

CORE 及分布式 CORE 的安装与使用

马朋飞

更新: November 26, 2020

1 CORE 简介

1.1 简介

CORE (Common Open Research Emulator) 是一个搭建虚拟网络的工具。作为仿真器，与使用抽象模型的模拟相反，CORE 建立了实时运行的计算机网络模拟环境。实时运行的仿真可以连接到物理网络和路由器。它利用 Linux 操作系统提供的工具，为运行实际的应用程序和协议提供了一个环境。CORE 通常用于网络和协议研究，演示，应用程序和平台测试，评估网络方案，安全性研究以及增加物理测试网络的规模。

主要特征：

- 高效且可扩展
- 无需修改即可运行应用程序和协议
- 拖拽操作的用户界面
- 高度可定制

1.2 结构

1.2.1 主要模块

图1中列出了 CORE 的主要模块之间的结构。

- core-daemon: core 的服务器，可以在前台或者后台作为服务运行。
- core-gui/core-pygui: core 的图形化界面。可以通过拖拽操作搭建网络以及调整部分 core 配置。其中 core-gui 为老版图形界面，core-pygui 为新版图形界面。
- coresendmsg: 用于将 TLV、API 消息发送到 core-daemon 的命令行实用程序。
- vcmd: 用于将 shell 命令发送到节点的命令行实用程序。

1.2.2 会话

CORE 可以一次创建并运行多个模拟会话，图2是在典型的 GUI 交互期间会话将转换的不同状态的概述。

2 安装

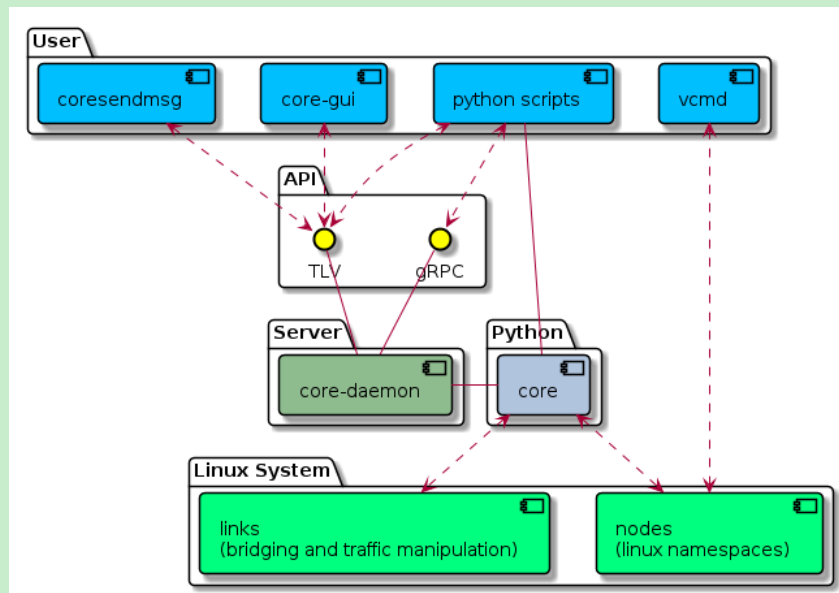


图 1: CORE 的结构

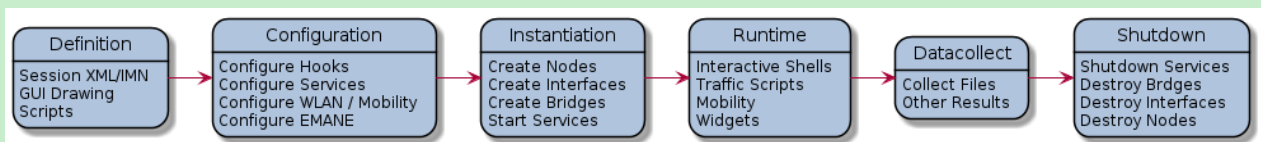


图 2: 会话状态转移过程

2.1 概览

CORE 提供了一个脚本来自动化安装依赖项，以及生成特定于 CORE 的 python 虚拟环境或构建并安装 python wheel。

注 警告： 如果安装了 Docker，那么默认的 iptables 规则将为屏蔽 CORE 的流量（可能造成分布式服务器通信问题）。

2.1.1 安装的文件

下面是自动安装过程中 CORE 将会安装的文件。

注 默认安装路径是 “/usr/local”，安装路径也可以在自动化安装中指定。

- 可执行文件
 - /bin/core-daemon, core-gui, vcmd, vnoded, etc
- tcl/tk 图形化界面文件
 - /lib/core
 - /share/core/icons
- 虚拟网络示例文件
 - /share/core/examples
- python 文件
 - poetry virtual env
 - `cd <repo>/daemon && poetry env info`
 - `/.cache/pypoetry/virtualenvs/`
 - local python install

- default install path for python3 installation of a wheel
- `python3 -c "import core; print (core.__file__)"`
- 配置文件
 - /etc/core/core.conf, logging.conf
- ospf mdr 存储库文件
 - (install.sh 路径)/../ospf-mdr
- emane 存储库文件
 - (install.sh 路径)/../emane

2.1.2 安装的可执行文件

安装完成后下列可执行文件将被安装。

名字	描述
core-cleanup	删除 CORE 创建的容器，网桥和目录的工具
core-cli	使用 gRPC 查询，打开 xml 文件和发送命令的工具
core-daemon	运行提供 TLV 和 gRPC API 的后台核心服务器
core-imn-to-xml	自动将.imn 文件转换为.xml 格式的工具
core-manage	添加，删除或检查服务，模型和节点类型的工具
core-pygui	运行新版基于 python/tk 的图形化界面
core-python	运行 CORE 的虚拟 python 环境
core-route-monitor	该工具可帮助监视跨节点的流量并将其馈送到 SDT
core-service-update	更新自动修改旧版服务以匹配当前命名的工具
coresendmsg	从命令行发送 TLV API 命令的工具

表 1: 可执行文件目录

2.1.3 支持的 Linux 版本

已验证的 Linux 版本:

- Ubuntu-18.04, 20.04
- CentOS-7.8, 8.0*

- 注
1. Ubuntu 20.04 需要安装旧版 ebttables 以实现 WLAN 功能
 2. CentOS 8 不提供旧的 ebttables 支持，WLAN 将无法正常运行
 3. CentOS 8 默认没有 netem 内核 mod

CentOS 安装 netem:

```

1 sudo yum update
2 #restart into updated kernel
3 sudo yum install -y kernel-modules-extra
4 sudo modprobe sch_netem

```

2.2 自动化安装

自动化安装过程：

```
1 #克隆CORE存储库
2 git clone https://github.com/coreemu/core.git
3 cd core
4
5 #自动安装脚本参数: install.sh [-v] [-d] [-l] [-p <prefix>]
6 #
7 # -v enable verbose install
8 # -d enable developer install
9 # -l 启用本地安装, not compatible with developer install
10 # -p 安装路径, 默认为 /usr/local
11
12 # 将CORE安装到虚拟环境 (我采用的安装方式)
13 ./install.sh -p <prefix>
14
15 # 将CORE安装到本地环境
16 ./install.sh -p <prefix> -l
```

自动化安装过程将安装以下内容：

- 安装必须的基本工具：python3, pip, pipx, invoke, poetry
- 安装 build core 的系统依赖
- clone/build/install [OPSF MDR](#)
- 根据-p参数将 CORE 安装到 poetry 管理的虚拟环境或者本地
- 根据安装类型安装指向适当 python 位置的脚本
- 根据安装类型安装指向适当 python 位置的 systemd 服务

注 1. 在本地安装可能会导致与系统软件包管理器安装的 python 依赖项潜在的依赖项冲突
2. 如果默认前缀/usr/local 无效，则提供以 sudo 运行时将在路径上找到的前缀

2.3 安装 EMANE

```
1 #<CORE_REPO>即安装CORE过程中的/core/文件夹
2 #此处建议重新开启一个终端进行操作 (上面安装完后会添加环境变量, 这一改变不会再已有的
   终端中生效, 会比较麻烦)
3 cd <CORE_REPO>
4
5 #安装到虚拟环境 (我采用的方式)
6 inv install-eman
7
8 #安装到本地
9 inv install-eman -l
```

3 使用 CORE

启动 CORE 分两部分，分别为 CORE 服务器（core-daemon）和 CORE 图形化界面（core-gui 或 core-pygui）。

3.1 core-daemon

启动 core-daemon 可以分为两种方式，一种是后台作为服务启动，一种是前台启动（终端中会实时输出日志）。

以服务形式启动 core-daemon，可以在终端输出以下命令：

```
1 #systemd
2 sudo systemctl daemon-reload
3 sudo systemctl start core-daemon
4
5 #sysv
6 sudo service core-daemon start
```

建议将这些命令写入.sh 文件以方便执行。

在启动 core-daemon 的.sh 文件中写入：

```
1 #!/bin/bash
2 #systemd
3 sudo -S systemctl daemon-reload<<EOF
4 root 密 码
5 EOF
6 sudo systemctl start core-daemon
7
8 #sysv
9 sudo service core-daemon start
```

关闭 core-daemon 的文件中写入：

```
1 #!/bin/bash
2 sudo -S killall core-daemon<<EOF
3 root 密 码
4 EOF
```

前台启动 core-daemon 比较简单，只需在终端输入：

```
1 sudo core-daemon
```

3.2 CORE 的图形化界面

旧版界面：

```
1 core-gui
```

新版界面：

```
1 core-pygui
```

图形化界面的使用方法参考[CORE 图形化界面的使用指南](#)。

如果安装了 EMANE，在老版 GUI 中的无线节点中可以配置 EMANE，在新版的 GUI 中可以看到普通的无线网络和 EMANE 无线网络。

3.3 分布式 CORE

首先需要在主服务器（master server）上有已经安装好的 CORE。

注 为了方便配置分布式服务器，建议将这些服务器的 IP 地址改为固定的。

然后在 CORE 的[releases](#)中下载分布式的 CORE 安装包 (.deb for ubuntu) 并安装到每个服务器中（包括主服务器和分布式服务器）。

```
1 #在分布式CORE的保存路径执行
2 sudo dpkg -i core_distributed_版本号_amd64.deb
```

注 安装分布式 CORE 的过程中可能会因为依赖的包没有安装而报错，此时可以在终端中输入并运行：

```
1 sudo apt --fix-broken install
```

3.3.1 配置 CORE

编辑/etc/core/core.conf 文件

```
1 distributed_address = 主服务器的IP地址
```

如果安装了 EMANE，则还需要修改/etc/core/core.conf 文件中的以下内容

```
1 controlnet = 198.168.189.0/24#(即分布式服务器的子网号，此处为一个实例)
2 emane_event_generate = True
```

3.3.2 配置 SSH

分布式的 CORE 利用 SSH 在远程服务器上运行命令。远程运行命令作者推荐使用 xterm 作为命令行工具，一般来说安装分布式 CORE 时会要求安装 xterm，如果没有安装 xterm，则需自行安装 xterm：

```
1 sudo apt install xterm
```

开始配置 SSH 远程无密码登录。

在主服务器上：

```
1 #安装SSH客户端
2 sudo apt install openssh-client
3
4 #生成SSH密钥
5 ssh-keygen -o -t rsa -b 4096 -f ~/.ssh/core
6 #当系统提示输入密码时，只需按Enter键即可生成没有密码的密钥
```

接下来，您需要将密钥传输到服务器。最简单的方法是使用 `ssh-copy-id`。要执行此操作，您必须暂时允许 `root ssh` 进入服务器

在分布式服务器上：

```
1 #安装SSH服务器
2 sudo apt install openssh-server
3
4 #编辑/etc/ssh/sshd_config
5 sudo vi/vim /etc/ssh/sshd_config
6
7 #修改以下内容
8 PasswordAuthentication yes
9 PermitRootLogin yes
10
11 #重启服务器
12 sudo service ssh restart
13
14 #设置root密码
15 sudo passwd
```

在主服务器：

```
1 #将密钥传输到服务器
2 ssh-copy-id -i ~/.ssh/core root@192.168.189.102#此处替换为分布式服务器的IP地址
3 #在询问时输入服务器root密码
4
5 #测试密钥
6 ssh -i ~/.ssh/core root@192.168.189.102#同上
```

在分布式服务器上：

```
1 sudo passwd -l root
2
3 sudo vi/vim /etc/ssh/sshd_config
4
5 #修改以下内容
6 PasswordAuthentication = no
7 PermitRootLogin without-password
8
9 #重启服务
10 sudo service ssh restart
```

在主服务器：

```
1 ssh -i ~/.ssh/core root@192.168.189.102#同上，此处为测试无密码登录
2
3 sudo vi/vim /etc/fabric.yml
4
5 #写入以下内容
6 connect_kwarg:{"key_filename": "/home/user/.ssh/core"}#user替换为具体用户名。
```

3.3.3 添加分布式服务器

对于老版 UI 界面：可以在 **Session->Emulation servers** 中添加服务器，也可以编辑文件 **/.core/servers.conf** 添加服务器：

```
1 server 192.168.189.102 4038
```

在新版 UI 界面中：可以在 **Session->Servers** 中添加分布式服务器。