，如果你也运行成功了，首先恭喜你，你已经踏出了自编协议工具的第一步👏，后面我们会慢慢改写和完善它的功能，以便实现我们之前想要实现的那些需求，还请持续关注哦，也希望能够帮忙转发给更多有需要的游戏测试小伙伴~🤙

#测试工具 10 #抓包工具 15

测试工具 · 目录 8

〈上一篇
游戏抓包工具制作（零）—— 抓包工具的前世今生

下一篇〉
游戏抓包工具制作（二）—— 设计一个简单界面

个人观点，仅供参考

喜欢此内容的人还喜欢

游戏抓包工具制作（三）—— 协议解析方法
游戏小测试

游戏抓包工具制作（二）—— 设计一个简单界面
游戏小测试

游戏抓包工具制作（零）—— 抓包工具的前世今生
游戏小测试