



天数智芯
Iluvatar CoreX

天数智芯

软件栈安装指南

版本：V4.0.1-MR

日期：2024.8.9

适用产品：智铠 50 | 智铠 100

1 声明

1.1 版权声明

版权所有。未经天数智芯书面许可，不得以任何形式或方式将本文档的任何部分复制，传播，转录或翻译成任何语言。

1.2 免责声明

天数智芯可以随时对本文档或本文档中描述的产品进行改进和/或更改。本文档包括与天数智芯产品有关的信息，作为说明典型应用的一种方式，因此，不一定提供足以进行生产设计的完整信息。对于本文档中内容的准确性或完整性，天数智芯不做任何陈述或保证。

1.3 联系方式

地址：上海闵行区陈行公路 2168 号 3 幢

电话：021-68886607

网址：www.iluvatar.com

Contents

1 声明	2
1.1 版权声明	2
1.2 免责声明	2
1.3 联系方式	2
2 软件栈安装指南	5
2.1 修订记录	5
2.2 安装概览	5
2.2.1 安装组件	5
2.2.2 安装方式	6
2.3 固件要求	6
2.4 系统要求	7
2.5 软件要求	7
2.5.1 天数智芯加速卡驱动的前置条件	8
2.5.2 天数智算软件栈的前置条件	9
2.5.3 深度学习框架的前置条件	9
2.6 开始之前	9
2.6.1 禁用系统自动更新	10
2.6.2 确认天数智芯加速卡已安装	10
2.6.3 卸载天数智算软件栈其它版本	10
2.7 安装软件栈：宿主机安装方式	11
2.7.1 (特定情况下) 准备并注册密钥对 - 宿主机安装方式	11
2.7.2 交互方式安装软件栈 - 宿主机	12
2.7.3 静默方式安装软件栈 - 宿主机	14
2.7.4 设置环境变量	15
2.7.5 安装深度学习框架	15
2.8 安装软件栈：Docker 安装方式	16
2.8.1 (特定情况下) 准备并注册密钥对 - Docker 安装方式	16
2.8.2 交互方式安装软件栈 - Docker	17
2.8.3 静默方式安装软件栈 - Docker	19
2.9 (仅适用于 x86 架构) 安装软件栈：虚拟化平台安装方式	20
2.10 更新天数智芯加速卡驱动	20
2.11 安装后目录结构	20
3 软件栈安装常见问题	22
3.1 驱动安装相关	22
3.2 框架安装相关	23
3.3 其它	24
4 附录：支持的 Linux kernel 版本	26
4.1 Mainline	26
4.2 CentOS	28
4.3 Ubuntu	31

5 商标声明

35

此文档仅供【zhangyj@skysolidiss.com.cn-天固信息安全系统（深圳）有限公司】查阅，请勿分享他人

此文档仅供【zhangyj@skysolidiss.com.cn-天固信息安全系统（深圳）有限公司】查阅，请勿分享他人

此文档仅供【zhangyj@skysolidiss.com.cn-天固信息安全系统（深圳）有限公司】查阅，请勿分享他人

此文档仅供【zhangyj@skysolidiss.com.cn-天固信息安全系统（深圳）有限公司】查阅，请勿分享他人

2 软件栈安装指南

2.1 修订记录

- COREX01-MR401-IN01-01: 2024/8/9
 - 更新“附录：支持的 Linux kernel 版本”，新增以下 Kernel：
 - * Mainline:
 - 6.6.0-060600-generic
 - 6.6.5-060605-generic
 - 6.6.10-060610-generic
 - 6.7.0-060700-generic
 - * Ubuntu:
 - 6.2.0-060200-generic

- COREX01-MR401-IN01-00: 2024/5/31

文档本次发布内容与 V4.0.0-MR 文档相比 (COREX01-MR400-IN01-00)，有以下更新：

- **固件要求** 更新本次发布适配的固件版本为 v2.0.20
- 新增 ARM 架构的适配

2.2 安装概览

天数智算软件栈支持 x86 和 ARM 等架构，本安装指南是关于在 x86 和 ARM 架构的安装说明。

2.2.1 安装组件

本次发布提供以下组件：

组件	内容
软件栈	包括天数智芯加速卡驱动、函数库、编译器、ixSMI、ixPROF、ixKN、ixSYS、ixGDB 等工具
测试工具	包括性能测试工具、矩阵算力测试工具、P2P 性能测试工具、通讯库性能测试工具和 GDS 工具等
示例脚本	常用模型推理的示例脚本
深度学习框架及相关领域库和加速库	天数智芯适配版，如 PyTorch、ONNXRuntime、IGIE 等

本文档提供软件栈和深度学习框架的安装步骤。安装天数智芯推理示例脚本请参考《推理示例模型安装指

南》。天数智芯测试工具的安装将随软件栈同时安装，测试工具的默认安装目录为 `/usr/local/corex-{v.r.m}/extras/test_demo/`。

2.2.2 安装方式

本次发布支持以下三种安装方式：

安装方式	操作步骤
宿主机安装方式	1. 使用 corex installer 安装包安装软件栈 2. (如需使用深度学习框架) 使用.whl 包安装天数智芯适配版深度学习框架 3. (推荐) 安装示例脚本
Docker 安装方式	1. 使用 corex docker installer 安装包生成天数智算软件栈对应的 Docker 镜像 2. 启动含天数智算软件栈环境的 Docker 容器 (容器中已包含软件栈和天数智芯适配版深度学习框架) 3. (推荐) 在 Docker 容器中安装示例脚本
虚拟化平台安装方式	1. 加载 mdev 驱动和 KVM 驱动 2. 使能 svGPU 特性 3. 使用 corex installer 安装包安装软件栈 4. (如需使用深度学习框架) 使用.whl 包安装天数智芯适配版深度学习框架 5. (推荐) 安装示例脚本 注：参考《天数智芯加速卡 GPU 虚拟化 (svGPU) 使用指南 (基于 KVM)》获取更多内容

2.3 固件要求

请使用以下版本：

- 智铠 50 加速卡固件 v2.0.20
- 智铠 100 加速卡固件 v2.0.20

如您的固件版本低于上述要求，安装当前版本的驱动和软件栈前，请先联系天数智芯应用工程师升级固件，以免安装失败。

下面简单介绍升级固件的操作 (通过 .run 包即可一键升级固件版本，无需您卸载当前固件)：

1. 检查当前环境中软件栈版本和固件版本，确认是否需要升级。

使用该 .run 包升级固件的前提是：已有的加速卡固件版本不低于 1.0.3。如果低于 1.0.3，则无法升级成功，需要您联系应用工程师更换加速卡。

2. 如需升级固件，先执行 **rmmod iluvatar** 命令卸载驱动。

3. 驱动卸载完成后，

a. 运行获取到的固件 .run 包 ixfw-{gpu_type}-{firmware_version}_{ARCH}.run 升级固件：

```
$ sudo bash ixfw-{gpu_type}-{firmware_version}_{ARCH}.run
```

b. 在交互界面，默认勾选所有加速卡进行固件升级，按下 Enter 键可以取消对应加速卡的固件升级。

c. 确认无误后，通过上下键或鼠标选择 **Upgrade** 并按下 Enter 键即可升级固件。

4. 固件升级完成后，根据系统提示重启当前设备：

```
Please reboot the system to valid firmware {firmware_version} !!!
```

此时，固件升级操作已完成。

2.4 系统要求

BIOS 设置：

为了能够在 PCIe BAR (Base Address Register) 空间上成功映射天数智芯加速卡的显卡内存地址，请确认 BIOS 设置中 **Above 4G Decoding** 选项已开启。

操作系统：

- x86 架构
 - CentOS 7, CentOS 8
 - Ubuntu 18.04, Ubuntu 20.04, Ubuntu 22.04
 - openEuler: 大于等于 5.1 且小于等于 5.10
- ARM 架构
 - Kylin Linux Advanced Server release V10 (Tercel)

Kernel:

x86 架构支持的 Linux kernel 请参考《附录：支持的 Linux kernel 版本》。

ARM 架构支持的 Linux kernel 有：

- 4.19.90-23.8.v2101.ky10.aarch64
- 4.19.90-23.17.v2101.ky10.aarch64
- 4.19.90-23.18.v2101.ky10.aarch64

2.5 软件要求

2.5.1 天数智芯加速卡驱动的前置条件

- GCC，且 GCC 版本需要与 Linux kernel 的版本匹配，如操作系统自带的 GCC，具体推荐如下：

Linux kernel 版本	GCC 版本	需要设置环境变量
3.x	GCC 4	无
4.0 ~ 4.5	GCC 5	export KCPPFLAGS='-fno-pie -Wno-pointer-sign -mfentry'
4.6 ~ 5.18	GCC 7/9	无
5.19 及以上	GCC 12	无

Tip

- 由于安装过程会通过动态编译生成适配您当前操作系统和 Linux kernel 的天数智芯加速卡驱动，因此依赖与 Linux kernel 版本匹配的 GCC。
- ARM 架构推荐 GCC 7。
- Linux kernel 版本相应的头文件

您可通过以下命令进行检查：

1. 检查 Linux kernel 的版本，例如：

```
$ uname -r
3.10.0-1160.42.2.el7.x86_64
```

2. 检查该 kernel 版本对应的头文件是否已安装：

- Ubuntu:

```
$ ls -l /usr/src/linux-headers-$(uname -r)
```

- CentOS/Kylin:

```
$ ls -l /usr/src/kernels/$(uname -r)
```

3. 如对应的 Linux kernel 头文件未安装，请查阅以下参考文档：

- Ubuntu: <https://www.tecmint.com/install-kernel-headers-in-ubuntu-and-debian>
- CentOS/Kylin: <https://www.tecmint.com/install-kernel-headers-in-centos-7>

- (特定情况下) 用于内核驱动签名的密钥文件

Note

该要求仅适用于 x86 架构的环境。

- 如果安装环境开启了 Secure Boot，或者开启了内核驱动签名校验，则内核在加载天数智芯加速卡驱动模块前会检查签名。如果检查签名失败，则该模块会被拒绝插入，此时会打印相应的内核日志。更多精彩内容，请参考 [SecureBoot - Debian Wiki](#)。

Tip

- * 通过注册用于内核驱动签名的密钥文件，可确保驱动文件的安全性。
 - * 如果驱动没有被签名，或者已签名但签名使用的公钥没有被注册到内核中，检查签名会失败。
 - * 您可以通过 **mokutil --test-key** 命令或查看 `/proc/keys` 来获取公钥的注册情况。
 - * 在安装软件栈之前 (开启 DKMS 的情况)，你需要准备并注册密钥对，详细操作请参考(特定情况下) 准备并注册密钥对 - 宿主机安装方式和(特定情况下) 准备并注册密钥对 - Docker 安装方式。
- 如果安装环境没有开启 Secure Boot 且没有开启内核驱动签名校验，请忽略此前置条件。

2.5.2 天数智算软件栈的前置条件

- CUDA Toolkit
 - 如果您的环境内还未安装 CUDA Toolkit，以 CUDA Toolkit 10.2 为例，可参考 [CUDA Toolkit 10.2 Download](#) 进行安装。
 - NVIDIA CUDA Toolkit 的默认的安装路径是 `/usr/local/cuda-<CUDA_VERSION>/`，并默认创建了指向 `/usr/local/cuda` 的软链接。安装完成后，您需记住该安装路径，下文使用 `<CUDA_PATH_LOCAL>` 指代该路径，您需要在天数智算软件栈安装过程中提供 CUDA Toolkit 的安装路径，允许安装脚本把所需文件复制到指定路径。
 - 安装程序默认使用 `/usr/local/cuda`，并从此拷贝所需文件。
- Python 3.x
- pip3 并保证 pip3 命令存在，如 pip3 命令不存在，您需要建立 pip3 与 pip3.x 命令的链接
- kernel-devel 和 kernel-headers，版本必须与操作系统的版本保持一致（操作系统版本可通过 **uname -a** 命令查看）
- kmod
- make
- ncurses5
 - 确保您的环境内已安装 ncurses5 包，Ubuntu 环境可执行 **apt list --installed | grep libncursesw5** 命令、CentOS/Kylin 环境可执行 **yum list installed | grep ncurses-libs** 命令检查 ncurses5 包是否存在。
 - 如不存在，Ubuntu 环境可执行 **sudo apt install libncursesw5** 命令、CentOS/Kylin 环境可执行 **sudo yum install ncurses-libs** 进行安装。
- 对于 CentOS 操作系统，请确保安装了 elfutils-libelf-devel，libelf-dev 与 libelf-devel 中的其中一个。如未安装，请执行 **yum install -y elfutils-libelf-devel** 命令
- (目前仅支持 x86 架构) 对于多机环境，如您需要使用 InfiniBand RDMA 通信，需要先安装 Mellanox InfiniBand RDMA driver，详见《PyTorch 框架功能说明》的“使用 InfiniBand 网卡 RDMA 通信”章节

2.5.3 深度学习框架的前置条件

- numpy，建议使用 numpy v1.23
- Ubuntu：libjpeg-dev 和 zlib1g-dev
- CentOS：libjpeg-turbo-devel 和 zlib-devel

2.6 开始之前

2.6.1 禁用系统自动更新

系统自动更新有可能会升级内核，内核升级后可能导致已编译的天数智芯加速卡驱动不可用，所以需要您禁用系统自动更新。

- 如您使用 Ubuntu 操作系统，您需要将 20auto-upgrades 和 10periodic 两个文件进行如下修改：

```
$ sudo vim /etc/apt/apt.conf.d/20auto-upgrades
APT::Periodic::Update-Package-Lists "0";
APT::Periodic::Unattended-Upgrade "0";

$ sudo vim /etc/apt/apt.conf.d/10periodic
APT::Periodic::Update-Package-Lists "0";
APT::Periodic::Download-Upgradeable-Packages "0";
APT::Periodic::AutocleanInterval "0";
```

- 如您使用 CentOS 或 Kylin 操作系统，您需要修改文件 /etc/yum.conf，在文件末添加下面一行代码：

```
exclude=kernel*
```

2.6.2 确认天数智芯加速卡已安装

您可执行 **lspci** 命令查看天数智芯加速卡信息，命令中 1e3e 是天数智芯对应的厂家标识（PCI vendor ID）。

例如，您得到以下输出内容，则可获知天数智芯加速卡对应于系统端口 af:00.0。

```
$ lspci -vv | grep 1e3e
af:00.0 Processing accelerators: Device 1e3e:0002
```

Note

0002 表示智铠加速卡，0001 表示天垓加速卡。

2.6.3 卸载天数智算软件栈其它版本

您需要确保其它版本的天数智算软件栈已被卸载。

- 检查环境中是否已经安装了天数智算软件栈，可查看目录 <installation-path>/corex-{v.r.m}/ 是否存在。若不存在，则当前环境中没有安装天数智算软件栈；若存在，请继续下列操作。

Note

本指南中，安装包和路径中的“{v.r.m}”表示已安装或待安装的天数智算软件栈的版本号。

- 通过以下方式查询天数智芯加速卡占用情况，您需要确保天数智芯加速卡处于空闲状态：

- 使用 ixSMI 工具查询天数智芯加速卡进程情况。

- 使用 `lsmod | grep -e 'iluvatar' -e 'bi_driver'` 命令查询天数智芯加速卡被使用情况。

Note

软件栈版本低于 3.1.1，天数智芯加速卡的驱动文件名称为“bi_driver.ko”。当您的软件栈版本更新为 3.1.1 及以上时，驱动文件名称将更新为“iluvatar.ko”。

3. 以默认路径为例，分别执行脚本卸载 Corex Driver 和 Corex Toolkit:

```
$ sudo /usr/local/corex-{v.r.m}/bin/corex-driver-uninstaller
$ sudo /usr/local/corex-{v.r.m}/bin/corex-uninstaller
```

Note

如果软件栈版本小于 2.3.0，需要使用如下命令卸载：

```
$ sudo /usr/local/corex-{v.r.m}/bin/uninstall_corex_{v.r.m}.pl
```

2.7 安装软件栈：宿主机安装方式

我们为您提供以下两种方式安装软件栈：

- 交互安装：默认安装包安装方式使用了 TUI，您可以使用方向键、回车键、ESC 键等进行跳转、上下移动
- 静默安装：当需要批量安装或不便使用 UI 时，可参考[静默方式安装软件栈 - 宿主机](#) 操作

联系您的应用工程师获取 `corex-installer-linux64-{v.r.m}_{ARCH}_10.2.run` 安装包。安装包内包括天数智芯加速卡驱动、函数库、编译器、ixSMI、ixPROF、ixKN、ixSYS、ixGDB 等工具。

Note

本指南中，安装包和路径中的“{ARCH}”表示天数智算软件栈安装包适用的架构，与您当前设备环境的架构保持一致。

Tip

安装完成后，软件栈的 `include` 目录下不包含 `cuda.h` 文件。如有需要，请您自行从 [NVIDIA 官网](#) 下载 V7.6.5 版本的 `cuda` 包，并将包中的 `cuda.h` 文件拷贝到软件栈的 `include` 目录下。

2.7.1 (特定情况下) 准备并注册密钥对 - 宿主机安装方式

Note

该操作仅适用于 x86 架构的环境。

如果您的环境开启了 Secure Boot 且您需要开启 DKMS (Dynamic Kernel Module Support)，请在安装软件栈之前参考如下步骤准备并注册密钥对：

1. 执行如需命令准备密钥对：

```
# private_key_file 是私钥文件, public_key_file 是公钥文件
$ openssl req -new -x509 -newkey rsa:2048 -days 7300 -nodes -subj "/CN=corex-installer
↳ generated signing key/" -keyout <private_key_file> -outform DER -out <public_key_file>
```

2. 将生成的密钥对放在 DKMS 的默认位置下 (DKMS 会读取密钥文件并对编译产生的内核模块进行签名):

• Ubuntu:

```
$ cp ./<private_key_file> /var/lib/shim-signed/mok/MOK.priv # 私钥存放位置
$ cp ./<public_key_file> /var/lib/shim-signed/mok/MOK.der # 公钥存放位置
```

• CentOS:

```
$ cp ./<private_key_file> /var/lib/dkms/mok.key # 私钥存放位置
$ cp ./<public_key_file> /var/lib/dkms/mok.pub # 公钥存放位置
```

3. 执行如下命令注册密钥对:

```
# 如果使用的内核大于 5.17, 或使用了 CentOS 7 及之后的版本, 请务必开启 Secure Boot, 否则内核
  ↪ 将不会从 Machine Owner Key (MOK) 中读取密钥
$ mokutil --import ./<public_key_file>
```

4. 重启设备, 根据界面提示注册密钥对。

5. 执行如下命令检查密钥对是否已被注册到内核中:

```
$ grep 'corex-installer generated signing key' /proc/keys
```

如果密钥对已经注册成功, 则在 /proc/keys 中会显示密钥的发行信息。

Note

• 如果您的环境是 Ubuntu 且对密钥没有额外的配置要求, 则以上步骤可以通过如下命令替代:

```
$ update-secureboot-policy --new-key
$ update-secureboot-policy --enroll-key
```

• 如果 DKMS 支持设置 mok_signing_key 和 mok_certificate, 则上述步骤 2 可以替换为在 /etc/dkms/framework.conf 中指定密钥位置。更多详细内容, 请参考 [DKMS 相关文档](#)。

Important

为确保信息的安全, 请您务必妥善保管您的密钥对。

2.7.2 交互方式安装软件栈 - 宿主机

执行以下命令安装软件栈:

```
$ sudo bash corex-installer-linux64-{v,r,m}_{ARCH}_10.2.run
```

安装说明:

- 同时安装天数智芯加速卡驱动和软件栈:
 - 首次安装, 对于 **Driver** 和 **Toolkit** 都按下 Enter 键。

- 使用 **modinfo bi_driver** 命令 (软件栈版本低于 3.1.1) 和 **modinfo iluvatar** 命令 (软件栈版本为 3.1.1 及以上) 查看是否安装过天数智芯加速卡驱动及对应版本。
- NVIDIA CUDA Toolkit:
 - 安装过程会与您确认 NVIDIA CUDA Toolkit 的路径, 默认路径是 `/usr/local/cuda-<CUDA_VERSION>/`。
 - 如您安装 CUDA Toolkit 时使用了自选路径, 需在运行 `.run` 安装包过程中提供相应的 `CUDA_PATH_LOCAL`。
- 手动禁用 DKMS:
 - 安装过程会提供 **Disable dkms** 选项, 默认不禁用 DKMS, 可按下 Enter 键禁用。
 - 建议不禁用 DKMS, 因为天数智芯加速卡驱动需要匹配内核, 内核更换后, DKMS 将自动构建对应内核的天数智芯加速卡驱动, 无需您重新安装。
- 安装后路径: 天数智算软件栈默认安装路径是 `/usr/local/corex-{v.r.m}/`, 并创建了 `/usr/local/corex` 链接指向安装路径。
- 提供密钥对:
 - 安装过程会提示您提供公私密钥对 **Module Secret Key** 和 **Module Public Key**。
 - 如果您开启了内核驱动签名校验, 但不需要开启 DKMS, 请提供已有的公私密钥对。如果不提供, 安装程序将自动生成公私密钥对。
 - 如果您开启了内核驱动签名校验且需要开启 DKMS, 由于已操作(特定情况下) 准备并注册密钥对 - 宿主机安装方式, 请勿提供 **Module Secret Key** 和 **Module Public Key**, 否则安装程序将禁用 DKMS。

安装完成后显示结果, 例如:

```
=====
= Summary =
=====

Driver:          Installed
For the Corex Driver uninstallation, please run command:
  sudo /usr/local/corex-{v.r.m}/bin/corex-driver-uninstaller
Logfile is /var/log/iluvatarcorex/driver_installer.log

Toolkit:         Installed at location '/usr/local/corex-{v.r.m}/'

Please make sure that:
- PATH includes /usr/local/corex-{v.r.m}/bin
- LD_LIBRARY_PATH includes /usr/local/corex-{v.r.m}/lib

For the Corex Toolkit uninstallation, please run command:
  sudo /usr/local/corex-{v.r.m}/bin/corex-uninstaller
Logfile is /var/log/iluvatarcorex/corex_installer.log
```

Note

当您同时安装了 Corex Driver 与 Corex Toolkit 才会出现上述回显结果。如果您只安装了 Corex Driver 或 Corex Toolkit, 回显结果将根据实际情况显示。

Tip

- 如果系统开启了 DKMS，安装程序会将“corex”模块注册到 DKMS，此时驱动会被安装到 `/lib/modules/$(uname -r)/updates/dkms/` 目录下，并且安装完成后可以通过 `dkms status` 命令查看，如：

```
$ dkms status
corex, {v.r.m}, 4.15.0-112-generic, {ARCH}: installed
```

此时，如果您更换了内核，DKMS 将自动构建对应内核的天数智芯加速卡驱动，无需您重新安装。

- 如果系统没有开启 DKMS 或您选择 **Disable dkms** 选项，驱动会被安装到 `/lib/modules/$(uname -r)/kernel/drivers/misc/` 目录下。

2.7.3 静默方式安装软件栈 - 宿主机

执行以下命令静默安装软件栈：

```
$ sudo bash corex-installer-linux64-{v.r.m}-{ARCH}_10.2.run --silent [--driver] [--toolkit] [--
↪ toolkit-path=<TOOLKIT_PATH>] [--no-symlink] [--disable-dkms]
```

安装说明：

- 安装天数智芯加速卡驱动和软件栈：
 - 首次安装，须键入 **--driver** 和 **--toolkit**。
 - 必须键入 **--driver** 与 **--toolkit** 中至少一项。
 - 使用 **modinfo bi_driver** 命令 (软件栈版本低于 3.1.1) 和 **modinfo iluvatar** 命令 (软件栈版本为 3.1.1 及以上) 查看是否安装过天数智芯加速卡驱动及对应版本。
- 安装后路径 (Toolkit Path)：如果不指明 **--toolkit-path**，将使用默认安装路径 `/usr/local/corex-{v.r.m}/`。
- 不创建软链接：
 - 键入 **--no-symlink** 表示不创建软链接。
 - 如果不键入此参数，安装过程将创建 `/usr/local/corex` 链接指向安装路径。
- 手动禁用 DKMS：
 - 键入 **--disable-dkms** 表示禁用 DKMS。
 - 建议不禁用 DKMS，因为天数智芯加速卡驱动需要匹配内核，内核更换后，DKMS 将自动构建对应内核的天数智芯加速卡驱动，无需您重新安装。
- 提供密钥对：
 - 指明 **--module-signing-secret-key** 和 **--module-signing-public-key** 用于提供密钥对。
 - 参数中的“<MODULE_SIGNING_SECRET_KEY>”和“<MODULE_SIGNING_PUBLIC_KEY>”分别指私钥文件路径和公钥文件路径。以私钥文件路径为例，可填私钥文件的绝对路径或相对路径：**--module-signing-secret-key =/test/mok.priv** 或 **--module-signing-secret-key =./mok.priv**。
 - 如果您开启了内核驱动签名校验，但不需要开启 DKMS，请提供已有的公私密钥对。如果不提供，安装程序将自动生成公私密钥对。
 - 如果您开启了内核驱动签名校验且需要开启 DKMS，由于已操作(特定情况下)准备并注册密钥对 - 宿主机安装方式，请勿指明 **--module-signing-secret-key** 和 **--module-signing-public-key**，否则安装程序将禁用 DKMS。

2.7.4 设置环境变量

宿主机上安装软件栈后，您需要修改 PATH 和 LD_LIBRARY_PATH 环境变量才能正常使用软件栈。以软件栈默认安装路径 `/usr/local/corex-{v.r.m}/` 为例，您需要：

- 为 PATH 环境变量加上 `/usr/local/corex-{v.r.m}/bin`
- 为 LD_LIBRARY_PATH 环境变量加上 `/usr/local/corex-{v.r.m}/lib`

Tip

如需使所有用户都能使用天数智算软件栈，请把安装路径（以 `/usr/local/corex` 为例）写到 `/etc/environment` 文件中，例如：

```
PATH="/usr/local/corex/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/
↳ games:/usr/local/games"
```

2.7.5 安装深度学习框架

联系您的应用工程师获取适用于不同版本 Python 的 .whl 包，例如：

- PyTorch v2.1.1
- torchaudio 领域库 v2.1.0
- torchvision 领域库 v0.16.0
- ONNXRuntime_gpu v1.13.1
- IGIE 推理框架 v0.9.1
- IxRT 推理引擎 v0.9.0
- IxFormer 大模型推理框架 v0.4.0
- vLLM 推理框架 v0.2.6
- IxStream 流分析工具包 v1.0.5
- cluster 加速库 v1.6.0 (支持 PyTorch)
- quiver 加速库 v0.1.0 (支持 PyTorch)
- scatter 加速库 v2.1.0 (支持 PyTorch)
- sparse 加速库 v0.6.16 (支持 PyTorch)
- FlashAttention 加速库 v2.0.1 (支持 PyTorch)

更多发布的 .whl 包，请参考《版本发布说明》。

联系您的应用工程师获取适用于不同 Python 版本的 .whl 包。例如，Python 3.10 版本的天数适配版 PyTorch .whl 包是 `torch-2.1.1+corex.{v.r.m}-cp310-cp310-linux_{ARCH}.whl`。

对您需要安装的 .whl 包逐一执行以下命令：

```
$ pip3 install <one_whl_file>
```

Note

本次发布将不再支持 Python 3.6 版本的相关验证。

2.8 安装软件栈：Docker 安装方式

我们为您提供以下两种方式安装软件栈：

- 交互安装：默认安装包安装方式使用了 TUI，您可以使用方向键、回车键、ESC 键等进行跳转、上下移动
- 静默安装：当需要批量安装或不便使用 UI 时，可参考[静默方式安装软件栈 - Docker](#) 操作

联系您的应用工程师获取 `corex-docker-installer-{v.r.m}-10.2-centos7.8.2003-py3.10-x86_64.run` 或 `corex-docker-installer-{v.r.m}-10.2-centos7.9.2009-py3.10-aarch64.run` Docker 镜像安装包。Docker 镜像安装包内包括天数智芯加速卡驱动、函数库、编译器、ixSMI、ixPROF、ixKN、ixSYS、ixGDB 等工具、深度学习框架等。

Tip

- 该安装包适用于 Ubuntu、CentOS (x86) 和 Kylin (ARM)。文件名中的 centos 是指 Docker 镜像基于 CentOS 基础镜像而创建、启动容器后的运行环境是 CentOS。
- 安装完成后，软件栈的 `include` 目录下不包含 `cuda.h` 文件。如有需要，请您自行从 [NVIDIA 官网](#) 下载 V7.6.5 版本的 `cuda` 包，并将包中的 `cuda.h` 文件拷贝到软件栈的 `include` 目录下。

2.8.1 (特定情况下) 准备并注册密钥对 - Docker 安装方式

Note

该操作仅适用于 x86 架构的环境。

如果您的环境开启了 Secure Boot 且您需要开启 DKMS (Dynamic Kernel Module Support)，请在安装软件栈之前参考如下步骤准备并注册密钥对：

1. 执行如需命令准备密钥对：

```
# private_key_file 是私钥文件, public_key_file 是公钥文件
$ openssl req -new -x509 -newkey rsa:2048 -days 7300 -nodes -subj "/CN=corex-installer"
↪ generated signing key/" -keyout <private_key_file> -outform DER -out <public_key_file>
```

2. 将生成的密钥对放在 DKMS 的默认位置下 (DKMS 会读取密钥文件并对编译产生的内核模块进行签名)：

- Ubuntu:

```
$ cp ./<private_key_file> /var/lib/shim-signed/mok/MOK.priv # 私钥存放位置
$ cp ./<public_key_file> /var/lib/shim-signed/mok/MOK.der # 公钥存放位置
```

- CentOS:

```
$ cp ./<private_key_file> /var/lib/dkms/mok.key # 私钥存放位置
$ cp ./<public_key_file> /var/lib/dkms/mok.pub # 公钥存放位置
```

3. 执行如下命令注册密钥对：

```
# 如果使用的内核大于 5.17, 或使用了 CentOS 7 及之后的版本, 请务必开启 Secure Boot, 否则内核
↪ 将不会从 Machine Owner Key (MOK) 中读取密钥
$ mokutil --import ./<public_key_file>
```


4. 重启设备，根据界面提示注册密钥对。
5. 执行如下命令检查密钥对是否已被注册到内核中：

```
$ grep 'corex-installer generated signing key' /proc/keys
```

如果密钥对已经注册成功，则在 /proc/keys 中会显示密钥的发行信息。

Note

- 如果您的环境是 Ubuntu 且对密钥没有额外的配置要求，则以上步骤可以通过如下命令替代：

```
$ update-secureboot-policy --new-key
$ update-secureboot-policy --enroll-key
```

- 如果 DKMS 支持设置 mok_signing_key 和 mok_certificate，则上述步骤 2 可以替换为在 /etc/dkms/framework.conf 中指定密钥位置。更多详细内容，请参考 [DKMS 相关文档](#)。

Important

为确保信息的安全，请您务必妥善保管您的密钥对。

2.8.2 交互方式安装软件栈 - Docker

具体安装步骤如下：

1. 运行安装包，以 x86_64 架构的安装包为例：

```
$ sudo bash corex-docker-installer-{v.r.m}-10.2-centos7.8.2003-py3.10-x86_64.run
```

- 安装天数智芯加速卡驱动和软件栈：
 - 使用 **modinfo bi_driver** 命令 (软件栈版本低于 3.1.1) 和 **modinfo iluvatar** 命令 (软件栈版本为 3.1.1 及以上) 查看是否安装过天数智芯加速卡驱动及对应版本。
 - 如果 Docker 所在的宿主机环境没有安装过天数智芯加速卡驱动，您需要选择 **Driver**。此时，驱动相关的文件 (如卸载程序，ixSMI 工具等) 都将存放在宿主机的 /usr/local/corex-{v.r.m}/ 路径下。
 - 如果 Docker 所在的宿主机已安装天数智芯加速卡驱动，则不需要选择 **Driver**。
- NVIDIA CUDA Toolkit：
 - 安装过程会与您确认 NVIDIA CUDA Toolkit 的路径，默认路径是 /usr/local/cuda-<CUDA_VERSION>/。
 - 如您安装 CUDA Toolkit 时使用了自选路径，需提供相应的 **CUDA_PATH_LOCAL**。
- 手动禁用 DKMS：
 - 安装过程会提供 **Disable dkms** 选项，默认不禁用 DKMS，可按下 Enter 键禁用。
 - 建议不禁用 DKMS，因为天数智芯加速卡驱动需要匹配内核，内核更换后，DKMS 将自动构建对应内核的天数智芯加速卡驱动，无需您重新安装。
- Docker 镜像名称：
 - 您可以在 **Options - Set Image Name** 中更改 Docker 镜像的名称。

- 默认值是 `corex:{v.r.m}`。
- 提供密钥对：
 - 安装过程会提示您提供公私密钥对 **Module Secret Key** 和 **Module Public Key**。
 - 如果您开启了内核驱动签名校验，但不需要开启 DKMS，请提供已有的公私密钥对。如果不提供，安装程序将自动生成公私密钥对。
 - 如果您开启了内核驱动签名校验且需要开启 DKMS，由于已操作(特定情况下) 准备并注册密钥对 - Docker 安装方式，请勿提供 **Module Secret Key** 和 **Module Public Key**，否则安装程序将禁用 DKMS。

Tip

- 如果系统开启了 DKMS，安装程序会将“corex”模块注册到 DKMS，此时驱动会被安装到 `/lib/modules/$(uname -r)/updates/dkms/` 目录下，并且安装完成后可以通过 `dkms status` 命令查看，如：

```
$ dkms status
corex, {v.r.m}, 4.15.0-112-generic, {ARCH}: installed
```

此时，如果您更换了内核，DKMS 将自动构建对应内核的天数智芯加速卡驱动，无需您重新安装。

- 如果系统没有开启 DKMS 或您选择 **Disable dkms** 选项，驱动会被安装到 `/lib/modules/$(uname -r)/kernel/drivers/misc/` 目录下。

安装完成后显示结果，例如：

```
Start to install the Iluvatar CoreX Driver.
Start to build image corex:{v.r.m}
It may take some minutes to build image, please wait...

Driver: Installed
For the CoreX Driver uninstallation, please run command:
    sudo /usr/local/corex-{v.r.m}/bin/corex-driver-uninstaller
Logfile is /var/log/iluvatarcorex/driver_installer.log

Docker image corex:{v.r.m} is ready, load corex container as example following:
    docker run -it -v /usr/src:/usr/src -v /lib/modules:/lib/modules -v /dev:/dev \
        --privileged --cap-add=ALL --pid=host corex:{v.r.m}
Logfile is /var/log/iluvatarcorex/docker_installer.log
```

Tip

在联网环境完成以上步骤生成 Docker 镜像 `corex:{v.r.m}` 后，您可以将该镜像包移植到离线环境启动 Docker 容器。

2. 执行以下命令启动 Docker 容器：

```
$ sudo docker run --shm-size="32g" -it -v /usr/src:/usr/src \
-v /lib/modules:/lib/modules -v /dev:/dev \
--privileged --cap-add=ALL --pid=host corex:{v.r.m}
```

如需了解 Docker 命令的更多信息,请参考 [Docker 官方文档](#)。例如,您还可使用 `--name=${container_name}` 指定 Docker 容器名称。

Tip

- `--pid=host`: 可选,此选项方便您在 Docker 容器内使用 ixSMI 工具查看进程信息。`--pid=host` 的作用是让容器的进程标识符 (PID) 与宿主机的 PID 相同,则可以查看到该容器的进程对 GPU 的使用情况。
- `--shm-size="32g"`: 可选,此选项用于设置 Docker 容器共享内存的大小,默认值是 64M。建议您结合环境资源和使用场景相应地扩大共享内存大小,如设置为 32G 或 64G,以避免使用 Docker 容器时因资源耗尽而报错。

天数智算软件栈在 Docker 容器内的安装路径是 `/usr/local/corex-{v.r.m}/`,并创建了 `/usr/local/corex` 链接指向该路径。

天数智芯适配版框架在 Docker 容器内的安装路径是 `/usr/local/corex-{v.r.m}/lib64/python3/dist-packages/`。

2.8.3 静默方式安装软件栈 - Docker

执行以下命令静默安装软件栈,以 x86_64 架构的安装包为例:

```
$ sudo bash corex-docker-installer-{v.r.m}-10.2-centos7.8.2003-py3.10-x86_64.run --silent [--  
↪ driver] [--cuda-path=<CUDA_PATH>] [--image-name=<NAME>] [--disable-dkms]
```

安装说明:

- 已内置天数智算软件栈。
- 安装天数智芯加速卡驱动:
 - 使用 **modinfo bi_driver** 命令 (软件栈版本低于 3.1.1) 和 **modinfo iluvatar** 命令 (软件栈版本为 3.1.1 及以上) 查看是否安装过天数智芯加速卡驱动及对应版本。
 - 键入 **--driver** 安装天数智芯加速卡驱动。
 - 如果您不键入 **--driver**,则需要确保 Docker 所在的宿主机已安装过天数智芯加速卡驱动。
- NVIDIA CUDA Toolkit Path: 如果您不指明 **--cuda-path**,将使用默认安装路径 `/usr/local/cuda-<CUDA_VERSION>/`。
- Docker 镜像名称: 如果您不指明 **--image-name**,将使用默认镜像名称 `corex:{v.r.m}`。
- 手动禁用 DKMS:
 - 键入 **--disable-dkms** 表示禁用 DKMS。
 - 建议您不键入此参数,因为天数智芯加速卡驱动需要匹配内核,内核更换后,DKMS 将自动构建对应内核的天数智芯加速卡驱动,无需您重新安装。
- 提供密钥对:
 - 指明 **--module-signing-secret-key** 和 **--module-signing-public-key** 用于提供密钥对。
 - 参数中的 "`<MODULE_SIGNING_SECRET_KEY>`" 和 "`<MODULE_SIGNING_PUBLIC_KEY>`" 分别指私钥文件路径和公钥文件路径。以私钥文件路径为例,可填私钥文件的绝对路径或相对路径: **--module-signing-secret-key =/test/mok.priv** 或 **--module-signing-secret-key =./mok.priv**。

- 如果您开启了内核驱动签名校验，但不需要开启 DKMS，请提供已有的公私密钥对。如果不提供，安装程序将自动生成公私密钥对。
- 如果您开启了内核驱动签名校验且需要开启 DKMS，由于已操作(特定情况下) 准备并注册密钥对 - Docker 安装方式，请勿指明 `--module-signing-secret-key` 和 `--module-signing-public-key`，否则安装程序将禁用 DKMS。

2.9 (仅适用于 x86 架构) 安装软件栈：虚拟化平台安装方式

在部署了 svGPU 方案的 KVM 虚拟化平台上安装天数智算软件栈，需要您先在宿主机上加载 mdev 和 KVM 驱动并使能 svGPU 特性。更多详细内容，请参考《天数智芯加速卡 GPU 虚拟化 (svGPU) 使用指南 (基于 KVM)》。

后续安装天数智算软件栈的操作步骤与在宿主机上安装一致，详情请参考[安装软件栈：宿主机安装方式](#)。

Note

当前仅支持在 x86 架构的环境上使用该方式安装软件栈。

2.10 更新天数智芯加速卡驱动

Linux kernel 升级后可能导致已编译的天数智芯加速卡驱动不可用，对于这种情况，您需要手动更新天数智芯加速卡驱动：

1. 参考《附录：支持的 Linux kernel 版本》，确认您的 Linux kernel 版本已通过验证。
2. 执行以下命令进入软件栈安装：

```
$ sudo bash corex-installer-linux64-{v.r.m}_{ARCH}_10.2.run
```

3. 对于 **Driver**，按下 Enter 键，对于 **Toolkit** 您可按需选择是否重新安装软件栈。

根据 Linux kernel 动态编译生成的驱动会自动覆盖旧版本的驱动。

2.11 安装后目录结构

软件栈安装完成后会得到如下目录结构：

```
.
├─ EULA.txt
├─ bin
├─ copyright
├─ examples
├─ extras
├─ include
├─ explorer
├─ lib -> lib64
├─ lib64
```

```
|─ nvvm  
|─ release-corex.txt
```

深度学习框架默认安装在 `/usr/local/lib/${python_version}/dist-packages/` 路径下。

3 软件栈安装常见问题

3.1 驱动安装相关

问题 1：安装驱动时出现提示。

问题描述

安装 4.0.0-MR 版本的驱动时，出现更新固件版本提示，例如：

```
Start to install the CoreX Driver.
For 0000:4e:00.0 Please upgrade your firmware to be >= 2.0.19 (current: drv:
↳ 4B4410F210E05831409FA2F fw:1.0.10)
```

问题分析

4.0.0-MR 版本的驱动和软件栈要求智铠加速卡固件版本为 v2.0.19 或更高版本。不匹配的固件版本会导致安装驱动时出现相关提示。

解决方法

联系天数智芯应用工程师检查固件版本。

如您安装了早前版本的软件栈，您也可通过 **ixsmi -q -d HARDWARE_MONITOR** 命令，查看“Firmware Version”。

问题 2：云平台环境下，安装驱动之后，无法查询到天数智芯加速卡。

问题描述

云平台环境下，安装驱动之后，在 `/dev/` 目录下无法查询到 iluvatar 设备 (天数智芯加速卡)。

问题分析

部分特定的云平台环境使用的是 OVMF BIOS，无法顺利分配空间给天数智芯加速卡，例如：

```
00:02.7 PCI bridge: Red Hat, Inc. QEMU PCIe Root port (prog-if 00 [Normal decode])
Flags: bus master, fast devsel, latency 0, IRQ 22
Memory at c2455000 (32-bit, non-prefetchable) [size=4K]
Bus: primary=00, secondary=08, subordinate=08, sec-latency=0
I/O behind bridge: [disabled]
Memory behind bridge: 81800000-819fffff [size=2M]
Prefetchable memory behind bridge: [disabled]
Capabilities: [54] Express Root Port (Slot+), MSI 00
Capabilities: [48] MSI-X: Enable+ Count=1 Masked-
Capabilities: [40] Subsystem: Red Hat, Inc. QEMU PCIe Root port
Capabilities: [100] Advanced Error Reporting
Capabilities: [148] Access Control Services
Kernel driver in use: pcieport
```

Figure 1: 云平台环境下无法分配到空间

使用天数智芯加速卡需要足够的显卡内存空间。

解决方法

以智铠 100 加速卡为例，在云主机 uuid.xml 文件中的 qemu 标签内，新增如下自定义参数解决问题：

```
<qemu:arg value='-fw_cfg'/>
<qemu:arg value='opt/ovmf/X-PciMmio64Mb,string=65536'/> # 这里的 65536 单位是 MB，即 64G，是
  ↳ 显存大小 (32G) 的 2 倍
```

Note

不同的加速卡，所需显卡内存空间不同，如智铠 50 加速卡需要 16G，智铠 100 加速卡需要 32G，天垓 100 加速卡需要 32G，天垓 150 加速卡需要 64G。

3.2 框架安装相关

问题 1：执行 pip3 install .whl 包时提示‘SSL’ 模块找不到。

问题描述

通过源码编译安装 Python 后，pip3 install 包时提示：(Caused by SSLError("Can't connect to HTTPS URL because the SSL module is not available."))

解决方法

1. 执行以下命令先安装 libssl 开发包：

- Ubuntu:

```
$ sudo apt-get install libssl-dev
```

- CentOS:

```
$ sudo yum install openssl-devel
```

2. 重新编译安装 Python。

问题 2：pip3 install .whl 包提示‘_ctypes’ 模块找不到。

问题描述

通过源码编译安装 python 后，pip install 包时提示: ModuleNotFoundError: No module named '_ctypes'

问题分析

Python 3.7 后 _ctypes 依赖 libffi，所以需要先安装 libffi 开发包，然后再重新编译安装 Python:

解决方法

1. 执行以下命令安装 libffi 开发包：

- Ubuntu:

```
$ sudo apt-get install libffi-dev
```

- CentOS:


```
$ sudo yum install libffi-devel
```

2. 重新编译安装 Python。

问题 3：在安装.whl 包过程中安装依赖包 pillow 时，缺少 zlib 或 jpeg 头文件。

问题描述

在安装.whl 包时到安装依赖包 pillow 这一步，系统提示缺少 zlib 或 jpeg 头文件。

解决方法

请您参考以下命令安装相关依赖库：

- Ubuntu:

```
$ apt-get install libjpeg-dev zlib1g-dev
```

- CentOS:

```
$ yum install libjpeg-turbo-devel zlib-devel
```

3.3 其它

问题 1：安装过程中提示 Perl 相关文件找不到。

问题描述

```
cp: cannot create regular file '/usr/lib/x86_64-linux-gnu/perl-base': No such file or directory
```

解决方法

环境没有安装 Perl，用以下命令进行安装：

```
$ sudo yum install perl
```

问题 2：缺少 libpython3.6m.so.1.0。

解决方法

在环境内搜索 libpython3.6m.so.1.0，找到后复制到以下路径：

- /usr/local/lib64/
- /usr/lib64/
- /usr/local/lib/
- /usr/lib/

问题 3：安装软件栈后，运行 ixSMI 工具收到驱动未安装的报错信息。

问题描述

Iluvatar- SMI has failed because it couldn't communicate with the Iluvatar driver. Make sure that the latest Iluvatar driver is installed and running.

解决方法

1. 若未开启 DKMS 或在安装软件栈时选择了 **Disable dkms** 选项，驱动会被安装到 `/lib/modules/$(uname -r)/kernel/drivers/misc/` 目录下。执行以下命令再次加载天数智芯加速卡驱动文件：

```
$ sudo insmod /lib/modules/$(uname -r)/kernel/drivers/misc/iluvatar.ko
```

2. 若开启了 DKMS，驱动会被安装到 `/lib/modules/$(uname -r)/updates/dkms/` 目录下。执行以下命令再次加载天数智芯加速卡驱动文件：

```
$ sudo insmod /lib/modules/$(uname -r)/updates/dkms/iluvatar.ko
```

您可以使用 **uname -r** 命令查询 Linux kernel 的版本，例如：

```
$ uname -r  
3.10.0-1160.42.2.el7.x86_64
```

问题 4：希望在同一台服务器上使用 NVIDIA GPU 设备和天数智芯加速卡，但遇到报错信息。

问题描述

天数智算软件栈安装指南中“设置环境变量”指定了使用天数智芯加速卡需要的环境配置。在这种配置情况下，无法使用 NVIDIA GPU 设备。反之亦然。

解决方法

使用天数智芯加速卡，请确保执行了《软件栈安装指南》中“设置环境变量”步骤。如果要使用 NVIDIA GPU 设备，需要针对 NVIDIA GPU 设备和 CUDA 软件栈对应更新环境配置。

4 附录：支持的 Linux kernel 版本

已验证天数智算软件栈对以下 Linux kernel 的支持，随着加速卡适配的推进，该列表会持续更新。

4.1 Mainline

- 4.1.0-040100-generic
- 4.1.40-040140-generic
- 4.1.50-040150-generic
- 4.2.0-040200-generic
- 4.2.5-040205-generic
- 4.2.8-040208-generic
- 4.3.0-040300-generic
- 4.3.3-040303-generic
- 4.3.6-040306-generic
- 4.4.0-040400-generic
- 4.4.100-0404100-generic
- 4.4.200-0404200-generic
- 4.5.0-040500-generic
- 4.5.3-040503-generic
- 4.5.7-040507-generic
- 4.6.0-040600-generic
- 4.6.4-040604-generic
- 4.6.7-040607-generic
- 4.7.0-040700-generic
- 4.7.5-040705-generic
- 4.7.10-040710-generic
- 4.8.0-040800-generic
- 4.8.10-040810-generic
- 4.8.15-040815-generic
- 4.9.0-040900-generic
- 4.9.100-0409100-generic
- 4.9.200-0409200-generic
- 4.10.0-041000-generic
- 4.10.5-041005-generic
- 4.10.15-041015-generic
- 4.11.0-041100-generic
- 4.11.5-041105-generic
- 4.11.10-041110-generic
- 4.12.0-041200-generic
- 4.12.5-041205-generic
- 4.12.10-041210-generic
- 4.13.0-041300-generic
- 4.13.5-041305-generic

- 4.13.15-041315-generic
- 4.14.0-041400-generic
- 4.14.100-0414100-generic
- 4.14.200-0414200-generic
- 4.15.0-041500-generic
- 4.15.10-041510-generic
- 4.15.15-041515-generic
- 4.16.0-041600-generic
- 4.16.10-041610-generic
- 4.16.15-041615-generic
- 4.17.0-041700-generic
- 4.17.10-041710-generic
- 4.17.15-041715-generic
- 4.18.0-041800-generic
- 4.18.10-041810-generic
- 4.18.15-041815-generic
- 4.19.0-041900-generic
- 4.19.100-0419100-generic
- 4.19.200-0419200-generic
- 4.20.0-042000-generic
- 4.20.5-042005-generic
- 4.20.15-042015-generic
- 5.1.0-050100-generic
- 5.1.10-050110-generic
- 5.1.20-050120-generic
- 5.2.0-050200-generic
- 5.2.10-050210-generic
- 5.2.20-050220-generic
- 5.3.0-050300-generic
- 5.3.10-050310-generic
- 5.3.15-050315-generic
- 5.4.0-050400-generic
- 5.4.100-0504100-generic
- 5.4.180-0504180-generic
- 5.5.0-050500-generic
- 5.5.10-050510-generic
- 5.5.15-050515-generic
- 5.6.0-050600-generic
- 5.6.10-050610-generic
- 5.6.15-050615-generic
- 5.7.0-050700-generic
- 5.7.10-050710-generic
- 5.7.18-050718-generic
- 5.8.0-050800-generic
- 5.8.10-050810-generic
- 5.8.16-050816-generic
- 5.9.0-050900-generic

- 5.9.10-050910-generic
- 5.9.15-050915-generic
- 5.10.0-051000-generic
- 5.10.15-051015-generic
- 5.10.30-051030-generic
- 5.11.0-051100-generic
- 5.11.10-051110-generic
- 5.11.20-051120-generic
- 5.12.0-051200-generic
- 5.12.5-051205-generic
- 5.12.15-051215-generic
- 5.13.0-051300-generic
- 5.13.5-051305-generic
- 5.13.15-051315-generic
- 5.14.0-051400-generic
- 5.14.10-051410-generic
- 5.14.20-051420-generic
- 5.15.0-051500-generic
- 5.15.10-051510-generic
- 5.15.20-051520-generic
- 5.16.0-051600-generic
- 5.16.10-051610-generic
- 5.16.20-051620-generic
- 5.17.0-051700-generic
- 5.17.5-051705-generic
- 5.17.15-051715-generic
- 5.18.0-051800-generic
- 5.18.6-051806-generic
- 5.18.10-051810-generic
- 6.6.0-060600-generic
- 6.6.5-060605-generic
- 6.6.10-060610-generic
- 6.7.0-060700-generic

4.2 CentOS

- 3.10.0-327.el7.x86_64
- 3.10.0-327.3.1.el7.x86_64
- 3.10.0-327.4.4.el7.x86_64
- 3.10.0-327.4.5.el7.x86_64
- 3.10.0-327.10.1.el7.x86_64
- 3.10.0-327.13.1.el7.x86_64
- 3.10.0-327.18.2.el7.x86_64
- 3.10.0-327.22.2.el7.x86_64
- 3.10.0-327.28.2.el7.x86_64

- 3.10.0-327.28.3.el7.x86_64
- 3.10.0-327.36.1.el7.x86_64
- 3.10.0-327.36.2.el7.x86_64
- 3.10.0-327.36.3.el7.x86_64
- 3.10.0-514.el7.x86_64
- 3.10.0-514.2.2.el7.x86_64
- 3.10.0-514.6.1.el7.x86_64
- 3.10.0-514.6.2.el7.x86_64
- 3.10.0-514.10.2.el7.x86_64
- 3.10.0-514.16.1.el7.x86_64
- 3.10.0-514.21.1.el7.x86_64
- 3.10.0-514.21.2.el7.x86_64
- 3.10.0-514.26.1.el7.x86_64
- 3.10.0-514.26.2.el7.x86_64
- 3.10.0-693.el7.x86_64
- 3.10.0-693.1.1.el7.x86_64
- 3.10.0-693.2.1.el7.x86_64
- 3.10.0-693.2.2.el7.x86_64
- 3.10.0-693.5.2.el7.x86_64
- 3.10.0-693.11.1.el7.x86_64
- 3.10.0-693.11.6.el7.x86_64
- 3.10.0-693.17.1.el7.x86_64
- 3.10.0-693.21.1.el7.x86_64
- 3.10.0-862.el7.x86_64
- 3.10.0-862.2.3.el7.x86_64
- 3.10.0-862.3.2.el7.x86_64
- 3.10.0-862.3.3.el7.x86_64
- 3.10.0-862.6.3.el7.x86_64
- 3.10.0-862.9.1.el7.x86_64
- 3.10.0-862.11.6.el7.x86_64
- 3.10.0-862.14.4.el7.x86_64
- 3.10.0-957.el7.x86_64
- 3.10.0-957.1.3.el7.x86_64
- 3.10.0-957.5.1.el7.x86_64
- 3.10.0-957.10.1.el7.x86_64
- 3.10.0-957.12.1.el7.x86_64
- 3.10.0-957.12.2.el7.x86_64
- 3.10.0-957.21.2.el7.x86_64
- 3.10.0-957.21.3.el7.x86_64
- 3.10.0-957.27.2.el7.x86_64
- 3.10.0-1062.el7.x86_64
- 3.10.0-1062.2.0.el7.x86_64
- 3.10.0-1062.1.2.el7.x86_64
- 3.10.0-1062.4.1.el7.x86_64
- 3.10.0-1062.4.2.el7.x86_64
- 3.10.0-1062.4.3.el7.x86_64
- 3.10.0-1062.7.1.el7.x86_64

- 3.10.0-1062.9.1.el7.x86_64
- 3.10.0-1062.12.1.el7.x86_64
- 3.10.0-1062.18.1.el7.x86_64
- 3.10.0-1127.el7.x86_64
- 3.10.0-1127.8.2.el7.x86_64
- 3.10.0-1127.10.1.el7.x86_64
- 3.10.0-1127.13.1.el7.x86_64
- 3.10.0-1127.18.2.el7.x86_64
- 3.10.0-1127.19.1.el7.x86_64
- 3.10.0-1160.el7.x86_64
- 3.10.0-1160.2.1.el7.x86_64
- 3.10.0-1160.2.2.el7.x86_64
- 3.10.0-1160.6.1.el7.x86_64
- 3.10.0-1160.11.1.el7.x86_64
- 3.10.0-1160.15.2.el7.x86_64
- 3.10.0-1160.21.1.el7.x86_64
- 3.10.0-1160.24.1.el7.x86_64
- 3.10.0-1160.25.1.el7.x86_64
- 3.10.0-1160.31.1.el7.x86_64
- 3.10.0-1160.36.2.el7.x86_64
- 3.10.0-1160.41.1.el7.x86_64
- 3.10.0-1160.42.2.el7.x86_64
- 4.18.0-80.el8.x86_64
- 4.18.0-80.1.2.el8_0.x86_64
- 4.18.0-80.4.2.el8_0.x86_64
- 4.18.0-80.7.1.el8_0.x86_64
- 4.18.0-80.7.2.el8_0.x86_64
- 4.18.0-80.11.1.el8_0.x86_64
- 4.18.0-80.11.2.el8_0.x86_64
- 4.18.0-147.el8.x86_64
- 4.18.0-147.0.3.el8_1.x86_64
- 4.18.0-147.3.1.el8_1.x86_64
- 4.18.0-147.5.1.el8_1.x86_64
- 4.18.0-147.8.1.el8_1.x86_64
- 4.18.0-193.el8.x86_64
- 4.18.0-193.1.2.el8_2.x86_64
- 4.18.0-193.6.3.el8_2.x86_64
- 4.18.0-193.14.2.el8_2.x86_64
- 4.18.0-193.19.1.el8_2.x86_64
- 4.18.0-193.28.1.el8_2.x86_64
- 4.18.0-240.el8.x86_64
- 4.18.0-240.1.1.el8_3.x86_64
- 4.18.0-240.10.1.el8_3.x86_64
- 4.18.0-240.15.1.el8_3.x86_64
- 4.18.0-240.22.1.el8_3.x86_64
- 4.18.0-305.el8.x86_64
- 4.18.0-305.3.1.el8.x86_64

- 4.18.0-305.7.1.el8_4.x86_64
- 4.18.0-305.10.2.el8_4.x86_64
- 4.18.0-305.12.1.el8_4.x86_64
- 4.18.0-305.17.1.el8_4.x86_64
- 4.18.0-305.19.1.el8_4.x86_64

4.3 Ubuntu

- 4.15.0-20-generic
- 4.15.0-22-generic
- 4.15.0-23-generic
- 4.15.0-24-generic
- 4.15.0-29-generic
- 4.15.0-30-generic
- 4.15.0-32-generic
- 4.15.0-33-generic
- 4.15.0-34-generic
- 4.15.0-36-generic
- 4.15.0-38-generic
- 4.15.0-39-generic
- 4.15.0-42-generic
- 4.15.0-43-generic
- 4.15.0-44-generic
- 4.15.0-45-generic
- 4.15.0-46-generic
- 4.15.0-47-generic
- 4.15.0-48-generic
- 4.15.0-50-generic
- 4.15.0-51-generic
- 4.15.0-52-generic
- 4.15.0-54-generic
- 4.15.0-55-generic
- 4.15.0-58-generic
- 4.15.0-60-generic
- 4.15.0-62-generic
- 4.15.0-64-generic
- 4.15.0-65-generic
- 4.15.0-66-generic
- 4.15.0-69-generic
- 4.15.0-70-generic
- 4.15.0-72-generic
- 4.15.0-74-generic
- 4.15.0-76-generic
- 4.15.0-88-generic
- 4.15.0-91-generic

- 4.15.0-96-generic
- 4.15.0-99-generic
- 4.15.0-101-generic
- 4.15.0-106-generic
- 4.15.0-108-generic
- 4.15.0-109-generic
- 4.15.0-111-generic
- 4.15.0-112-generic
- 4.15.0-115-generic
- 4.15.0-117-generic
- 4.15.0-118-generic
- 4.15.0-121-generic
- 4.15.0-122-generic
- 4.15.0-123-generic
- 4.15.0-124-generic
- 4.15.0-126-generic
- 4.15.0-128-generic
- 4.15.0-129-generic
- 4.15.0-130-generic
- 4.15.0-132-generic
- 4.15.0-134-generic
- 4.15.0-135-generic
- 4.15.0-136-generic
- 4.15.0-137-generic
- 4.15.0-139-generic
- 4.15.0-140-generic
- 4.15.0-141-generic
- 4.15.0-142-generic
- 4.15.0-143-generic
- 4.15.0-144-generic
- 4.15.0-147-generic
- 4.15.0-151-generic
- 4.15.0-153-generic
- 4.15.0-154-generic
- 4.15.0-156-generic
- 4.15.0-158-generic
- 4.15.0-159-generic
- 4.15.0-160-generic
- 5.4.0-26-generic
- 5.4.0-28-generic
- 5.4.0-29-generic
- 5.4.0-31-generic
- 5.4.0-33-generic
- 5.4.0-37-generic
- 5.4.0-39-generic
- 5.4.0-40-generic
- 5.4.0-42-generic

- 5.4.0-45-generic
- 5.4.0-47-generic
- 5.4.0-48-generic
- 5.4.0-51-generic
- 5.4.0-52-generic
- 5.4.0-53-generic
- 5.4.0-54-generic
- 5.4.0-58-generic
- 5.4.0-59-generic
- 5.4.0-60-generic
- 5.4.0-62-generic
- 5.4.0-64-generic
- 5.4.0-65-generic
- 5.4.0-66-generic
- 5.4.0-67-generic
- 5.4.0-70-generic
- 5.4.0-71-generic
- 5.4.0-72-generic
- 5.4.0-73-generic
- 5.4.0-74-generic
- 5.4.0-77-generic
- 5.4.0-80-generic
- 5.4.0-81-generic
- 5.4.0-84-generic
- 5.4.0-86-generic
- 5.4.0-88-generic
- 5.4.0-89-generic
- 5.8.0-23-generic
- 5.8.0-25-generic
- 5.8.0-28-generic
- 5.8.0-29-generic
- 5.8.0-33-generic
- 5.8.0-34-generic
- 5.8.0-36-generic
- 5.8.0-38-generic
- 5.8.0-40-generic
- 5.8.0-41-generic
- 5.8.0-43-generic
- 5.8.0-44-generic
- 5.8.0-45-generic
- 5.8.0-48-generic
- 5.8.0-49-generic
- 5.8.0-50-generic
- 5.8.0-53-generic
- 5.8.0-55-generic
- 5.8.0-59-generic
- 5.8.0-63-generic

- 6.2.0-060200-generic

此文档仅供【zhangyj@skysolidiss.com.cn-天固信息安全系统（深圳）有限公司】查阅，请勿分享他人

此文档仅供【zhangyj@skysolidiss.com.cn-天固信息安全系统（深圳）有限公司】查阅，请勿分享他人

此文档仅供【zhangyj@skysolidiss.com.cn-天固信息安全系统（深圳）有限公司】查阅，请勿分享他人

此文档仅供【zhangyj@skysolidiss.com.cn-天固信息安全系统（深圳）有限公司】查阅，请勿分享他人

COREX01-MR401-IN01-01

34

V4.0.1-MR

5 商标声明

- 天数智芯、天数智芯 logo、Iluvatar CoreX 等商标、标识、组合商标为上海天数智芯半导体有限公司之注册商标或商标，受法律保护。
- 除了天数智芯的注册商标外，本内容中使用的其他产品名称及标志仅用于识别目的，该等名称及标志可能是归属于其各自公司的商标。我们否认对该等名称及标志的所有权利。
- CentOS 标识为 Red Hat 公司的商标。
- Docker 为 Docker 公司在美国和其他国家的商标或注册商标。
- Linux 为 Linus Torvalds 在美国和其它国家的注册商标。
- NVIDIA 和 CUDA 为 NVIDIA 公司在美国和/或其它国家的商标和/或注册商标。
- PyTorch 为 Facebook 公司的商标。
- TensorFlow 为 Google 公司的商标。
- Ubuntu 为 Canonical 公司的注册商标。