

deepin 集结

12^期
2018 年 12 月

【深度·春秋】

- 02 深度编辑器 V1.0 正式发布
- 04 深度商店 V5.1 发布
- 27 2018DDUC

【深度人·在说】

- 34 奔跑吧, Deepiner

【行业·观察】

- 38 中国软件业正成为开源生态贡献者
- 41 邬江兴院士: 新型网络技术发展思考

【深度·社区】

- 59 普通用户使用 deepin 比其他版本 linux 方便!!!!

【深度·讲坛】

- 62 anything is possible
- 72 Hello World CUDA!

【深度·伙伴】

- 80 Linux 是否适合作为桌面系统使用? 或许你该试一试 Deepin Linux

【深度·案例】

- 60 浅谈操作系统在农信系统实践国产化的意义



P50【特别策划】
深度操作系统
15.8
极致体验
美观高效

P54 如何让 deepin 变得更加开放和透明



很荣幸能够被邀请在《deepin集结》即将三岁的这一期写卷首语。记得在《deepin集结》创刊号的封面上，是深度科技100多位小伙伴的笑脸。他们当中有人已经离开了深度，甚至离开了操作系统这个行业，但是仍然有更多的小伙伴在为Linux操作系统而不懈奋斗。

一个伟大的事业，从来都不是一开始就会带着伟大而耀眼的光环让人觉得充满希望和信心，更多的是不被理解、不被相信和充满怀疑。但正是由于奋斗者们夜以继日的不懈努力，通过一点一滴的进步、经年累月的沉淀，最终才会铸就“伟大”这两个字。

因为深度人有着对发展国产操作系统这项事业的信仰、有着对自身变革图存的强烈愿望、有着面对市场和用户的需要不断去颠覆自我和推进技术创新的魄力与决心，所以只需要再有一些耐心、坚持和严谨的科学精神，我们必定能够用一个满意的答卷来回报我们在这条道路上所付出的青春。

——深度科技 总经办 王棣

策划 Hosted by
武汉深之度科技有限公司 Wuhan Deepin Technology Co., Ltd.
编辑 Edited by
《deepin集结》杂志编辑部 Editorial Office of DEEPINJIIE

总编辑 Editor-in-chief
刘闻欢 Liu Wenhuan
副总编 Deputy Editor
许可 Xu Ke
执行编辑 Executive Editor
秦娣 Qin Di
编辑 Editor
王棣 应鸿莉 Wang Di Ying Hongli
采编 Assistant Editor
李会会 黄锦敏 Li Huihui Huang Jinmin
美术设计 Art Editor
云云 Yun Yun

网站 Website
<http://www.deepin.com>
邮箱投稿 Contribution
deepin-magazine@deepin.com
市场推广 Marketing
account-marketing@deepin.com

武汉联络处 Wuhan Office
地址 Address
武汉市光谷大道77号
光谷金融港B18栋6楼
邮编 430223
电话 +86-27-87805607

北京联络处 Beijing Office
地址 Address
北京市西城区新街口外大街28号普天德胜B座603室
邮编 100088
电话 +86-10-62669499

上海联络处 Shanghai Office
地址 Address
上海市长宁区
延安西路1030弄16号404
邮编 200052
电话 +86-21-60250973

准印证号 (鄂) 4300107
承印单位 武汉金港彩印有限公司
出版日期 2018年12月
发放对象 公司员工、用户及合作伙伴、Linux爱好者

02 深度·春秋

- 02 深度编辑器 V1.0 正式发布——自由编辑，随心由我
- 04 深度商店 V5.1 发布
- 06 数字世界、智领未来，深度操作系统亮相第十四届中国（南京）国际软博会
- 09 安全保障，可靠运行，保障“安全可靠技术和应用研讨会”顺利召开
- 12 深度操作系统亮相“首届青岛军民融合 科技创新成果展”
- 14 深度操作系统参加第七届中国国际版权博览会
- 16 深度科技亮相 2018 年保密技术交流大会暨产品博览会
- 19 让公益更“工艺”，愿国产软件强国之梦走进孩子们的心中
- 22 国产软件强联合：深度操作系统与向日葵远程控制完成适配
- 24 深度操作系统海思平台服务器版软件 V15.2 产品发布！
- 26 湖北省副省长曹广晶带队调研东湖高新区有关企业
- 27 2018 第八届深度开发者与用户大会：武汉·北京·南京

34 深度人·在说

- 34 奔跑吧，Deepiner

38 行业·观察

- 38 中国软件业正成为开源生态贡献者
- 41 邬江兴院士：新型网络技术发展思考
- 49 法国力争芯片国产化替代 不被美国卡脖子

58 深度·社区

- 58 深度 Linux 已足够好，尤其对于一般办公者和开发者
- 59 普通用户使用 deepin 比其他版本 linux 方便！！！！

60 深度·案例

- 60 浅谈操作系统在农信系统实践国产化的意义

62 深度·讲坛

- 62 anything is possible
- 72 Hello World CUDA !

50 深度·策划



深度操作系统

15.8

极致体验
美观高效

P54

如何让 deepin 变得更加开放和透明

78 深度·伙伴

- 78 国产办公软件
从涅槃重生到跨越发展
- 80 Linux 是否适合作为桌面系统
使用？或许你该试一试
Deepin Linux

84 深度·生活

- 84 走吧，去太平洋看海



深度编辑器 V1.0 正式发布

——自由编辑，随心由我

● 深度科技 产品部 / 文

深度编辑器是深度科技自主开发的一款简单易用、可灵活定制部分功能的轻量级文本编辑器。

亮点介绍

整体沉浸式标签栏

类似 Chrome 的标签栏设计，支持标签拖拽、移动、拆分、合并等自由操作。



智能右键菜单

支持 272 种代码语法高亮，查看代码时更清晰结构化，发现问题一目了然。

更有“开启只读模式”、“在文件管理器中打开”两大高频刚需功能。

多套主题 UI

个性化工作界面，内置 5 套主题 UI，一秒快速切换，总有一款能捕获你的心。

快捷键映射

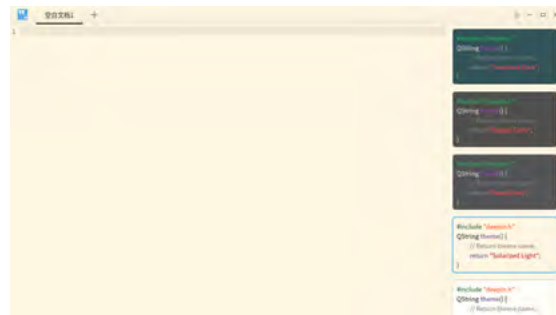
支持 3 种快捷键灵活映射：标准、Emacs、自定义。无缝贴合您的个性化习惯。

编辑器界面，按 Ctrl+shift+? 可呼出快捷键大全。

你是否已经迫不及待地想体验这款编辑器了呢？

安装方式

在深度商店搜索关键字“深度编辑器”，点击“安装”即可。 [d](#)





深度商店 V5.1 发布

● 深度科技 产品部 / 文

深度商店是深度科技打造的一款集应用展示、下载、安装、评论、评分于一体的应用程序。深度商店为您精心筛选和收录了不同类别的应用，同时每款应用都经过人工安装并验证，您可以进入商店搜索热门应用，一键下载并自动安装。

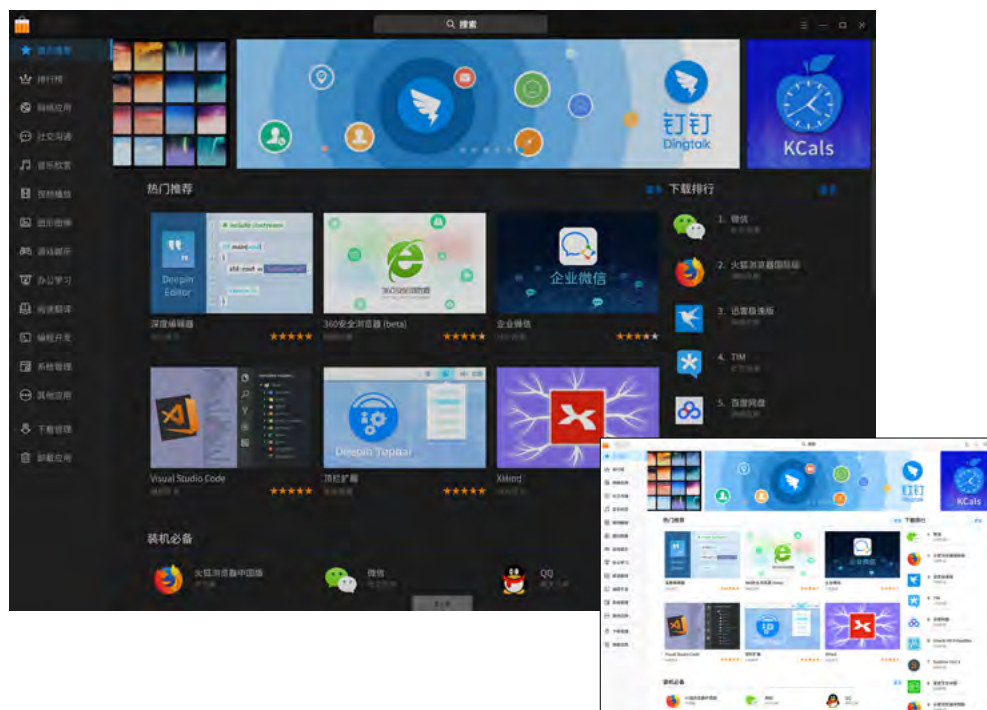
此次发布的版本为深度商店 V5.1。深度商店 V5.1 的亮点主要有：

新增打赏渠道

新增 PayPal 渠道，方便海外用户打赏。



微信、支付宝渠道是人民币元，Paypal 渠道是美元。



新增深色主题

切换方法: 点击商店界面右上角的主菜单按钮, 点击“深色主题”, 即可切换。

BUG 修复

- 修复弹窗内容、评论区内容不可选中并复制的问题。
- 修复应用搜索无效的错误。
- 修复大图预览高分屏模糊, 关闭预览自动重新打开的错误。
- 修复在商店打开其他应用后, 无法退出商店的问题。
- 优化排行榜记分规则。

投递应用入口优化

投递应用介绍:

<https://www.deepin.org/deliver-applications/>

投递应用入口:

<https://dstore-metadata.deepin.cn/app/create>

新增: 必填项星标、角色信息登记等, 助你投递应用不再迷茫。

更多亮点等待大家去发现。大家可通过系统更新来获取最新版的深度应用商店。d



数字世界、智领未来

深度操作系统亮相第十四届中国（南京）国际软博会

● 深度科技 市场部 / 文

8月31日上午，第十四届中国（南京）国际软件产品和信息服务交易博览会在南京国际博览中心隆重开幕。国家工信部张峰总工程师、信软司李冠宇副司长，江苏省及南京市领导娄勤俭、吴政隆、张敬华、樊金龙、马秋林、蓝绍敏、陈建刚、杨琦、王志忠、徐曙海、蒋跃建，以及20多个国家和地区的专家学者、企业高管、机构代表和重要客商共1000余人出席了大会开幕式及中国数字经济高峰论坛。

本届软博会以“数字世界、智领未来”为主题，以营造产业发展环境、展示发展成果、探索产业发

展趋势、推进交易合作为主线，集技术研讨、人才交流、成果交易、项目对接等专业功能于一体，突出专业化、引入市场化、彰显国际化，内容更加丰富、成效更加显著、影响更加广泛。

本届展会规模达11万平方米，美国、爱尔兰、荷兰、印度等20多个国家和地区的1000多家企业参展参会，包括富士通、ARM、阿里、腾讯、华为等国内外知名软件企业。

开幕式由南京市政府市长蓝绍敏主持，江苏省委常委、南京市委书记张敬华、苏州市市长李亚平、

工信部总工程师张峰分别在开幕式上致辞，江苏省政府省长吴政隆就推动数字经济发展做了重要讲话。省委书记娄勤俭、省长吴政隆、工信部总工程师张峰共同为第十四届中国（南京）软博会启动开幕。

高峰论坛上，中国科学院院士梅宏、中国工程院院士倪光南、加拿大工程院院士凌晓峰、浪潮集团董事长孙丕恕、运满满创始人苗天冶、中软国际董事局主席陈宇红等国内外专家学者、企业家代表围绕软件产业面临的机遇、数字经济新智能、构筑数字产业生态和构建数据治理体系等方面发表了主题演讲，对软件产业和数字经济发展进行了全方位研讨。

深度科技作为江苏申威安可产业联盟的成员，携众多奋斗在安全可靠事业中的企业组团参加了此次软件博览会。深度科技在博览会上展出了基于申威 411、421 的深度操作系统桌面版软件。

目前，深度操作系统已完成了申威等国产 CPU 的适配，能够为武器装备、网络安全等设备提供定制化的操作系统替换方案，日前已完成或对接了国内主流的网安公司，如 360 企业安全，绿盟科技，杭州安恒，启明星辰，中科网威等，形成了基于申威处理器+深度操作系统的网络安全产品开发平台。

深度科技在软件生态方面已与国内主流数据库、





中间件、安全软件有长期合作和适配。2015年以来，深度科技参与了装备发展部和部分军种的操作系统相关预研项目，并支持了包括火箭军、陆军、海军、战略支援部队多个军种的装备和信息化项目，在操作系统领域，能够完全替代国外同类产品。

自2016年以来，深度科技也参与了多个核高基项目，范围涉及国产芯片产业化操作系统平台支撑、党政办公操作系统平台研发与推广、操作系统基础版本共性技术研究等，深度操作系统也通过核高基项目的建设提高了产品能力，在国产化环境里将更适应党政办公及信息系统业务需求。

深度操作系统为逐渐替代国外操作系统而设计，能够替换大多数关键领域对操作系统的要求，包括：政务、金融、交通、能源、教育、医疗等众多领域。并已在多个部委应用，涉及到政务云、大数据、虚拟化等多领域范畴实施，在金融行业已经实现了ATM机的应用替换。

深度操作系统作为目前国产操作系统的佼佼者，

不仅拥有国内唯一完全自主研发的桌面环境（DDE），并已开发了近30款深度原生应用，包括：深度截图、深度影院、深度商店、深度录屏等常用应用，做到了真正开箱即用，极大地方便了用户的使用，完全满足日常工作。

借助此次第十四届中国（南京）国际软件产品和服务交易博览会，深度操作系统将自身的易用性、稳定性、安全性得以展示给关注信息安全、自主可控的相关领导，相信随着国家政策和市场需求，国产操作系统将越来越多地应用到相关安全领域。

未来，深度操作系统将以与时俱进的开发研究精神和扎实稳打的专业技术打造更优、更稳、更安全的国产操作系，在注重信息安全的国产化办公领域贡献成果。

更多亮点等待大家去发现。大家可通过系统更新来获取最新版的深度应用商店。d



安全保障，可靠运行

深度操作系统保障“安全可靠技术和应用研讨会”顺利召开

● 深度科技 市场部 / 文

9月18日，在美丽的山东济南召开了2018第三季度“安全可靠技术和应用研讨会”，会议由山东省经济和信息化委员会、安全可靠技术和产业联盟（以下简称安可联盟）共同主办。

国家部委和地方用户60余家单位，安可联盟90余家会员单位，共400余人参加会议。工业和信息化部信息化和软件服务业司、山东省经济和信息化委员会有关领导到会并致辞。

新品发布——深度商店 V5.0

深度科技副总经理在“安全可靠新产品发布”

环节，向与会领导深入介绍了深度商店 V5.0。深度商店是深度科技打造的一款品质精良、内容丰富的应用商店。集应用展示、下载、安装、评论、评分等功能于一体，精选热门推荐、新品应用和专题介绍，支持一键式下载安装、更新、卸载等功能。

深度商店 V5.0 具有以下特性：

- **统一样式：**使用统一的 DTK 控件，与其它 deepin 应用保持统一风格，窗口操作更加流畅；
- **更友好的出错提示：**当安装失败时，会尝试自动修复问题；如果无法自动完成，会显示出错误



原因以及可能的处理方法；

- 对高分屏的支持更完善：适配支持 4k 高分屏幕的分辨率，在高分屏上图片显示更清晰；
- 内容为王：高标准的产品，来源于高标准设计；高标准的产品，离不开高标准的运营。

私有化部署

基于安全可靠领域的特殊要求，深度商店为企业、政府级用户特地打造了深度商店私有化部署方案，即用户可以选择部署独立的应用商店，或者将深度商店部署在企业的私网里面，以获得安全、管控、策略执行等专属特性。

“私有化部署”可支持私网部署、企业部署、内网部署、保密部署。这即是安全可靠环境的统一操作系统平台要求，也是深度科技对安全可靠生态的承诺和贡献。

应用投递、打赏与体现

深度商店支持应用软件投递，深度科技有专人负责收录优秀应用并上架到深度商店，以“精品应用商店”作为全新定位，评估严格，坚持只上架精品应用，杜绝滥竽充数现象，用户可以在全局菜单里面“推荐应用”，向我们推荐优质的应用。另外，如果用户发现某个应用有了更新的版本，可以通过催促更新来告知官方，官方会按照催更的频率和优先级尽快安排版本更新。

除此之外，深度商店 V5.0 还开通了打赏功能，秉承着尊重开发者的劳动价值的基本原则，官方默认开通其打赏功能，提供上架后打赏分成、提现申请、使用等功能，通过用户的真实使用来奖励优秀应用，并为应用开发者带来一定的收益。

该功能是深度科技为建立健全国产操作系统生态环境迈出的重要一步，希望能够得到各大应用软件开发及合作伙伴的支持和认可，并早日将国产操作系统的软件生态建立起来。



深度操作系统携手龙芯，全国产化保障会议进行

此次安全可靠技术和应用研讨会，是安全可靠领域每季度的一次重要会议，这也是深度操作系统第二次全程保障会议的视频播放、PPT演示的工作。

此次演示工作采用的是搭载深度操作系统的龙芯 3A3000，在匹配 LED 大屏、翻页笔等外设方面非常方便、快捷，演讲演示的 PPT 通过 WPS 也达到了正常演示的效果，这充分表明，“深度操作系统 + 龙芯 + WPS”的全国产化组合，完全可以满足日常办公需求。

通过此次保障“安全可靠技术和应用研讨会”的演示工作，更加坚定了用户对国产化发展的信心，从“可用”、“能用”到“好用”，国产圈的同仁一直在孜孜不倦地努力着，通过上下游厂商不断的坚持协作，相信国产化的发展会越来越美好！

深度操作系统，友好美观·易用可靠

在安全可靠产品与应用展区，展示了深度操

作系统龙芯版软件，该版本提供了基于安全可靠环境的国产操作系统替换方案，目前已支持龙芯 3A3000、3A2000 等型号的芯片。

另外深度操作系统龙芯版集成了“龙芯版应用商店”，这是在国产化平台上做出的一项突破性的成果。目前龙芯版深度商店中已有近 800 款应用软件，涵盖了社交沟通、视频播放、办公学习、游戏娱乐等类别的应用，为全国产平台软件应用的下载、安装、使用提供了方便快捷的途径。

此次“安全可靠技术和应用研讨会”的顺利召开，连同深度科技在内，共发布了近 40 款新产品和新应用，这足以体现出在安全可靠领域的上下游厂商们协同推进，苦干实干的作风，也为我国的网络安全和信息化工作做出了卓越贡献，深度科技将携手业内同仁，坚定不移地坚守自己的使命，打造更加美观易用·安全可靠的国产操作系统，也将不遗余力地发展建设更加广阔的国产操作系统新生态。d



深度操作系统亮相“首届青岛军民融合科技创新成果展”

● 深度科技 市场部 / 文

以“融合·创新”为主题的首届青岛军民融合科技创新成果展，10月10日在位于青岛西海岸新区的青岛世界博览城开幕。本届展会持续5天，包括11家央企军工集团、8所军工科技院校等近400家单位参展。

展会共设置西海岸新区军民融合成果展、军工企业成果展、综合科技成果展三大展区，展览面积达4万平方米，参观人数突破30000大关。

深度科技作为国产操作系统的生产厂商，在成

果展中展示了美观易用·安全可靠的深度操作系统申威版软件。

深度操作系统作为近年来发展迅速的基于Linux的国产操作系统软件，目前已完成申威等国产CPU的适配，能够为武器装备、网络安全等设备提供定制化的操作系统。目前已完成或对接了360、绿盟科技、杭州安恒、启明星辰、中科网威等国内主流的网安公司。形成了基于申威处理器+深度操作系统的网络安全产品开发平台。在拥有国内顶尖的操作系统研发团队的基础上，深度操作系



统已具备了完全替代国外操作系统的能力。

除此之外，深度操作系统还支持其他CPU平台，如龙芯、飞腾、兆芯、X86等；并与国内主流数据库、中间件、安全软件有长期合作和适配。2015年以来，参与了装备发展部和部分军种的操作系统相关预研项目，并支持了包括火箭军、陆军、海军、战略支援部队多个军种的装备和信息化项目。

深度操作系统包括：深度操作系统桌面版、深度操作系统服务器版、深度安全操作系统等，深度

操作系统系列产品均已支持全国产CPU平台，为信息安全领域达到全国产化做好了万分准备。

通过此次“首届青岛军民融合科技创新成果展”，深度科技不仅展示了交互良好，运行稳定的深度操作系统，更认识了自主可控领域的更多优秀企业及产品，相信在国产自主可控领域上下游厂商的不懈努力下，更具创新精神的优秀国产软硬将会有更好的未来，为提高我国关键信息基础设施自主安全、促进自主基础软件、提升关键信息基础设施安全保障贡献自己的力量。d



深度操作系统 参加第七届中国国际版权博览会

● 深度科技 市场部 / 文

10月19日，由中国国家版权局主办，江苏省版权局、苏州市人民政府承办的第七届中国国际版权博览会在苏州拉开帷幕。博览会以“交流、合作、创新、发展”为主题，旨在通过展示版权产业成果，交流国内外版权工作经验，推动版权产业持续快速发展。

中国国家版权局版权管理司司长于慈珂，世界知识产权组织（WIPO）副总干事王彬颖，韩国文化体育观光部著作权局局长文荣皓、江苏省版权局局长缪志红、苏州市人民政府市长李亚平在开幕式

上致辞，国际版权机构高级别官员，海内外版权政府部门官员、版权相关单位和产业界代表等中外嘉宾共同出席了开幕式。

中国国际版权博览会是国家版权局按照国际化、专业化、市场化原则举办的唯一常态化综合性的国家级版权专业博览会，2008年创办至今，已在北京、成都、广州等地成功举办了六届，得到众多国家、地区和企业的积极响应，同时赢得了世界知识产权组织和各级政府、行业协会的高度赞誉，成为开展国际版权交流、促进中外合作的具有国际



影响力的博览会。

本届博览会展示面积达 2.6 万平方米，参展单位和机构共计 300 余家，设有国际展区、国内展区、版权产业展区、江苏展区和版权项目路演五大展区，重点展销图书音像、影视音乐、动漫游戏、计算机软件、工艺美术等优秀版权作品，充分展示近年来国内外版权产业发展的丰硕成果。

深度科技作为湖北省唯一一家国产操作系统生产厂商，随湖北省版权一起参加了此次版权盛会。



近两年，深度科技也在积极地参与国家版权局号召的正版软件国产化的相关工作，在今年 8 月，完成了云梦县国家级正版化软件试点项目的验收，深度操作系统在项目中满足实施安装率 92.42%，其中安装单系统的电脑 408 台，占比 83.61%。从开始安装到项目基本完成，深度操作系统软件技术成熟、性能稳定、界面清晰、运行顺畅、使用便捷、满意度高，比起 Windows 操作系统毫不逊色，获得了用户的高度认可。

深度操作系统通过近年来的努力，产品在美观易用的同时，也越来越稳定，软件兼容程度高，可满足用户的基本办公需求。深度团队也在各党政机关试点办公中，将遇到的涉及软件厂商迁移衔接、系列驱动程序优化、业务系统实用技术迁移等关键技术的问题，按类分解，集中骨干力量，使软件功能更加完善。并总结了一套专门应用在党政办公领域的解决方案，相信通过深度团队的匠心精神，技术成熟度将会越来越高。d



深度科技亮相 2018 年保密技术交流大会暨产品博览会

● 深度科技 市场部 / 文

10月31日~11月2日，为期3天的“2018年保密技术交流大会暨产品博览会”在青岛国际博览中心举行，今年展会的主题是“坚持创新驱动，携手打造保密产业良好生态”。近600家企业和科研单位报名参展，参观观众达70000人。中办副主任陈世炬、中央保密办主任、国家保密局局长田静，青岛市委副书记、市长孟凡利等领导，以及中国电子董事长、党组书记芮晓武，多名两院院士出席开幕式并到各展台参观。

本次展会由中国保密协会主办，青岛国家高新技术产业开发区、青岛市保密协会等承办，展会展览面积预计60000平方米。着重围绕贯彻落实习近平总书记关于保密工作、科技创新工作的重要讲话精神和中央关于保密工作的决策部署，旨在大力推动保密技术创新和产业转型升级，推动企业与用户、企业与企业、国内与国际间进行保密技术交流合作，增强机关单位干部职工保密意识，普及公众信息安全保密防范常识，充分发挥保密科技发展政策引领



作用、市场资源配置作用、企业创新主体作用，提高保密科技研发和产业发展水平。

会展由保密技术交流大会、2017年度保密科学技术奖励大会、保密技术产品博览会、保密技术论坛等活动组成，同时，设立了信息安全保密公众体验区，围绕公众关心的信息安全保密问题，通过警示教育、态势感知、互动演示、视频播放、图书展览等方式，普及保密法律法规和信息安全知识，增强公众信息安全保密防范意识，提高基本防范技能。

深度科技携手申威联盟，在 S1 馆青岛展团的



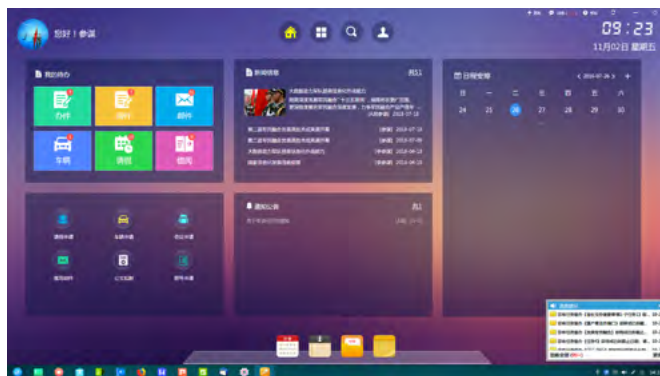
基于深度操作系统的申威 411 笔记本

主场位置亮相此次产品博览会。除此之外，和信创天、龙芯中科、中宏立达、方正等战略单位也均展示了美观易用、安全可靠的多版本深度操作系统。

深度科技展台此次展示了最新的基于申威的深度操作系统桌面版软件。

目前，深度操作系统已完成包括申威、龙芯等国产 CPU 的适配，为武器装备，网络安全等设备提供定制化的操作系统，并积极参与构建更广阔的国产软硬件生态体系，合作伙伴包括 360、绿盟科技、杭州安恒、启明星辰、中科网威等国内主流的





基于深度操作系统的全国产通用办公信息系统



网安公司。

深度科技总经理刘闻欢也在此次保密技术交流会暨产品博览会中向众多领导介绍了深度操作系统。

自 2015 年以来，深度操作系统参与了装备发展部和部分军种的操作系统相关预研项目，并支持了包括火箭军、陆军、海军、战略支援部队多个军种的装备和信息化项目。

自 2016 年以来，深度操作系统也参与了多个核高基项目，范围涉及国产芯片产业化操作系统平台支撑、党政办公操作系统平台研发与推广、操作系统基础版本共性技术研究等，深度操作系统也通

过核高基项目的建设提高了产品能力，在国产化环境里将更适应党政办公、信息系统业务、保密需求。

目前，深度操作系统也将软件生态扩展到各保密应用软件中来，并以美观、稳定、安全的姿态服务更多。

借助此次“2018 年保密技术交流会暨产品博览会”，深度科技将深度操作系统的易用性、稳定性、安全性展示给关注保密建设及军队、军工建设的相关领导，相信随着国家政策和市场需求，国产操作系统将越来越多的应用到相关安全领域。未来，深度操作系统将以与时俱进的开发精神和扎实稳打的专业技术，打造更优、更稳、更安全的国产操作系，为国家信息安全及保密事业做出积极贡献。d

让公益更“工艺”

愿国产软件强国之梦走进孩子们的心中

● 深度科技 市场部 / 文

9月17日，我作为深度科技的一名普通员工，有幸参与了一次有意义的授课——向湖北省某镇一中学七年级的学生讲解“信息安全之国产操作系统”。

经老师介绍，这里的孩子多是留守儿童，父母都在大城市工作打拼。孩子们眼中充满着对知识的渴求。在我看来，艰苦的教学环境并没使孩子们落下教育，反而使他们更具艰苦奋斗的精神，就像我们的工作一样。而我作为在国产操作系统一线工作的员工，也很荣幸能有机会让祖国的花朵们了解我的工作，认识国产操作系统给我们的生活、我们的国家带来的深远意义。

我从三个反问为题，向孩子们介绍了信息安全的重要性，和国产操作系统的发展，还有对孩子们寄予的深厚期望。

为什么要有国产操作系统？

操作系统对于大家并不陌生，因为这是我们几乎每天都会接触到的东西，手机里面有操作系统、电脑里面有操作系统、汽车里面有操作系统、银行取款机里面也有操作系统，每天乘坐公共交通也会接触操作系统……操作系统像是身体里的血液，已经融汇在我们生活中的点点滴滴。

然而就是这样普遍使用、不可或缺的技术，时至今日，我们国家还处在被外国技术“卡脖子”的境地。因为不能掌握原始代码，别人暗插进去的后门漏洞，随时都有可能让我们国家的关键领域、关键部门、关键信息被窥视、被窃取、被控制。国家领域如此，个人信息安全就更难把握了。

像去年的 WannaCry(想哭) 病毒，一时爆发，瞬间殃及全球 100 多个国家的计算机，而我国国内高校校园网、政府部门、银行、中石油加油站等网络均遭受不同程度的感染，被感染的电脑会在 10 秒内锁住，所有文件被全部加密，只有按照弹窗提示交纳一定数量的比特币作为赎金才能解锁。超过期限，黑客就会“撕票”销毁数据。

今年爆发了“中兴被制裁”事件，也给我国信





息安全敲响了警钟。核心技术花钱买不来，如果再不痛定思痛地做自己的技术，让外国人掐着我们的脖子，那何谈中国崛起？而这样的“事故”其实离我们并不远，也许明天，就会发生在我们身上。

国产操作系统怎样给国人信心？

一提到“国产”、“自主可控”，这样的词语似乎被玩坏了。一方面由于我们技术发展起步晚，另一方面就是投机取巧、唯利是图的偷工减质而造成的“笑话”太多，那不能说有问题就不再给国产技术信心，现在不发展、不关心，只会让这个被人卡脖子的时代进入恶性循环。

好在近年来国家非常重视信息安全，领导人也多次强调要发展核心技术，不能让核心技术受制于人，而我们国家，依然有奋斗在自主可控领域的人才在奋起耕耘。

在国产操作系统领域，我们的产品和技术已经具备了在办公领域对外国同类产品替代的能力。这里就以深度操作系统为例，向大家介绍国产操作系统如今的面貌。

从第一版深度操作系统问世至今，深度操作系统已经拥有了 33 个国家的粉丝群体，深度操作系统的镜像站点遍及六大洲共有 105 个，当然，这些数字还在不断增长中。除了在国际 Linux 发行版中崭露头角，深度操作系统也多次入围中央国家机关政府采购名录，并已具备了相关军工资质，积极参加军队军工建设，在党政军多单位实际应用，获得了一致好评。

深度操作系统是一款基于 Linux 的操作系统，Linux 操作系统内核代码经过二十年全球数千开发



者不断的更新，加上包括各大硬件厂商的积极参与，其代码质量完全达到了一流商业软件的水平。

在这样的基础之上，深度操作系统自主研发了适合中国人自己的桌面环境（Deepin Desktop Environment，简称“DDE”），DDE 完美呈现了深度操作系统的“美观易用”产品理念，将 Linux 下复杂的操作全部图形化，用鼠标操作代替原本的键盘操作，并设计了多样式的任务栏，全局及迷你的应用列表，及一目了然的控制中心。这样突破性的设计，完全打破了外界质疑国产操作系统“只会做皮肤”的言论。DDE 也通过版本的迭代更新，得到了全球千万级用户的认可，并已被土耳其国家支持的操作系统移植过去，深度操作系统由此也成为第一个被外国政府拿去移植使用的来自中国的操作系统。

除了 DDE，深度操作系统还自主研发了近 30 款使用简单、交互良好的深度原生应用。这其中最为受欢迎的便是深度商店，目前深度商店集结了几千款应用，都是一键下载安装，完全解除了 Windows 下复杂的安装步骤。除了深度商店，还有深度截图、深度录屏、深度影院等应用，好评如潮。

深度操作系统自带的应用还是有限的，为使用户真正能用国产操作系统办公学习，深度还与搜狗



输入法、网易云音乐、360 浏览器、WPS 等达成战略合作，并采用 deepin-wine 的方式，解决了 QQ、微信等常用应用在 Linux 下的使用问题。

目前深度操作系统刚刚完成了国家版权局在湖北省云梦县的全县党政机关的国产操作系统替换项目，替换使用率高达 83%+。在教育领域，也成功为福建省“班班通”进行了安装使用，在金融领域，上线运行了中国第一台基于 Linux 的 ATM 机……未来，将有更多领域使用国产操作系统。

国家信息安全，我们能做什么？


作为学生，我们应该将“信息安全”、“自主可控”的概念牢记于心，从思想上认识到信息安全对我们国家和人民的重要意义，如果将来有机会走上相关行业，用自己的所学贡献在国产自主可控的发展中。只有通过越来越多年轻一代学子的新鲜力量，才能使我国信息化建设保持活力与创造力，早

日追上我国在信息安全领域落下的课。

国家信息安全与个人信息安全都到了迫在眉睫需要关注的时刻了，希望我们此次之行能够将其重要意义播撒在孩子心中，也殷切希望我们的下一代都能关注信息安全，能够接下发展自主可用事业的大旗，让国产芯片、国产软件及基础操作系统都能发展起来，为打造国家信息安全的盾牌而努力。

话外

深度科技在关心助力乡村教育的同时，也为湖北省孝感市某村捐款 15 万元，用于修建村里的公路。希望通过深度科技的微薄之力，能够帮助那里的孩子走出乡村，用自己的勤劳与智慧，知识与能力开创自己的未来，也希望借此让更多的老师和孩子们对信息产业有所接触，加入到深度及信息安全相关行业，为祖国的科技强国战略添砖加瓦。

在此向那些不忘初心，扎根基层的乡村教师致敬！



国产软件强联合：深度操作系统与向日葵远程控制完成适配

● 深度科技 市场部 / 文

近日，深度操作系统与国内远控行业领导者向日葵远程控制软件达成合作——深度操作系统与向日葵 Linux 客户端完成适配。

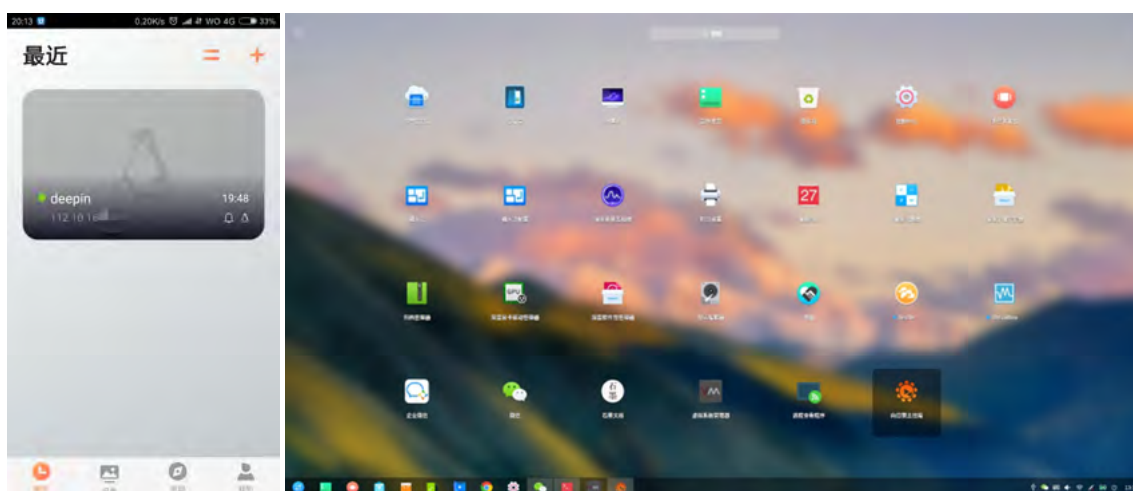
国产软件的骄傲，远控行业的风向标

向日葵远程控制是国内知名的远程控制服务商，目前已研发向日葵远程控制软件、向日葵开机棒、控控、UU 等多款硬件，通过软硬结合的方式

提供完美的远程控制解决方案。

通过软硬件的搭配，可以实现纯内网状态下的远程控制，解决网络受限企业的维护难题；支持无人状态下，远程开机到控制一体化的体验……

历经 9 年的打磨，向日葵远程控制目前已成为国内远控行业的标杆，拥有 3000 万用户，敢



为人先承诺为所有用户提供 7x12 小时的真人在线客服，并永久免费开放核心功能“远程桌面控制”。

国产操作系统与国产软件的相互支持、共同进步

全球化推动下的国内企业与国外企业可谓是处处竞争，部分国内企业凭借一流的科研实力以及工匠精神，闯出了一番天地。

深度操作系统和向日葵远程控制的合作将是国内优秀软件协同发展的见证，“深度操作系统”是进入过全球开源操作系统前十的中国操作系统产品，目前已广泛用于国内党政军、金融、运营商、教育等行业，拥有众多国内外用户，而向日葵远程控制软件，通过完美适配深度操作系统，能够为深度操作系统用户提供更简单、更好用的远程控制方案。

今后，广大用户通过使用向日葵远程控制软

件，可实现手机远程控制 Linux 系统的主机，完成远程维护、远程文件传输、远程 ssh 等操作，弥补了在深度系统上无法使用远程控制软件的遗憾。

严峻的国际形势催促着国内各行业进行战略创新，向日葵远程控制与深度操作系统的合作，提高了国产操作系统与国产软件的国际竞争力，相信能惠及更多企业和个人用户。此次完美的适配也展示了国产操作系统与国产软件打造优质互联网生态圈的决心！

深度操作系统一直致力于打造更为完善的操作系统生态，相信通过与向日葵远程控制的适配工作，可以为用户带来更便捷的操作体验，同时，我们也期待更多软件厂商能够关注国产操作系统的生态建设，深度操作系统将始终以热情的态度欢迎大家积极适配，协同发展，共创佳绩。d



深度操作系统海思平台服务器版软件 V15.2 产品发布!

● 深度科技 市场部 / 文

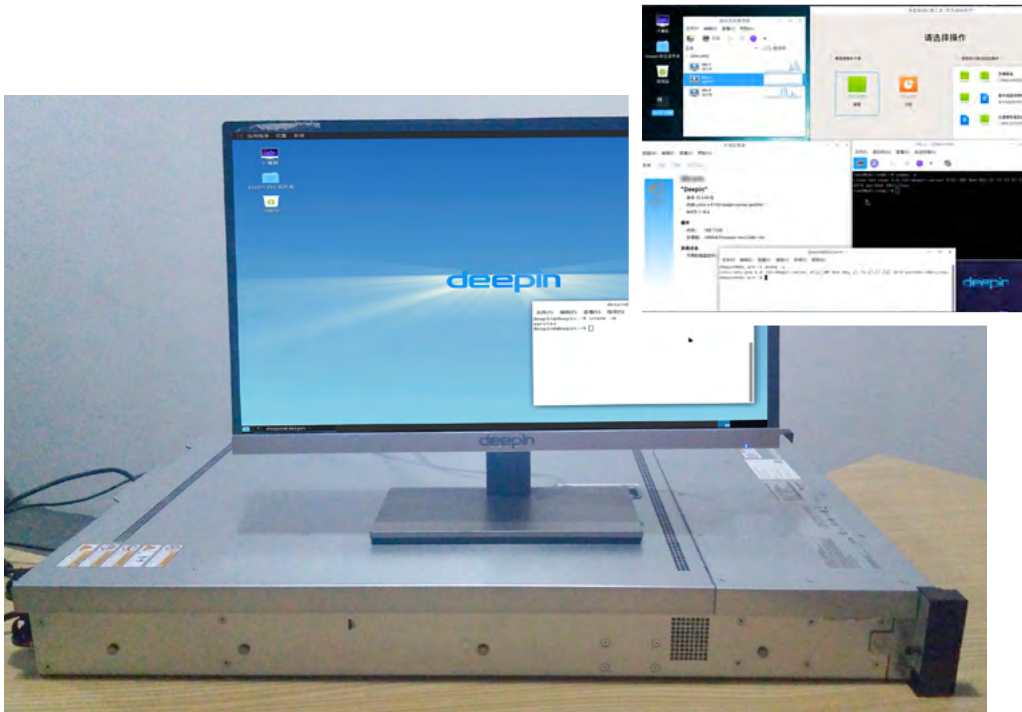
深度操作系统海思平台服务器版软件是武汉深之度科技有限公司发布的针对华为海思平台的TaiShan系列服务器发布的企业级服务器操作系统软件产品，主要面向企业级服务器应用场景，为用户在国产化平台上提供更具可用性优势的数据中心、云平台、分布式存储、大数据等解决方案选择。

深度科技研发团队经过近年来产品开发和项目实施，基于深度操作系统产品已积累了大量的技术经验，此次结合华为自研海思CPU平台特性，

通过整合上游社区资源，成功发布了针对Hi16xx CPU平台的深度操作系统海思平台服务器版软件V15.2产品。

该服务器操作系统产品经过深度科技和华为公司的共同严格测试，整体系统稳定可靠、性能卓越，可以为企业级应用提供全面可靠的保障，针对国产化应用需求打造新的国产软硬件平台。

该服务器操作系统产品，已经适配国产主流中间



件产品：金蝶、东方通等；国产主流数据库产品：达梦、南大通用、人大金仓、神舟通用等。深度科技欢迎更多的国产软硬件厂商，共同参与国产化应用生态环境的建设！

深度科技专注于基于 Linux 的操作系统的产品研发，面向党政、金融、能源、教育、医疗等行业提供基于 Linux 操作系统的解决方案、技术支持和培训等专业服务。深度科技将致力于在该平台上为用户提供更专业的企业化云计算、虚拟化解决方案技术支持，提供更高效能的开源云基础设施选项，完成国产化云平台突破。用户与合作伙伴，可以陆续从深度科技获取到面向企业级虚拟化管理的解决方案、分布式存储解决方案、大数据平台解决方案。

作为安全可靠技术和产业联盟的成员单位，深度科技和华为的积极配合，以及联合华宇、太极共同完成了基于新打造的国产化应用新平台，针对电子公文应用的兼容性测试，涉及主流国产数据库和中间件的应用。测试结果表明，该平台充分满足安全可靠电子公文的应用要求，整体系统性能突出，能够适应更高要求的业务场景需要，具有很大的市场竞争优势。

基础硬件平台的自主可控是国家信息安全体系的底座。只有基础硬件平台有了全国产化方案，才能支撑打造完整、可控的国产化应用生态，产业上下游也就有了加快与应用系统的适配和优化的基础，最终实现提升国产 IT 产品的交互体验、满足应用需求的目的。d



湖北省副省长曹广晶带队调研东湖高新区有关企业

● 深度科技 市场部 / 文

2018年11月02日，湖北省人民政府副省长曹广晶、湖北省人民政府副秘书长胡道银、湖北省经信委副主任王建民、李瑞勤、刘文高、赵华文、郑汪洋、曾旷怡一行领导参观调研了湖北省唯一一家国产操作系统高新企业——武汉深之度科技有限公司（以下简称‘深度科技’）。

当前，国家高度重视信息安全发展，习近平总书记也讲到：核心技术靠化缘是要不来的，我们必须认识到，没有网络安全就没有国家安全，安全是发展的前提，发展是安全的保障，安全发展需要同时推进，及早投入力量坚决突破，一劳永逸地掌握信息核心技术，绝不能有侥幸心理。

深度科技作为一家民营企业，已专注国产操作系统的研发与服务近十年，为广大党政军及企业级用户提供了安全可靠、美观易用的国产操作系统解决方案，并一直致力于打造国产操作系统的良性生态建设。调研期间，深度科技总经理刘闻欢向参加调研的领导作了工作汇报，并分析了

目前国内操作系统行业发展的概况，国内和国际操作系统产品的现状和对比情况，以及对国产自主可控提出的认识和建议。刘闻欢总经理还表示：“深度科技将立足湖北，为国家信息安全和自主可控贡献力量。”

调研座谈期间，曹广晶副省长指出，在响应习总书记关于信息安全号召的同时，也为了更好、更快地发展企业，需要尽快地完善国产操作系统的生态链，从自主可控产业链以及军民融合的大方向上出发，立足于湖北，打造属于湖北的国产生态圈。

此次湖北省副省长曹广晶一行领导对企业的实地考察，不仅了解了国产操作系统的发展及产品情况，参观了国产操作系统的应用展示，还通过召开座谈会，详细了解企业技术研发及市场拓展等方面的情况，这对加快发展湖北省操作系统自主可控产业链指明了方向，也对湖北省软件的发展提出了新的目标。湖北省各调研领导也对我省发展操作系统的自主可控建设表示大力支持。d





第8届深度开发者与用户大会

武汉·北京·南京

deepin
深度科技



WHLUG

开源中国
oschina.net

IT大咖说

活动行



2018 第八届深度开发者与用户大会

武汉站

9月16日 / 湖北大学

>>>

9月16日，武汉并没有受到台风山竹的影响，气温适宜，第八届深度开发者与用户大会武汉站在湖北大学图书馆一层顺利召开。在活动开始前的一个半小时，就已经有热爱 deepin、热爱开源的小伙伴结伴而来……

此次第八届深度开发者与用户大会由深度科技主办，并得到了湖北大学、中国计算机学会、武昌首义学院、WHLug 等其他院校、组织的大力支持，共有 110 余人参与现场活动，该活动全程由 IT 大咖说进行了视频直播，截止到发稿前，共有 2500 人在线观看了武汉站的精彩活动，湖北省电视台也对此次活动进行了报道。

此次活动由湖北大学计算机学院张彦副院长、

深度科技高级研发工程师、高级产品经理、研发工程师、HR，Arch Linux 开发者，武昌首义学院、武汉 Linux 研究组组长参与并发表了精彩演讲。

湖北大学计算机学院的张彦副院长在活动一开始，就做了精彩致辞。他表示，湖北大学与深度科技有着密切的联系，这不只是因为深度科技的总经理刘闻欢等负责人皆毕业于湖大，为湖大做出了贡献，更是因为湖北大学也正积极地探索开源，并已与深度科技达成了部分战略合作。希望有更多的学生和年轻人加入并参与到开源项目中，积极地将自己的所学贡献在开源世界里。

深度科技高级研发工程师王耀华在此次 DDUC 活动中，向大家介绍了深度操作系统 15.7 在研



发过程中的一些小故事，这个版本更新了底层的debian仓库，还为整个版本做了瘦身，开机内存从1.1G降到了800MB，体积减小为2.5GB，比上个版本的3.1GB优化了0.6GB。

应部分用户及粉丝的要求，此次DDUC由深度科技工程师王明栋向大家介绍了“deepin的成长”，deepin是一家由社区发展而来的商业公司，从最早的五人的线下研发团队慢慢成长为现在的规模，随着版本的迭代，deepin在众多国际发行版中渐渐崭露头角，向世界展示了中国人的智慧，同时，deepin也将自己的研发成果全部回馈给开源社区，希望通过deepin的贡献，能够为用户提供更便利的使用体验。

Arch Linux开发者晏然以“开源社区维护者的日常”为题，发表了精彩演讲，他向大家讲述了自己从一名普通的用户成长为Arch Linux开发者的小故事，并分享了在开源社区里当一名维护者的体验，他表示，这种体验更像是照顾嗷嗷待哺的用户们，看着自己的“娃”长大，痛并快乐着。

武昌首义学院、武汉Linux研究组组长徐斌的演讲题目是“飞奔的Linux”，他介绍了自己创办武汉Linux研究组的过程。而作为一名相关专业的老师，他更清楚学生该如何参与开源活动，

参与开源活动的同时，又该怎样为自己的开源事业打开门路。

deepin作为来自中国的“最美”操作系统，其界面设计和用户体验一直被用户看好，深度科技高级产品经理王珈做了“用户体验的平凡之路”的精彩演说，向用户讲述了deepin界面及功能设计的方方面面，并举例说明了产品经理在产品设计过程中该如何取舍。

在WHlug中非常活跃并拥有超高人气的深度科技研发工程师乔老师也参与了此次活动，并以“initrd/initramfs机制简介”为题，带大家重新认识了initramfs，简要分析了其工作原理，并介绍了如何对initramfs做简单的定制与修改。

活动最后，深度科技的HR小姐姐向大家公布了deepin的最新招聘信息和求职中应当注意的一些问题，希望更多的小伙伴加入到deepin这个大家庭。

大会结束后所有参会人员合影留念，在这里我们感谢湖北大学、中国计算机学会、武昌首义学院、WHlug、湖北省电视台、IT大咖说的支持和关注，更要感谢到达活动现场的deepin用户和粉丝们。d



2018 第八届深度开发者与用户大会

北京站

10月14日 / 中国人民大学

>>>

10月14日，第八届深度开发者与用户大会北京站在中国人民大学顺利召开，活动得到了中国人民大学及其计算机学院、开源中国、IT大咖说的支持。

此次活动应粉丝要求，deepin 首席技术官张磊、高级研发工程师王耀华等 deepin 大咖为在场用户分享了 deepin 的最新成果和技术，本次活动还邀请了 360 浏览器产品总监杭程和曾参与 Firefox Gecko 内核开发者李沫南，及 GNOME 基金会成员佟辉、TUNA 会长姚沛然。

活动开始，deepin 工程师王明栋以“深度商店不再是单机版”为题，向大家介绍了深度科技全新打造的深度商店 V5.0，深度商店截止到 2018 年 10 月，已经经历了 5 个版本的迭代，以往简单的

对仓库中的软件包进行展示和下载的单机版已经逐渐沦为垫脚石，新版的深度商店将开始尝试在操作系统、开发者和用户三者之间搞事情，希望通过打赏与提现的方式让更多开发者将更好、更优的应用献给广大用户。

接下来，360 浏览器的产品总监杭程为与会者介绍了“360 浏览器技术与规划”。截止到 10 月 11 日的的数据，360 安全浏览器上架到深度商店后，下载量达 8793 次，也获得了很多用户的赞赏。360 安全浏览器是一款基于 Chrome 内核的浏览器。它拥有全国最大的恶意网址库，采用恶意网址拦截技术，可自动拦截挂马、欺诈、网银仿冒等恶意网址。目前，360 安全浏览器已经开始支持国内多款 Linux 操作系统和国产 CPU，相信未来 360 安全浏览器能带给我们更多惊喜。



李沫南



张磊

作为 deepin 的首席技术官，张磊以“anything is possible”为题，向大家介绍了最新版 deepin 操作系统将要加入的 anything 模块。在此之前，也有很多用户表示对 anything 技术充满了好奇，张磊通过对 everything 的解析和对 anything 原理的解释为与会用户解开了它神秘的面纱。

李沫南是一位在开源圈摸爬多年的大佬，他脱离 PPT 的束缚，在台上以轻松愉快的方式向大家讲述了多年来在开源圈奋斗的经历。在这里，我们也有感而发，每个人都经历了从懵懂到能够披甲上阵的过程，在这个过程中，我们有过激情、兴奋，有过落寞、无奈，但好在，我们都不曾放弃，正是一份执拗和坚持，才成就了现在的自己。小小感悟希望与各位伙伴共勉。

deepin 技术总监张木梁在活动中向用户介绍了“好玩儿的 deepin”。deepin 通过多年的沉淀，除了拥有自主研发的桌面环境，更是研发了系列深度原生应用，从便捷办公的“深度远程协助”到“深度录屏”、“深度录音”等，都体现了 deepin 通过更简单的操作来便捷用户的初衷和原则。

活动还特别邀请到了 GNOME 基金会成员、前任北京 GNU/Linux 用户组主要负责人佟辉，他在活动中向大家做了“我如何带领女朋友进入开源世界”的精彩演讲，通过他引导女友 Mandy 慢慢加入到

开源中来的经过，号召更多女性朋友加入到更多开源活动中来。值得一提的是，Mandy 作为一名女开发者，参与了面向女性的 Outreachy 计划，该计划是今年秋季报名的最后阶段(截止日期10月30日)。

* 声明：本次活动，佟辉仅代表本人立场参加，不代表北京 GNU/Linux 用户组，由于会前沟通有误，特此订正。

接下来由 deepin 高级研发工程师王耀华做了“如何给一个进程加速”的精彩演讲。作为一个开发工程师，编码、调试和优化是永恒不变的话题，其中优化是一个比较高深的方面，但是同时也是非常有意思的一方面，本次分享了 deepin 的系统开发工程师是如何与系统优化这项任务死磕到底的、为什么创建了 deepin-turbo 这个项目、它是如何工作的、以及 deepin 的系统开发工程师在优化的过程中积累的一些经验和心得。

TUNA 是一支充满活力的团队，TUNA 会长姚沛然向大家介绍了 TUNA 的历史，目前，TUNA 可以提供开源镜像站、直播、弹幕等服务，同时 TUNA 正积极地参与到更多的开源活动中，也欢迎更多同学加入到 TUNA。

借助活动，我们认识了更多的小伙伴，再次感谢大家对 deepin 的关注，相信通过我们自身版本的迭代和广大用户的真实使用，deepin 将会成为国内更受欢迎的 Linux 发行版。d



2018 第八届深度开发者与用户大会

南京站

10月20日 / 南京 ICisC

>>>

南京站是一场国产软硬件拥抱开源世界的聚会，此次“聚会”我们邀请到了龙芯俱乐部创始人石南、向日葵远程控制 CTO 张小峰、上海 LUG 负责人杨伟、南京工业大学副教授孙冬梅、deepin 技术总监张木梁、deepin 高级研发工程师沈静、deepin 高级产品经理王珈等。

活动前，我们特别安排了展示台，展示了搭载深度操作系统 V15.7 的小米笔记本；以及基于深度操作系统龙芯版的台式机，及龙芯创客们使用智龙芯片做的机器人小车，得到了大家踊跃体验。

活动开始，由 deepin 工程师王明栋向在场小伙伴们介绍了 deepin 从一个三五人的线下小团体，逐渐成立公司进入商业化的一些小故事。deepin 在发展过程中，有过用户的质疑，也得到过开源朋

友们的援助；有过坎坷，也得到了一同奋斗的小伙伴们坚定的眼神；有过困惑，但也得到了全球用户的称赞。发展到今天，deepin 已逐渐成熟，也向国际开源社区贡献了大量的代码，相信 deepin 团队一定会不负所望，我们将不忘初心，砥砺前行。

接下来由上海 LUG 的负责人杨伟向大家介绍了上海 LUG，上海 LUG 是上海热爱开源的小伙伴们集结而成的民间用户群体，每周四晚，他们将会聚在一起交流传播使用技巧和技术热点，在上海的小伙伴们，可以通过：<http://www.shlug.org> 找到并联系他们。在这里我们也欢迎有更多热爱开源、热爱 Linux 的小伙伴加入上海 LUG。

龙芯俱乐部的创始人石南也为我们做了精彩分享，龙芯俱乐部是由龙芯首批个人用户发起的龙芯



爱好者社区。成员来自五湖四海、各行各业，目的是为了促进龙芯爱好者间的交流和龙芯的宣传，并以开源的方式推广龙芯技术和产品。多年来，龙芯俱乐部举办了多场开源活动。

Linux 作为日常使用的操作系统用起来如何呢？相信接触过的小伙伴们都各有见解，有人说他太难用，什么应用都没有；有人说他干净，用了 Linux 后世界才是自己的，那么 Linux 下的办公和生活应该是什么样的呢？deepin 技术总监张木梁以自己首次接触 Linux 到加入到 Linux 操作系统的工作中来的经历，向与会的小伙伴们做了精彩分享。

前几天 deepin 发布了与向日葵的合作，很多用户都为我们点赞，Linux 操作系统太需要各种应用的加入和支持了，活动现场我们也特别邀请了向日葵远程控制 CTO 张小峰为我们介绍了 Oray 生态圈和互联网开发，以及剧透花生壳、向日葵这两大产品的一些技术细节，活动中张总也与现场小伙伴们做了互动环节，通过远程控制张总办公室电话呼叫到现场的手机，来向大家展示了向日葵的真正魅力。

deepin 有个美女产品经理王珈，此次活动她向大家分享了 deepin 的很多功能设计在“出厂前”的各种纠结与决断，好的产品必然有好的产品思维在里面，deepin 在为用户打造一款产品时，设计思路可能会被推翻无数次，在设计部和产品部的多

次挣扎后，将会有多个设计方案的迭代和取舍，但请用户相信，我们一定是把最好的产品奉献给大家。

南京工业大学副教授孙冬梅老师为大家介绍了高校 Linux 教学与比赛，并围绕开源龙芯创客主板“智龙”这款基于国产龙芯以全开源方式推广的嵌入式最小系统主板为例，向大家作了介绍。孙老师也作为授课老师指导学生参赛，作为大赛指导在线上解答问题，在此，我们也希望南京高校的学生加入到全国大学生嵌入式芯片与系统设计竞赛中来。

近年来 deepin 与国产 CPU 进行了大量的适配工作，目前深度操作系统龙芯版、申威版都得到了广大用户的喜爱，活动中，deepin 的高级研发工程师沈静以“当开源软件邂逅国产芯片”为题，向小伙伴们做了精彩分享。deepin 与国产芯片的适配从 2014 年开始，至今已开发了基于龙芯、申威、海思在内的 12 个版本，支持了 14 款芯片，未来，deepin 也希望能够携手国产芯片，为用户提供更好的服务。

通过南京站活动，我们对国产软硬件的新发展有了新的认识，希望更多的优秀软硬件企业可以加入到开源世界中来，分享成果，收获更多用户的拥戴，再次感谢对本次活动提供支持的南京 ICisC、龙芯中科、龙芯俱乐部、开源中国、IT 大咖说。d

deepin



奔跑吧，Deepiner

● 深度科技 首席技术官 张磊 / 文

不记得是哪位名人说过，当一个人开始频频回顾往日时光的时候，就表示这个人开始变老了。所以出于怕死充嫩的本能，我一直不愿回忆过去，以避免暴露出油腻中年危机程序员的本质。但在《deepin 集结》编辑大人坚忍不拔、孜孜不倦地追稿下，拖延症晚期患者的我也不得不把自己的琐事从这几年的岁月中整理出来，以飨大家。

不过，其实我更想说的是关于我们这个团队的

事，而不仅仅是我个人。因为只有在和团队一起合作的过程中，一个人才会发现自己的短板、无力、以及爬上树后暴露出来的红屁股，也才会明白和一个紧密合作、充满活力的团队在一起奋斗，能创造出多少最初谁都难以想象的奇迹，并且能够对自己的成长带来多大的帮助。

说实话，在最开始接触计算机的时候，我就是个蒞弱。在那个古老的年代，我们使用的还是传说中的 Apple II，开机进去以后就是用 Basic 语言来

编程。时至今日，我唯一记得的就是每行代码都必须有标号，还有 PRINT、GOTO、RUN 等指令。但我当时对程序究竟是怎么跑起来的特别迷惑，导致编起程序来简直是一头雾水。

我至今还记得最后的考试题目是要打印一个杨辉三角形，要不是当时 dwd 同学的热心帮助，说不定就会打破本人中学生涯中的零挂科记录了。这是我第一次在编程上体会到团队的好处。

至于首次接触 Linux，那已经是在清华 26 号楼的时候了。

一次在走廊尽头的寝室里，某哥们正在噼里啪啦地敲键盘，屏幕上显示的不是熟悉的 Windows 窗口，而是一片黑乎乎的背景，上面绿色的字符在不停地刷屏。我好奇地看着这个当时还比较小众的系统，真的蛮有传说中黑客的感觉的。不过看上去用起来挺麻烦，都是手工输入命令，那我要是真用起来还不得要了老命了。

ATM 迁移

工作以后，东奔西走，不知不觉又转回到 Linux 都已经是好些年以后的事情了。在公司刚开始商业化不久的 2015 年，正好迎来了 ATM 机转用 Linux 操作系统的机会。

在这里我先介绍下背景，平时我们存取款的 ATM 机，使用 Windows 操作系统的占了 90% 以上，而其中一大半仍在运行 Windows XP。其实从技术层面上来说，Windows 与 Linux 都可以完美支撑 ATM 业务，但在实践中，要更换操作系统首先需要通过实例给用户以信心。

在 ATM 机系统中，前端系统可以大致划分为 SP 与 AP 两部分，其中 SP 主要负责进行设备管理（如出钞机芯、存款机芯、密码键盘等）；AP 则负责提供前台的用户界面与控制。因为在可行性验证阶段无法改动现有的代码，因此在此期间，深度团队决

定使用 deepin-wine 技术对 AP 进行迁移，但同时因为技术的限制，无法对设备驱动进行迁移，因此暂时只对部分设备，例如密码键盘（PIN）、读卡器（IDC）、打印机（PTR）等设备进行迁移。

整个迁移过程断断续续持续了约 4 个月的时间，我一直奔波在北京和杭州之间，其他负责开发的同事则是在武汉和杭州之间奔波，加班熬夜更是家常便饭。在这个期间，深度团队首先需要熟悉 ATM 系统现有结构与各部分的接口，其次需要适配完全没有接触过的外设，同时还要负责给前端界面使用到的 IE 各模块填坑，最后更要熟悉业务流程，以通过系统测试。

终于，在以 ch 同学为首的 deepin 团队的努力下，各设备都在深度操作系统上正常运行了起来，与此同时，读卡、查询等业务流程也都一一通过了测试验收。

我们的团队用短短几个月的时间，实实在在地证明了 ATM 机上操作系统迁移的初步可行性。

接下来我们需要和更多的干系方讨论，这将不再仅仅是技术的问题了，还需要对 ATM 行业现有的 XFS 标准进行相应的改造，在尽量保持兼容性的同时，我们需要将 XFS 标准与 Windows 解绑，这将涉及到多个厂商共计上万台 ATM 机的迁移改造规划。将近十个公司的人整整开会讨论了一周，终于得到了一个初步的标准草案和技术框架，涉及到 API、SPI、以及多个设备标准的改写。

其中深度团队负责 API/SPI 接口规范的修改，并承担开发对应的管理器系统，需要对下接口 SP，对上承接 AP。在其后的一个半月内我们发布了第一版，并在随后的两个月内使用增量迭代的方式快速开发，发布了八个版本更新，稳定了上下接口、功能与性能，修复了多个软件缺陷，并提供了多个演示程序与开发指导文档，给其他厂商后续的开发迁移打下了坚实的基础。要是没有 Iceleaf 和其它同事



加上的数千行 QML 代码，还有不断骚扰 sonald 学来的 Qt 开发知识，我一个人肯定累趴下了也搞不定的。

当然，后期的系统定制、内核 panic 问题的修复、涉及多个厂商的开发培训等工作的工作量更大。记得有一个问题是某个设备驱动会导致偶发的系统死机，但是驱动厂商又不愿意提供源码，ch 硬是把驱动强行反汇编出来，再将汇编代码手工转译最终查出的 bug，并给出了相应的解决方案。其他问题还有例如 SP/AP 不当耦合导致系统挂起、dbus-daemon 内存溢出、I 卡驱动导致屏幕撕裂等，zx 带领的深度开发和支持团队真可说是逢山开路，遇水搭桥，搞定了所有技术问题，这才终于修成了正果。

云打印

说到系统迁移，那真的是一个大坑。我们在建设生态上首先基于包最多的 Debian 来做，让大家可用的软件基数更大，其次自己也在不停地造轮子，开发了 DDE 等一系列软件，此外还主动联系搜狗、网易等大厂，向他们要授权，由我们来把对应的 Windows 软件重新开发，做出 Linux 版给大家做贡献，再就是把 Windows、Android、H5 等应用迁移过来了。但在 Linux 下有一个众所周知的问题，那就是外设驱动比 Windows 少太多，不说被 Linus 大爷树中指的 N 卡，就说打印机等办公设备，其实驱动也少很多。

在某个月黑风高的夜晚，deepin 同学突然有了个通过网络来解决打印机问题的想法，然后 andrew、zml、synh、wmd、我、叶子等同学开始盖楼灌水，分析 cups、ppd、postscript、samba 等一系列技术，果然是互相激发容易有想法，于是我一边陪着小朋友上奥数课，一边写原型，一晚上就用 shell+golang 做了一个特别粗糙的可用版本，成为了后来深度云打印的雏形，iceyer 小组则在短

短两三个月之内就做出了云打印产品，不仅功能丰满了许多，还顺手修了好多 bug，着实厉害，当然 zx 和 ch 团队后来推出的普适打印栈技术，那就更强了。

可以完全脱离网络支持原生打印，已经大大超出了我们最初最好的设想。

驱动优化

当然，做操作系统也免不了和内核、处理器打交道。其中，龙芯是国产领域的主要处理器之一。在去年的某个项目里刚好接触到了龙芯 3a1000，该系统有个关键的指标就是数据处理的吞吐量。

在之前的系统上，数据处理的吞吐量是每秒 500Mbps，由于刚开始无法接触到驱动程序源码，只能根据系统表现与测试程序的代码对可能产生影响的系统参数进行调整，包括编译参数例如：CONFIG_HZ、CONFIG_NO_HZ、CONFIG_IRQ_TIME_ACCOUNTING，运行参数例如 sched. 下的各参数，测试程序使用的内核函数如 msleep、schedule、usleep_range 等，而内核代码更是从 2.6.32 一直追踪到 3.10……

看材料、读代码、插桩、测试，中间 haitao 负责构建系统和集成各方补丁，我则负责写片段代码，



两个人一起跑客户、调设备和熬夜加班，终于在一个多月后，把问题定位到了。原来，设备驱动程序在从外设到内部数据处理之间的数据转换环节使用了一个低效的 C 语言实现，但实际上在内核中有更高效的指令集实现，仅需要修改一行代码，就能把系统吞吐性能从 500Mbps 提高到了 740Mbps，当我们告诉设备厂商和用户问题原因和解决方案的时候，大家真感觉神了，原来只需要这么小的改动就能提高 50% 的性能。当时我的心里确实是挺高兴的，不仅是因为解决了一个悬疑已久的技术问题，还因为真的是帮到了所有人，而不是技术自嗨。

文件快搜

在写程序方面，我应该算是一个 oldschool，如果硬要在程序漂亮的界面和更高的运行性能之间做一个选择的话，十有八九我会选择后者。这导致在深度内部曾经流传过一个段子，说是我喜欢用 Windows 记事本编程，天地良心，那玩意儿连语法高亮都没有，也忒挫了吧，我最多也就是用 vim 或者 notepad++ 来写程序而已……

去年刚好在某项目期间有个空隙，终于有时间可以琢磨下怎么提高 Linux 下文件名搜索的速度了，由于之前就了解过 Windows 下类似的软件 everything 和 Linux 下的 rlocate，又抽时间写了几个 kprobes 的 demo 程序热身，我就这样开始写 anything 了。

由于几年前就搞过搜索引擎，所以一上手就直接用上了一个简单的切词，然后是基于哈希链表的倒排索引，加上用 kprobes 勾住了 VFS 的相关内核函数监听文件系统的变更，再设计一个线性树保存文件树，经过 hualet 和 synh 对代码与文档的测试与 review，三周内我就出了一个 anything 的原型。

在 cli 界面下测试，和 find 比起来，简直跑得

飞快，两者的时间比大概是 10s : 0.1ms，相差 4~5 个数量级。接下来我又在 deepin 同学的催命下，把倒排索引改成了直接基于线性树的搜索，仍然能保持 3 个数量级的性能优势，匆忙结束了原型开发后，我转过头来又要继续接着赶项目。

当然，这个原型确实也挺粗糙的，zccrs 和肥猫修复了包括内核版本兼容、缓存区越界、缓存区动态扩展导致地址错误等好几个 bug，anything 才达到了产品质量，能真正融合进文件管理器中，也终于在今年和大家见面了。

其实我自己心里不免有点小得意，不过反过头来看，还是有蛮多东西要继续完善的，比如要把 kprobes 改成 ftrace 减少对于内核不可睡眠的要求，把内核模块改成 eBPF 减少对内核的依赖，或者把索引全落盘提高软件的伸缩性……就像千年前的湖北人屈原说的，路漫漫其修远兮，吾将上下而求索。

软件的优化与重构是永不停歇，永远可以做的更好的。

尾声

又一位伟人曾说，你的知识边界就像一个圆，随着知识的增长，你接触到未知疆域的边界也在不断增长，从而使得你谨记保持谦逊。其实何止知识如此，在深度，接触到了这么多高手，sonald、synh、ch、肥猫、hualet、iceyer、zccrs、zx……

平时低调的他们让你感觉不出什么来，只有遇到问题一筹莫展，并在他们的帮助和激励下解决了问题的时候才会恍然大悟，果然是“低调为王”、“高手在身边”，要不然，是善战者无赫赫之功？

很庆幸能和这些战友为伍，一个人单独可以跑得很快，多个人一起则可以跑得更远。那么，就让我们一起奔跑吧，deepiner。d



中国软件业正成为开源生态贡献者

◎ 张汉青 北京报道

国内 IT 圈曝出的红芯事件，引发了市场和公众的极大关注与思考。

8月28日，中国工程院院士倪光南在接受《经济参考报》记者采访时称，红芯事件并未在公司的道歉声明之后告一段落，持续不减的热度反而使国内开源软件生态险遭殃及池鱼。看来，有些问题有澄清的必要，以免造成将开源软件与创新对立起来的误解。

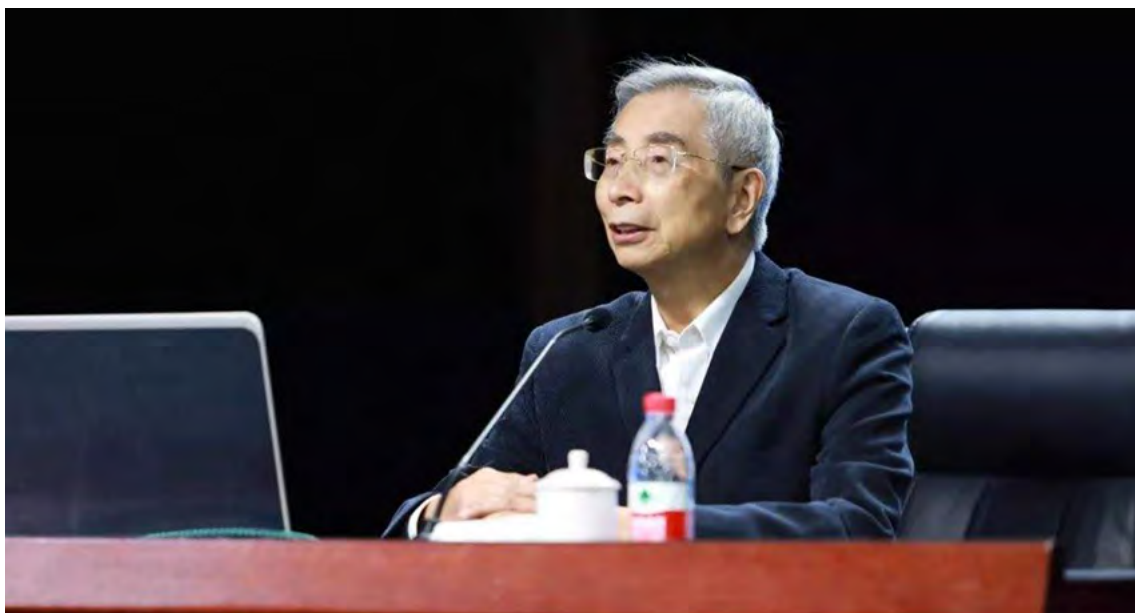
多部委致力推动开源生态的建立

由于可以大大节省软件开发的时间、人力成本，并有利于技术的推广和发展，开源早已成为全球

IT 行业公认的重要创新方式。

从 Linux 进入中国普遍被视为中国开源产业的起点，至今已有近 20 年时间。较长一段时间内，起步较晚的中国企业普遍是开源软件的使用者。诸多互联网企业、软件和硬件厂商、应用方案提供商、运营商、芯片厂商、学校和研究机构等等，都是开源软件的应用者、参与者与贡献者。

而且，在国家层面，科技部、工信部、发改委也均已致力于打造和推动开源社区、开源平台、开源基金会等开源生态的建立，仅 2018 年，三部委就在云计算、大数据、人工智能、工业互联网、



数字经济等行业启动了大量与开源相关的重大专项、重大工程。

从 2008 年之后，移动互联网的爆发使得国内诞生了数十家在 Android 开源基础上定制操作系统的终端厂商，以及上百万的 Android 开发者持续开发移动互联网应用，中国软件企业和开发者的参与使得 Android 生态在国内迅速成熟并发展壮大，逐步发展成为全球最大的移动互联网生态体系。

倪光南表示，在批评红芯的夸大宣传时，不必回避其在 Chromium 之开源基础上做出的隐盾、云适配、支持国密算法等安全方面的改进；而在讨论开源软件与自主创新时，更不能回避中国软件业目前整体正在从开源生态的应用者向参与者、贡献者的方向发展的事实。

许多企业完成从参与者向贡献者升级

“近年来，随着云计算、大数据、人工智能、物联网等新一代信息技术蓬勃发展，中国企业一直在不断增加其在世界开源软件界的贡献度及话语权，许多中国企业已经完成了从参与者向贡献者的升级，成为一些开源社区的重要成员、关键成员，发挥了骨干作用。”倪光南表示。

自 2006 年开始，华为就鼓励使用开源软件和开源社区开发模式，经过长期努力和发展，目前已经在多个开源社区成为核心贡献者。

在 LinuxKernel 社区，华为已经贡献了超过 1000 个补丁，在 400 多家贡献者中，华为排名前 30 位；而在云计算开源项目 OpenStack 社区中，华为拥有 16 名核心贡献者，贡献度排在全球第六。根据 OpenStack2017 年报统计，OpenStack 目

前拥有 8.4 万名社区成员，其中来自中国的成员数量排名第三，华为是其中贡献度最高的。

中国信息通信研究院在 2018 年 3 月份发布的《中国云计算开源发展调查报告(2018年)》中指出，在使用云的企业中，80% 已经应用了开源技术，60% 的企业正在应用 OpenStack。目前，中国云计算市场规模已经超过 700 亿元，市场的繁荣发展推动开源社区规模持续壮大，中国开源云计算生态已有一定规模。

除华为之外，国内互联网巨头也早已在多个开源社区成为主要贡献者，并且不断将自身技术成果向全球开源。以腾讯为例，自 2010 年开始拥抱开放战略至今，腾讯已经凭借其贡献度成为 LinuxDeepLearning、OpenStack 基金会的白金会员，Linux、CNCF 基金会的黄金会员。过去一年中，腾讯在 Hbase、Hadoop、ceph、Spark 等开源社区贡献了 150 多个补丁。同时，在腾讯云、游戏、AI、安全等领域，腾讯已经对外开源了 56 个项目，其开源项目在 Github 上已经累计获得了 14 万的点赞。

与腾讯路线类似，阿里巴巴也已经对外开放了 150 个开源项目。值得一提的是，BAT 三巨头目前均开源了自己的深度学习平台，以期望通过开源模式在至关重要的 AI 领域增强国际话语权、影响力。

除此之外，中国软件人员目前还主导了诸如麒麟操作系统、深度 deepin 操作系统、SylxOS 操作系统等开源社区，其中，SylxOS 是由我国科技人员从 2006 年开始自主编写的工控实时操作系统，已经在航空航天、国防军工、机器人等重要领域广泛应用，且经工信部赛普评测中心评估拥有 90%



左右的内核自主化率。

根据开源社区统计，目前由国人发起的开源软件已达 9719 款。开源软件已经不仅仅是中国企业在国内的创新模式，也是在全球提高技术、商业影响力的主要路径。

上面关于我国开源软件和生态发展的简单回顾足以证明，中国软件界早已走出简单应用开源软件的阶段，不能因为出现某些事件，而一笔抹杀中国软件业运用开源模式进行大量创新（包括原始创新、集成创新、引进消化吸收再创新等类型）的客观事实。贬低开源软件，说基于开源软件只能做“二次创新”是没有根据的。

开源软件在满足自主可控方面颇有优势

倪光南认为，开源软件在满足自主可控要求方面也颇有优势，尤其是与外国闭源的专有软件相比。

以桌面操作系统为例，像 Win10 那样的闭源操作系统会存在被监控、被劫持、被攻击、被禁售、密钥和证书失控、无法加固、无法打补丁、不支持国产 CPU 等等安全风险。相比之下，基于开源 Linux 研发的国产操作系统的主要风险是被病毒、木马所攻击，其安全风险远低于 Win10。另外，

与相当于“黑盒子”、用户无法改进安全性的闭源软件不同，使用开源软件可以自己分析源代码、通过自己打补丁等方式增强安全。

回到红芯的问题上来。人们批评红芯是以为红芯借自主研发之名骗取国家经费，但实际并没有；红芯能获得资本投资也主要是因为它在移动互联网、网络安全方面做出的改进。

不过，以前很多人对开源问题认识不足，例如上述许多基于 Android 定制的移动操作系统，除了华为、小米等进入国际市场的那些大公司，能恰如其分地称自己做的只是“界面（UI）”外，很多公司都自称做了“自主知识产权”的操作系统，他们犯了和红芯类似的错误，但以往他们都没被追究、都被宽恕了，所以现在出了红芯，追责也要恰当。试想，如果人们对这类错误从一开始就一视同仁地严加追究，恐怕红芯也不会重蹈覆辙了。

至于“汉芯”事件，则是一起罕见的性质严重的欺骗国家科技计划的行为，将红芯基于开源的创新成果进行过度包装等同于“汉芯”事件显然不妥。

倪光南表示，从网络安全的角度考察，客观上，红芯基于开源软件再做一定的改进，安全性可能比外国不开源的专有软件还要好些。现在正在推行的、包含“自主可控测评”的“多维度测评”很容易对它的创新成果进行客观评价。与此相比，对网络安全威胁更大、欺骗性更强的是那些给不可控、不开源的外国专有软件“穿马甲”的行为。有人将“响当当”的国产品牌为这类软件做掩护，欺骗性很大，如果不及时发现，危害性很大。为此，更需要实施“自主可控测评”来抵御这种“穿马甲”的风险。d

邬江兴院士：新型网络技术发展思考

© 邬江兴，中国工程院院士，工业智能化、中国科学信息科学

面对互联网与经济社会深度融合发展带来的专业化服务需求，现有互联网基础架构及由此构建的技术体系在智慧化、多元化、个性化、高鲁棒、高效能等方面仍面临一系列挑战。邬江兴院士在《中国科学：信息科学》“观点与争鸣”栏目最新发表的文章，分析了新型网络技术创新的基础，明确了新型网络的基本技术特征，以全维可定义的开放架构为主线、以基线技术重塑为切入点，探讨了基线技术重塑下新型网络的基础架构和核心机理，为互联网技术创新与突破探索了可能的方法和途径。

引言

随着网络技术和应用的不断发展，特别是大数据、云计算、人工智能等的出现和运用，互联网迎来了加速裂变式的新一轮革命，它促使社会各方面发生颠覆性变革，并深刻改变着人类世界的空间轴、时间轴和思想维度。

互联网业已成为与国民经济和社会发展高度相关的重大信息基础设施，对提高社会生产力、助推经济社会升级转型、创造新的就业机会等均具有深远影响，为加快建设创新型国家，实现“网络强国”和“智慧社会”提供了强大的基础支撑。

近年来，随着互联网应用外延的迅速拓展，“如何充分用好网络”的问题已基本解决。然而，面对互联网与经济社会深度融合发展所带来的专业化服务承载需求，互联网技术内涵的发展却未能充分支撑网络应用外延的拓展，现有网络基础架构及由此构建的技术体系在智慧化、多元化、个性化、高鲁棒、高效能等方面仍面临一系列重大挑战，制约了其在

更广更深层次上支撑经济社会发展。

“什么样的网络更好用”成为互联网当前亟待解决的核心问题。

最近十余年来，新型网络技术始终是全球学术界和产业界关注的焦点，广大科技人员为此提出了各种解决思路及相应的技术和方案，已在多种场景下初步应用并展现出强大生命力，特别是基于开放架构的软件定义网络(SDN)和网络功能虚拟化(NFV)等技术蓬勃发展，预计在未来5~8年内将逐步成为信息基础设施的主流技术架构，并进入成熟应用发展阶段，为互联网技术的持续创新与演进消除了“壁垒”。

在强大的外部需求牵引和自身技术发展的双向驱动下，当前互联网技术体系的坚冰正在被打破。在推动互联网增量式部署和演进式发展的同时，充分吸收和利用新思路、新方法，对互联网技术体系进行基础性变革，加强互联网技术架构创新，促进技术研发由外挂式向内生性转变，构建智慧化、多元化、个性化、高鲁棒、高效能的新型网络，将是互联网下一步发展趋势。

对于我国互联网发展而言，后发劣势使得我们在跟跑过程中付出了沉重的经济代价、时间代价和机会代价，相关产业饱受缺乏核心技术自主掌握带来的发展限制。因此，在这次新型网络发展的变革期，抓住机遇谋求更深层次、更广领域的发展，推动我国互联网技术由跟跑到领跑，实现由“通信产业大国”到“通信技术强国”的历史性转折，是新时代互联网科技工作者的崇高使命。



本文分析了当前网络发展面临的主要挑战，以全维可定义的开放架构为主线，以基线技术重塑为切入点，阐述了新型网络应具备的基本技术特征和核心机理，为互联网技术创新与突破探索可能的方法和途径，供广大科技人员参考。

互联网发展面临的主要挑战

随着互联网与经济社会深度融合发展，互联网+、工业 4.0 等逐渐成为国民经济命脉领域的新支柱。互联网在当前社会中扮演的角色日益增多，使得用户对网络的专业化、个性化需求不断提升；多元化终端类型、接入方式不断发展，人-人、人-机、机-机、网-网通信等成为常态，要求网络必须为海量业务提供多元、个性、智慧、高效、鲁棒等服务。

面对上述需求的演进变化，现有互联网的技术内涵与外延发展存在严重的不充分不平衡，无法满足泛在场景下各类型各层次用户对美好用网体验的需求，使得当前网络对质量、安全、融合、扩展、可管可控、效能、移动等的支持能力低下，在应对巨复杂用网需求时面临巨大挑战。

主要挑战之一：封闭的网络架构和刚性的基线技术，限制了网络多元化发展，制约了网络专业化服务，禁锢了网络创新技术的应用。

回顾互联网几十年来的发展历史，其始终采用基于 IP 瘦腰模型的封闭架构以及刚性的传输协议、路由控制和转发方式等基线技术，能力简单而薄弱，网络功能体制单一，对泛在的信息服务、多样化和全方位的网络业务、确保质量的通信效果、安全可信的信息交互等的支持能力已经严重不足。

追溯这一现象或者模式的成因可以发现，封闭网络架构和刚性基线技术之所以成为现有网络的选择，并非技术因素驱动，而是市场机制作用的结果。在封闭模式下，基线技术一旦占据主导地位就会不断强化。不可否认，在封闭网络架构和刚性基线技

术下，网络按照简单的用网需求、有限的业务类型和单调的用网场景构建与之匹配的运行机制，在发展初期为互联网快速普及和成熟提供了强大的支撑和有力保障。

然而，随着互联网作为全球基础设施地位的确立，其多元化、专业化服务需求不断提升，以尽力而为传输和统计复用为代表的网络资源供需匹配和调度机制存在先天不足，而重大的基础技术创新又始终无法在网络上得到实施，使得网络“能”与“效”之间的矛盾始终存在且无法得到根本性解决。这种窘境，使得试图靠被动的增加网络资源或技术补丁来实现网络“能”力增长，不一定带来“效”果的增加，网络越来越臃肿已成为不争的事实。

主要挑战之二：互联网的多元化发展，导致当前僵化的网络运行机制下传输控制、资源管理、配置维护等复杂性倍增，网络效率低下，用户体验差。

当前，互联网已经从早期非实时业务承载为主，发展到目前以文本、图像、语音、视频综合内容承载为基础，并逐渐成为连接现实世界中各种生产、生活要素的基础方式，呈现出终端类型多元化、接入方式多元化、设备形态多元化、业务场景多元化等趋势。

然而，现有互联网的运行方式僵化，其单一固定甚至基于人工的配置管理和调度机制等已无法满足网络多元化发展所带来的泛在用网场景需求，导致网络传输控制、资源管理、配置维护等复杂性倍增，网络运行维护开销巨大。同时，现有网络中用于刻画网络功能、性能的指标和模型，以及由此构建的匹配用户业务需求的方法，也已无法适应泛在场景用网需求，使得网络效率低下，用户体验差，面临深刻变化。

如何使网络具备面向泛在用网场景需求的“无人驾驶”能力，在结构优化、资源分配与管理、业务承载与服务质量保障、网络运行监测与故障定位

恢复等方面逐渐摆脱对人力或僵化运行机制的一味依赖，充分吸收和利用人工智能等技术的发展成果，降低网络传输控制、资源管理和配置维护的成本，同时使得网络具备自我发展能力，改善复杂网络环境下由人为认知局限造成的网络运营低效等不利局面，引导和升级传统简单粗放的网络资源管理和运营模式，在新型网络技术创新中必须着力考虑。

主要挑战之三：网络节点或链路随机性失效、网元系统包含漏洞后门等不确定扰动事件频发，使得互联网的广义鲁棒性问题日趋突出。

网络规模快速膨胀及各种网络要素的巨复杂性，使得互联网运行中经常会受到不确定性的干扰或者破坏，从而导致其性能降低甚至功能丧失，无法满足用户需求。

事实上，互联网目前尚缺乏为其软硬件系统提供抑制包括漏洞后门等“暗功能”在内的广义不确定扰动能力，所以原本属于目标对象或网络元素设计和制造过程中的质量控制问题，就万般无奈地“溢出”成为网络空间最主要的安全挑战，互联网的“潘多拉魔盒”由此打开。生产厂商或企业“不承诺软硬件产品安全质量”或“不对产品安全质量引起的后果承担任何责任”的行为，都可以归结为网络广义鲁棒性问题所致。

因此，在网络的可靠性、可信性均无法保障的生态环境中，恢复产品质量神圣承诺和商业信誉，治理被严重毒化污染的网络空间环境，将“魔鬼”关回到潘多拉盒中，研究使网络既能抑制节点或链路失效等不确定失效扰动，也能防范系统后门、漏洞等不确定性威胁扰动影响的广义鲁棒控制技术，保持满意的服务品质，增强网络对各种破坏意图的抵抗能力，是新型网络面临的另一重要挑战。

新型网络技术创新的现实基础

应该指出的是，学术界和产业界对新型网络及

其支撑技术发展已关注多年，国内外相关研究和实践成果颇丰，这为打破网络发展困境、创新新型网络技术提供了良好的现实基础。其中，软件定义网络(SDN)/网络功能虚拟化(NFV)等为新型网络技术创新提供了开放环境基础，多样化寻址、人工智能、广义鲁棒控制等技术为新型网络技术创新提供了核心要素基础，而“硅红利”和“光红利”的充分释放为新型网络技术创新提供了实现基础。

1、SDN/NFV 等为新型网络技术创新提供了开放环境基础

近年来，以 SDN/NFV 为代表的开放可编程网络及相关技术蓬勃发展，其通过转发与控制分离机制对计算、存储、网络资源进行灵活调度和管理，使得网络中链路、路由、流量等按需调度，并基于功能可重构、可编程等实现了网络开放、可扩展和自演化能力，从而提升了网络传输效率并优化资源配置等。其中，开放可编程是对网络整体功能和行为的高度抽象以及软件编程的自定义，其核心思想是通过在网络节点提供开放可编程接口，利用编程语言向网络设备发送强大的编程指令，实现对网络功能和行为的按需管控和新业务的快速部署。网络功能虚拟化(NFV)则强调通过将这些功能部署到虚拟化的网络资源上，使得网络在灵活性、动态资源扩展、能量效率等方面更具优势。同时，网络功能虚拟化也支持虚拟化资源与物理资源混合的场景，具有广阔的应用前景。SDN/NFV 等为开放架构和可编程技术创造了网络技术创新和试验环境，使得新型网络具备了由封闭架构到开放架构彻底转变的技术和环境基础。

相关研究方面，国家已经支持了“新一代互联网体系架构与协议研究”、“面向服务的未来互联网体系结构与机制研究”、“可重构信息通信基础网络体系研究”、“软件定义网络(SDN)关键技术



研发与示范”等研究工作，并启动国家重大科技基础设施项目“未来网络试验设施”，从体系架构、关键技术、运行机理和组网试验等层面对开放网络架构技术进行了探索。其中，我国独创的可重构网络技术，明确提出了打破传统网络技术“封闭”和“刚性”束缚、创造开放架构的未来网络体系，释放技术创新的活力，全面支撑新型网络技术的创新。

工程实践方面，从大型运营商到数据中心和各类专用网络，都已经开始规模化应用 SDN/NFV 来提升自身的网络能力。在国内外各大运营商制定的未来发展战略中，均明确将 SDN/NFV 作为其未来几年网络建设的主要技术基础。此外，我国已开通了覆盖十六个城市的基于新一代网络体系架构创新试验设施，在软件定义路由交换设备、信息资源智能调度系统等方面取得了突破，形成了一批具备自主知识产权的成果。

2、多项关键技术的突破为新型网络技术创新提供了核心要素基础

首先，软件定义一切的理念从理论走向实践。以软件定义转发 (SDF)、软件定义互连 (SDI)、软件

定义硬件 (SDH)、软件定义协议 (SDP)、软件定义芯片 (SDC) 等为代表的软件定义技术蓬勃发展，可实现对基础网络的拓扑、协议、软/硬件、接口等进行全维度定义，从而为多元化、个性化应用提供了精细化、可定义的网络组件和服务。软件定义思想在互联网领域内的全链条应用和实践，支持为不同类型的通信主体构建满足其个性化需求的服务承载网络，支持现有典型协议以及各种新型协议的快速部署，有望提高网络资源利用率，降低维护成本，并带来网络架构演进、设备形态变化和组网运营模式等变革。

其次，多样化寻址和路由技术创新酝酿重大突破。现实世界呈现出以邮政编码、门牌号为基础的传统寻址模式和“以服务内容搜索为中心”、“以空间坐标定位服务为中心”等新型寻址模式多样化发展的局面，并展现出高效多样化服务的强大生命力。与物理世界强关联的网络空间也已呈现出与之匹配的多样化寻址与路由技术创新。尽管在未来相当长一段时间内以 IPv4/v6 为基础的寻址和路由方式仍将持续发挥重要作用，但近年来以内容为中心的寻址路由方式、以空间坐标位置为中心的寻址路由

方式、以标识分离为中心的寻址路由方式等技术发展迅速，已在现实网络中初步应用并取得良好效果。

第三，面向泛在用网场景需求的网络“无人驾驶”技术研究活跃。近年来，网络大数据分析、人工智能技术等蓬勃发展，在此基础上建立网络自我驱动的智慧化运行机制，成为学术和产业界探索的重点。Clark 等提出网络“知识平面”概念，建议基于人工智能与认知系统来实现网络的自配置、自适应、自修复。Mestres 等提出了知识定义网络的概念，动态监测网络状态并基于机器学习算法进行分析决策，进而优化网络配置和性能。国内外主流设备制造商和互联网企业如华为、思科、中兴、谷歌、阿里等也纷纷着力探索复杂网络的智慧运营方法并取得了初步效果。

最后，拟态构造技术为解决网络广义鲁棒控制问题提供了切实可行的途径。网络广义鲁棒控制问题本质上可以对应为面对漏洞后门攻击时的静态性、确定性和相似性之构造缺陷问题。利用网络服务功能与视在结构的不确定性关系，在信息系统或处理装置中运用功能等价动态变结构技术，无疑可以扰乱或瓦解基于目标对象漏洞后门攻击链的稳定性，以视在的不确定目标场景应对网络空间安全威胁。基于上述认知，我国科学家和科技工作者创新性地提出了拟态构造技术并已在文件存储、路由交换、域名服务等进行设备开发与试点应用，为在信息系统软硬件供应链可信性不能确保的全球化生态环境下，运用创新的系统构造技术解决网络广义鲁棒控制问题开辟了一条新途径。

3、“硅红利”和“光红利”的充分释放为新型网络技术创新提供了实现基础

芯片与器件水平是网络技术赖以发展的物质基础之一。近年来，“硅红利”和“光红利”持续释放，不断提升网络核心元器件水平，推动新型网络基础原理创新和核心设备系统研制，为新型网络技术创

新提供了实现基础。

在“硅红利”方面，随着网络核心元器件的工艺水平不断提升，网络核心设备的计算、存储和传输能力大幅提高，不仅为网络新技术的部署和实施提供了重要支撑，而且使得新型网络技术的创新逻辑得以拓展。在此基础上，网络演进已经不再局限于“传输”能力的单边提升，而是全面发展为基于“传输、计算、存储”一体化的融合提升模式。通过存贮、计算与传输能力间的动态转换和组合应用，显著提升了网络服务能力。

在“光红利”方面，光波长处理、多芯光缆等方面的技术进步，推动了光纤和光器件不断刷新更高性能纪录，使得新型网络技术能够具备更多的可用带宽，通过更多的带宽资源总量达到更强的网络服务能力。这就使得新型网络技术的研究能够聚焦于网络核心机理和核心逻辑，从而能够在更深层次和更基础的环节来解决网络发展面临的核心问题。请加微信公众号：工业智能化 (robotinfo) 马云都在关注。

新型网络的基本技术特征

基于上述认知，我们认为新型网络应该具备全维可定义、多样化寻址路由、智慧化和广义鲁棒性等基本技术特征，并以这些特征来分别确定新型网络的“开放基因”、“功能基因”、“效能基因”和“鲁棒基因”，以基因特有的内生作用为新型网络发展提供持续演进动力。

1、以全维可定义确定新型网络的“开放基因”

随着网络业务形态不断丰富，业务对网络的需求越来越多样和多变，而传统网络架构的刚性和封闭特性，导致其服务能力是有限的且确定的，这就直接导致了业务需求与网络固有能力之间的差距日益扩大，难以甚至不能支持不断演进发展的网络业务需求。



新型网络的首要技术特征就是打破传统网络的刚性、封闭架构，能够对开放架构下基础网络的软/硬件、协议、接口、芯片等进行全维可定义，从而使网络的运行机理不再受制于单一功能或技术，使得网络具备多维资源的柔性组织和适配能力，在服务灵活性和业务适应性上满足多样化业务需求，并自然适应未来业务的复杂不确定演进。

因此，新型网络将通过全维可定义来确定其“开放基因”，在全维可定义的基础上支撑网络架构“开放”，改变当前网络固化的运行模式，由其作为“网络功能构件市场”机制的提供者，允许多样化应用动态装配、构建与之匹配的定制化服务构件链，动态部署个性化应用，实现全网业务的灵动适配。

在开放网络架构下实施全维可定义，新型网络将不再存在基线技术垄断的风险，内在的开放性使得各个层次的创新都能够有效开展。网络的功能和工作机理将不再受制于具体的协议，其服务能力不再依赖节点的初始植入设计，其服务能力空间将超越已知协议和机制的限制，从而实现由已知协议定义的网络到需求驱动的网络形态和服务能力转变，充分满足新型网络不断演进的业务需求。

2、以多样化寻址路由确定新型网络的“功能基因”

当前，IP 是传统网络的唯一寻址方式，无法满足多样化、多元化、专业化的应用需求。“千军万马过 IP 独木桥”的同质化格局，使网络世界因为失去多样化的内在活力而演进缓慢。

事实上，自然界物种多样性为构建和谐多元化的网络空间共同体提供科学的理论依据。自然界的多元化物种之间因为其相互补充、相互制约、相互平衡而构成一个和谐统一的发展格局。如果没有竞争性变异的累积和推陈出新的延续，生物界会因为同质化而逐渐走向凋亡。

因此，新型网络应以多样化寻址路由重新确定其“功能基因”，在基础网络功能上体现出多元融合发展，并由此构建出基于基因信息表达的体系化网络功能生成和按需演化机制。同时，基于多样化标识的寻址路由技术也将为新型网络基线技术创新提供功能基础环境，为多元化业务需求提供资源深度融合使用支撑。通过多样化标识寻址空间的协同技术创新和优势互补，以先天内生方式解决现有网络存在的诸多弊端，有效提升网络服务能力、安全性、移动性、资源利用率等。

3、以网络智慧化确定新型网络的“效能基因”

网络智慧化的目标是，在全维可定义和多样化寻址路由的基础上，以网络传输效能、节点运行效能、业务承载效能和服务提供效能等为约束，建立“感知—决策—适配”一体的自我驱动运行机制，实现网络资源管理和传输控制的智能适配以及网络运维的自动化，使得用网便捷且网络能够在无感的情况下随着环境和用户需求的改变而智能调节。

将网络智慧化确定为新型网络的“效能基因”，使得“用网过程”和“用网体验”不再是用户所关注的问题，将用户的关注点归结为“用网目的”这一简单二元问题。此时，网络可在海量的用户、网元和业务之间进行适配协调，根据用户用网目的直接决定网络服务的提供方式，并智能动态地适应用户需求变化，从根本上释放网络潜力。

网络智慧化将充分借鉴人类社会、生物机体内部运行以及生物群体智慧等领域的研究成果，拓展人类智慧在网络构建和运行中的应用深度和广度，提供“源于现实世界启迪，创造高于现实世界”的体验。具有智慧化特征的网络，在网络构建时能够从近似于人类自然语言描述的需求出发，自动推演出合适的网络资源组成、节点协作模式、运行控制模式等，并且能够根据实际运行效果进行自我调整

和优化；同时，在满足用户和业务需求方面，具有智慧化特征的网络也能够通过自身积累“经验”，基于智能感知、大数据和人工智能等技术，促使网络在智能满足现实需求方面不断优化服务效果和服务性能。

4、以广义鲁棒控制确定新型网络的“鲁棒基因”

新型网络必须具备广义鲁棒控制构造的“鲁棒基因”，不仅能够有效抑制目标对象内部传统的不确定扰动影响，也能够基于漏洞后门等人为扰动下维持系统服务功能和性能的鲁棒性，从而在很大程度上抵消网络元素广义鲁棒控制功能缺位对互联网服务品质造成的负面影响。

面对新型网络的广义鲁棒控制需求，仅仅使攻击效果不确定或者只能在不同程度上瓦解不确定扰动因素并非防御者的终极诉求。理想目标是，无论对已知风险还是未知威胁导致的确定或不确定扰动效果，都能被某种构造或者机制变换为一种概率可控的可靠性问题，以便借助成熟的可靠性理论和方法统一解决。

因此，为网络植入广义鲁棒控制基因，在网络与平台设计中导入鲁棒控制机制，设计完整的网络鲁棒控制架构，使网络既能抑制节点或链路失效等不确定失效扰动，也能防范系统后门、漏洞等不确定性威胁扰动影响，实现“网元鲁棒构造、网络鲁棒控制、服务鲁棒提供”，是新型网络实现稳态服务提供的必然要求。

基线技术重塑下的新型网络核心机理

在新型网络体制下，我们认为基线技术将不再是传输协议、转发模式和路由控制这一传统封闭环境下的刚性桎梏，而是在全维可定义的开放网络架构基础上重塑为多样化寻址与路由、网络智慧化和广义鲁棒控制。文章分别从基础运行环境、多元功

能组成、服务提供方式和运行保障机制 4 个方面讨论基线技术重塑下的新型网络核心机理。

1、全维可定义的开放网络架构

以 SDH、SDI、SDF、SDC 等为代表的全维可定义技术，是支撑开放网络技术持续创新的基础。在全维可定义的开放架构下，新型网络以可定义的构件和网络柔性化组织为基础，建立网络连接、硬件、协议、转发等全维可定义的基础结构，实现基础连接、节点、网络等层面全链条可定义。

首先，为了保证现有业务和网络设施能够适应网络的演进，开放架构下的全维可定义构件及其运行平台必须具有良好的稳定性，即构件能够在保持新的网络协议或应用增量部署的同时也能够保证原有应用的正常运行；

其次，为了保证网络能够在多维评价指标上呈现良好的变化以满足业务的变化以及新型应用的部署，构件必须具有可变化的内在结构，也就是说包处理的方式以及网络协议的运行方式可以动态改变；

最后，在构件结构可变的基础上，构件要能够以某种“柔性”的方式对其结构进行调整，进一步地，柔性是构件针对应用要求对其内在结构、资源做出的隐性调整，以实现网络服务效果对应用需求的动态、紧密跟随。

2、多样化寻址与路由

多元化的网络应用需要多样化的寻址与路由方式，以及高效灵活编址与路由的融合发展。具体而言，我们认为新型网络的多样化寻址与路由机制应包括但不局限于如下典型方式。

以 IPv4/v6 为基础的寻址和路由。该方式采用等级地址模式、地址自动配置、源认证等技术，具有较强的灵活性和快速处理能力，在新型网络中仍将持续发挥重要作用。



以内容标识为基础的寻址和路由。该方式将网络通信模式从关注“where”转变为关注“what”，将网络中的一切数据内容都看作是可以传输的信息，实现了直接以内容互联的方式而非主机互联。

以身份标识为基础的寻址和路由。该方式通过身份与位置分离、资源与位置分离、接入与核心分离，综合有效解决安全性、移动性、可扩展性、用户体验等问题。

以空间坐标位置标识为中心的寻址和路由。该方式基于地球剖分网格进行网络位置编码，可实现网络空间与现实空间位置的直接映射，为提升网络应用效能提供支撑。

3、“感知—决策—适配”一体的智慧化传输与管理

新型网络通过引入群体智能、人工智能等技术，建立“感知—决策—适配”一体的网络智慧化传输与管理机制，采用网络资源智能协调控制技术、智能传输优化技术和业务智能适配的服务承载技术等，实现面向用户体验的大规模网络智慧化管理与传输。

网络的“感知—决策—适配”一体的网络智慧化传输与管理包括：

(1) 感知：通过泛在互联、可定义感知等技术，动态、实时感知网络业务与网络资源分布，并基于高层感知语义的统一描述模型生成全网视图，支持网络感知对象、感知动作、关联规则和功能分配的可定义；

(2) 决策：针对网络状态复杂、流量行为多变、业务模型不确定等特点，进行复杂不确定业务与资源间的拟合决策生成，实时决策网络中的资源管理策略、运维控制规则等；

(3) 适配：针对互联网复杂不确定情况下网络结构对业务需求的适应性问题，运用 SDN、NFV、可重构等新兴技术，进行路由调度、功能重构、资源配置、服务承载等自适应调节，增强网络的业务适应性和可扩展性，支持跨域资源的动态协同分配和深度融合利

用，使其具有柔性组织能力和持续演进能力。

4、内生性鲁棒控制机制

在不确定性威胁日益严重的情况下，为实现网元鲁棒构造、网络鲁棒控制、服务鲁棒提供，新型网络需构建“高可靠、高可信、高可用”三位一体的内生性鲁棒控制机制。

源于生物界启迪的拟态构造技术，通过在网络中引入动态异构冗余特性，采用负反馈机制应对系统中的不确定失效扰动。基于拟态的内生性鲁棒控制机制具有如下特性：

(1) 将针对目标对象执行体漏洞后门的、人为的、确定性的不确定扰动，转变为系统层面扰动效果不确定的事件；

(2) 将系统效果不确定的扰动事件变换为概率可控的可靠性问题；

(3) 基于拟态裁决的策略调度和多维动态重构负反馈机制，能够呈现出扰动发起者视角下的“测不准”效应；

(4) 借助“相对正确”公理的逻辑表达机制，可以在不依赖扰动信息或行为特征情况下感知不确定扰动；(5) 将非传统扰动因素变换或归一化为经典的可靠性和鲁棒性问题并处理之。

总结

践行互联网科技工作者的时代使命，以开放架构下 SDN/NFV 技术为基础，以基线技术的重塑为支撑，创建增量部署的演进发展模式，打造“中国智造”的新型网络，推动我国通信网络技术由“跟跑”到“领跑”，实现由“通信产业大国”到“通信技术强国”的转变是新型网络技术创新的基本逻辑。

该文从网络发展面临的主要挑战出发，提出了新型网络应具备的基本技术特征，对新型网络的核心机理进行了探讨，供广大网络科技工作者参考。d

法国力争芯片国产化替代 不被美国卡脖子

◎ 铁流

据外媒报道，法国欧洲导弹公司决定和 Soitec 公司联合收购 Dolphin Integration 研发法国自主芯片。

此次收购被认为是和前段时间美国阻止法国向埃及出售巡航导弹有关，由于阵风战斗机上的“暴风之影”巡航导弹此前使用了美国人的芯片，美国国会依照“美国国际武器运输条例”上的规定，阻止了法国的此次武器出口。

外媒报道，法国总统马克龙在今年 4 月对华盛顿进行国事访问期间，试图说服美国总统唐纳德·特朗普为巡航导弹部件提供许可。不过目前看来，美国并不愿意为他们的盟友提供巡航导弹芯片。

可能是受到了卡脖子刺激，法国人决定增强芯片自主研发能力。7 月 4 日，法国国防部长弗洛朗丝·帕利在向国民大会国防和武装力量委员会报告时候表示，法国不愿意听从美国人的摆布，要加强在武器上的独立性。

帕利表示：我们的武器正受到美国人的摆布……法国缺乏完全独立于美国的手段……欧洲依赖美国武器已经持续了 70 多年，看起来这种情况还会持续一段时间。我们都知道，事情并不会在明天就得到改善。我们要不要改变现状？答案是肯定的。

Dolphin Integration 是一家开发低功耗芯片的公司，年销售额在 1700 万欧元左右，雇佣了 155 名员工，因美国芯片公司的强势，Dolphin Integration 于今年 7 月破产。欧洲导弹公司和 Soitec 公司在完成收购后，将会为海豚整合公司注资 660 万欧元。

法国自戴高乐时代在外交上就比较独立，一直比较有想法，而且是西欧各国中，唯一还保留了比较完整军工体系的国家，因而这一次决定绕开美国自己搞一套并不让人意外。

就技术实力而言，把西欧绑在一起，确实有实力另立山头，但问题在于欧洲内部各怀鬼胎，无法形成合力，而且或多或少对美国都存在技术依赖，单凭法国一个国家无法包打整个产业链。

诚然，Soitec 公司能够解决部分原材料问题，并且在 FD-SOI 技术上有较丰厚的积累，ST 能够提供主流的制造工艺，而 Dolphin Integration 能够完成芯片设计工作，而欧洲导弹公司能够做具体应用，把芯片用在导弹上。

乍一看好像晶圆、设计、制造、应用环节都打通了。但原材料并非仅仅局限于晶圆，还有靶材、光刻胶等等，单凭几家公司无法包办。另外，像 EDA 工具、光刻机、刻蚀机、薄膜设备等全套半导体设备法国也无法自产自足。

法国现在面临的问题，和中国是类似的，都是要冲破美国的垄断，不过，由于美国在集成电路产业上的强势，以及一些欧洲国家未必会全面配合法国，可能还是会唯美国马首是瞻，因而法过想要自己搞一套，与美国分庭抗礼的可能性很低。即便西欧大合作，但很多关键技术还是要依赖美国，比如 ASML 光刻机中的光源就需要从美国进口。

法国人国产化替代的最后结局，可能只是在军工行业中，依靠他国提供的设备和工具切入芯片设计和制造领域，对美国的部分芯片实现替换。这与中国目前军工行业开始国产化替代，但民用芯片大量依赖国外技术有点类似。

还有一点启示是，就连快要变成法兰西斯坦，民族前途堪忧的法国人都要自主研发，把美国芯片换掉。以实现中华民族伟大复兴为己任的中国人，为啥还要对自主研发顾左右而言他，整天想着买国外芯片，或买国外技术授权呢？[d](#)



深度操作系统

15.8

极致体验
美观高效

深度操作系统是基于 Linux 内核，以桌面应用为主的开源 GNU/Linux 操作系统，支持笔记本、台式机和一体机。深度操作系统 (deepin) 包含深度桌面环境 (DDE) 和近 30 款深度原创应用，及数款来自开源社区的应用软件，支撑广大用户日常的学习和工作。

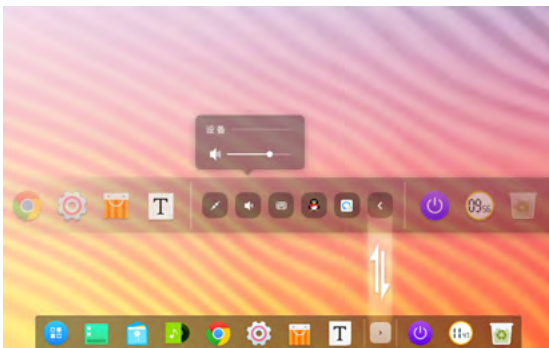
另外，通过深度商店还能够获得近千款应用软件的支持，满足您对操作系统的扩展需求。深度操作系统由专业的操作系统研发团队和深度技术社区 (www.deepin.org) 共同打造，其名称来自深度技术社区名称“deepin”一词，意思是对人生和未来深刻的追求和探索。

与上一个版本相比，深度操作系统 V15.8ISO 镜像经过优化，体积又减少了 200MB。采用全新设计的控制中心、任务栏托盘和开机引导主题，再加上性能更加优异的深度原创应用生态，希望能带给您更美观更高效的体验。

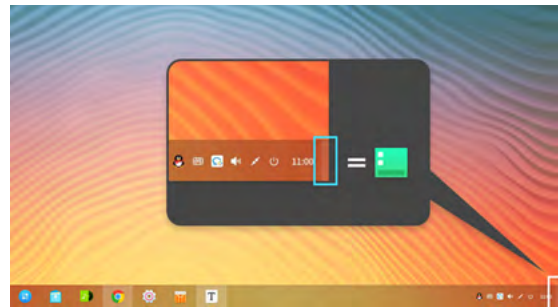
新增的功能

任务栏——选您所爱，一体双面

全新打造时尚模式的任務栏托盘布局，新增隐藏和显示控制按钮，可隐藏应用托盘列表，节省任务栏空间。同时优化托盘区域内容，将电源按钮单独突出，一方面减少点击步骤，另一方面也避免与托盘区域的功能混淆。



高效模式新增“显示桌面”区域，隐藏现有“显示桌面”绿色图标于启动器中。方便高效模式用户无障碍操作。

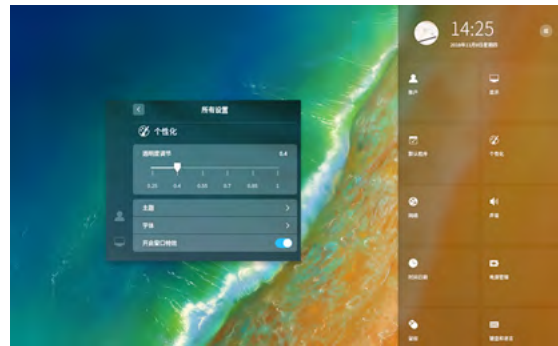


控制中心——全新布局，简约而不简单

新版控制中心精简天气插件、音乐控制功能，弱化通知板块，去除冗余开关。科学分配布局空间，灵活适配不同分辨率以及各国语言，极大提高了控制中心的使用效率。

新增“透明度调节”功能，可在控制中心中灵活调节控制中心、任务栏、启动器迷你模式的透明度，令桌面更加美观。

新增“自动调节亮度”功能（需硬件支持）。屏幕亮度随电脑所处环境光线亮度自动变化，有助于保护视力。

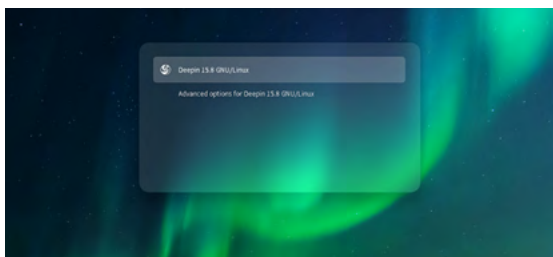


引导菜单——绚丽主题，赏心悦目

V15.8 进入系统引导界面，新增绚丽主题菜单，

从开机就能获得美好体验。

我们会在后续系统更新中推出更多主题，敬请期待。



黑色主题——低调沉稳，酷炫华丽

新增黑色图标主题，配合深色界面主题一起使用，给您不一样的感觉。



安装器——全盘加密，数据无忧

V15.8 的 ISO 镜像安装推出全新“全盘加密”功能。在系统用户密码之外再加一层防盗措施，保障硬盘数据安全无忧。



深度文件管理器——配置升级，功能强大

深度文件管理器在 V15.8 中得到进一步升级，功能将更加强大。

新增“最近使用”功能支持，可迅速找回历史文件，也可在设置菜单中隐藏。



更多细节：

- * 文件夹支持“打开方式”选项
- * 双击文件列表视图下两列之间的分割线，自动调整该列列宽
- * 完善对 Office 文档格式的识别支持
- * 增加了 KDE 和 GNOME 风格的新建文件模板支持
- * 完善了对 HiDPI 的支持
- * 修复了部分情况可能导致的文件管理器崩溃
- * 改善访问手机设备时的卡顿体验
- * 修复从属性对话框重命名 .desktop 文件（快捷方式）时文件名显示空白的问题
- * 托盘挂载插件重新支持网络位置的挂载
- * 修复部分情况预览图片可能崩溃的问题
- * 修复状态栏部分情况显示文件数量和大小不正确的问题
- * 修复回收站中复制文件可能会额外弹出提示对话框的问题
- * 修复部分情况不能使用键盘选中文件的问题

修复的缺陷与改进的功能：

V15.8 系统从后台架构、前台应用双管齐下，修复已知 BUG，优化运行效率，新增亮点功能，给您

稳定、流畅的操作体验。以下是修复、优化记录节选:

深度看图:

- * 功能精简: 只保留图片查看功能
- * 修复分辨率较大的图片在缩小后显示失真的问题

dde-session-ui:

- * 优化背景绘制
- * 优化双屏显示
- * 优化登录过程
- * 优化通知的动画
- * 修复验证密码时切换到多用户选择会显示错误
- * 修复无法登录用户
- * 修复无法设置用户的键盘布局
- * 添加网络密码验证对话框

dde-dock:

- * 修复已连接网络识别错误
- * 修复启用热点时 dock 网络 cpu 占用过高
- * 修复网络正在连接标识不能正确消失
- * 新增拖放任何 desktop 文件到 dock 上固定
- * 新增预览时识别窗口是否可以关闭
- * 新增支持改变透明度 (在控制中心中调整)
- * 新增支持新的托盘协议 (SNI)
- * 新增高效模式下显示桌面按钮
- * 新增新的时尚模式托盘
- * 热区默认关闭无预设方案, 需自定义设置

深度显卡驱动管理器:

- * 修复大黄蜂方案识别错误
- * 修复高分辨率下图片的缩放问题
- * 修改 prime 方案使用 glvnd 系列驱动
- * 优化执行错误处理

已知但未修复的缺陷:

- * 在加密分区上安装 live 系统后无法使用

- * 安装完成拔掉 U 盘后 (重启之前) 系统崩溃
- * 2D 窗管下开启屏幕缩放后, 在启动器上拖动的图标显示不全
- * 在最大化 / 取消最大化窗口时, 偶尔会导致窗口直接消失
- * 文件管理器列表模式表头, 鼠标移动到列表分割处时光标无变化 (小概率重现)

ISO 下载方式

64 位: <https://www.deepin.org/download/>

其他下载点:

百度云、Sourceforge、MEGA、Google Drive、国内外镜像源

社区内测情况总结和感谢

深度操作系统在面向广大用户发布正式版之前, 通常会在社区小范围进行一次测试, 并在正式版发布之前修复社区版反馈的问题, 同时记录社区反馈的意见。此次深度操作系统 V15.8 发布之前, 我们进行了系统内升级测试以及 ISO 安装两种形式的测试。

在此感谢所有参与测试, 提出意见和建议的广大社区用户, 深度社区有您更精彩!

关于我们

深度操作系统是一款针对普通用户发行的开源 Linux, 您可自由下载、分发、修改和使用。

欢迎您关注我们的微博、微信 (深度操作系统)、Twitter、Facebook、Github, 以第一时间获取最新动态和源代码, 同时也欢迎您前往我们的社区论坛, 与广大社区成员交流和分享您的快乐。

最后, 我们郑重感谢为深度操作系统提供测试、文档、翻译和镜像支持的社区团队与企业, 感谢您们的无私的贡献, 开源有您们更精彩。也要感谢一直支持、理解和等待我们的用户, 是您们给了深度操作系统不断前行的动力, 和不断自我修正的勇气。d



如何让 deepin 变得更加开放和透明

不 早也不晚，2018 年 11 月 15 日 deepin V15.8 如期发布。这虽然是个好消息，但是没有了延期的梗，我竟然不知道这篇研发心得应该如何开头…… 😞

开头这件事情总是很难，因为要做到一些以前没有做到的事情，就要求能将自己的水平提高哪怕那么一点点。同时，开头也不宜于高调，高调的开头太容易造成虎头蛇尾以致不能坚持的局面。在我的价值观里，与其浪费时间和精力在不能坚持的事情上，还不如什么都不做。所以很多事情我都不愿意开头，就像写这篇研发心得一样。

不过，幸运的是我们对发布系统这件事情倒是充满着热情，以至于我们能在每一次系统发布的“轮回”里，都无所畏惧地开头，并且坚持如一。比如 V15.4 的毛玻璃效果、V15.5 的高分屏、V15.6 的应用深色主题、V15.7 的性能优化等等……

当然，这次的主角 V15.8 也不输从前。

新特性介绍

V15.8 中新的变动着实不少，但是从一个研发的角度来看，我觉得最重要的变化应该是组织系统开发和发布的人发生了变化。之所以说这个比较重要，是因为 deepin 的前辈们把接力棒交到我这里，我终于可以把它顺利地转交到合适的人手中了，这句话的言外之意是什么呢？就是大家以后如果遇到什么 bug，进门左转找 @zccrs 😂。

说明了以上信息，可放心地介绍这次的新功能了。不过说实话，这次的新功能都相当直观，用不着我费什么口舌，再加上本次研发心得的重点不在此，所以大部分的新功能请看系统发布新闻，这里只说两个我最喜欢的点吧。

第一个要说的，是 V15.8 的新功能里面最让人惊喜的“Dock 时尚模式”的托盘插件的设计调整。

为什么呢？这主要是因为以我的聪明才智，都一直认为时尚模式的托盘是一个无解的问题：系统托盘占用大量空间、应用托盘使用不方便……除非引入类似 macOS 的 topbar（估计很多人也是这么想的，所以都默默地在商店里面安装了 deepin-topbar 吧）。

事实上，当时我们讨论解决方案的时候也是差不多一样的状态，每个人都欲言又止，显然是还没说出口的方案就已经被自己推翻了，然而就在这个时候，设计师默默递上了一副设计图，把所有的应用托盘也放大成了与应用图标无异的大小。

我：！？！？！？ 😞

我刚要扔臭鸡蛋，设计师咔嚓又递上了一张图，仿佛在说“蛋下留人~”然而转机就这样发生了，我看到设计图，就如大家在 V15.8 看到的 Dock 时尚模式一样，托盘问题竟然被 almost 完美地解决了——在最后的关键时刻设计师保住了自己的颜面。

怎么说呢，这基本上是我在深度继看到一个开发同事用钥匙给电脑开机感到由衷地佩服以来，再一次罕见地（尤其是被设计师？）觉得世界观被刷新的一次体验了。

另外一个要提的是一个很小的点，不过现在想起来都会有一种吃了广告中的德芙的感觉——丝滑。是什么呢？如果你现在使用 deepin 浏览这篇文章，应该能看到我特地在文章开头、上一段和中间甩锅那一段，几乎到处都贴了 emoji 表情：

·能正常显示，而不是豆腐块！

·关键是它们是彩色的！！！

整篇文章不再有黑乎乎的表情——这简直是使



用 Linux 写技术博客的 Blogger 的福音啊。如果你也想在文字编辑的时候输入开爱的 emoji 表情，可以打开 <https://getemoji.com/>，只需 copy&paste，瞬间让你拥有进入 21 世纪互联网的感觉。

至于这个新特性为什么没有写到更新笔记里面，是低调？是彩蛋？还是忘了写？咳咳，有点赶时间，继续往下写了。😄😄😄

开放和透明

上面说了发系统的接力棒已经成功交接（shuai）给别人了，那我去干啥了呢？总得做点啥吧，要不被开了就不好玩儿了。😞

我这段时间做得其实就是开放和透明。

在中国说起开源，应该没有人不知道深度作为“东半球最大的开源软件公司”的名头，如果有人不知道，那也让我们先姑且这么认为着，深度从出生的那一刻起就一直在开源——深度出品的软件产品中 95% 以上都是开源的，但是直到最近我才老有一种感觉，就是深度一直在“开源”，但是并没有在“做开源”。

深度是很擅长于做产品的，每个产品都可以做得既小巧玲珑，又戳中用户痛点，所以积累了不少海内外用户。这些用户使用着我们的系统，时不时灵感来了还会提出一些宝贵的意见和建议，这本是极好的。但是作为开发者的我发现，其实这批用户里面有不少同行，他们也想为深度的开源事业做出自己的贡献，天天问我：How to contribute.

我脑子飞速旋转：

“这个功能貌似他可以加一下？不对，这个功能需要知道很多故事背景，说来话长……还是算了”

“那个 bug 他好像可以修一下，但是好像也不能按照常规的方式去修复……一两句话也说不清楚”

……

思来想去，最后只能以

“你可以帮我们测试，测试也是一种贡献呀”

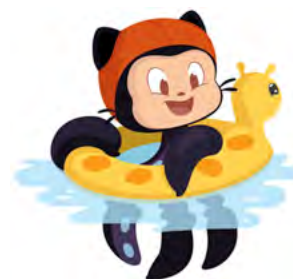
“你会德语呀，这么厉害，要不要帮我们加点翻译”

等等的回复搪塞过去，说完这些，内疚到简直想出去哭一场。

更有甚者，有些开发者克服了重重困难，终于提交了 CL（可以理解为补丁）想要一些功能，内部讨论半天给出一个决定和回复，然后对方回复，然后内部再讨论半天给一个回复……这样效率极低，还容易导致各种误解。更可惜的是，以后甚至还会有同样热心的开发者提的补丁，然后重复前面的过程，及其浪费社会生产力。

鉴于以上这些情况，我们经过长时间的思考（可能是从 V15.7 开发的过程中开始吧）和重新审视自己，决定在开发和透明上采取更多的行动，所以我们开始做了以下几个步子比较大的动作：

拥抱 Github



Inflatocat



Github 总有一种似曾相识的感觉——曾经琳姐拼了老命把 Github 上的项目都搬到了 Gerrit 上，现在我们就快要搬回来了。

拥抱 Github 主要分两个步骤：

- 第一个步骤是任务追踪和看板管理从 Tower 切换到 Github issue 和 Github project，任何人感兴趣都可以参与讨论；

- 第二个步骤是代码审核从 Gerrit 切换到 Github PR，让天下没有难提交的 CL。目前第一个步骤已经基本完成，第二步正在准备中，详细内容参见相关链接。

需要在此特别说明和广告的是：内部相关的讨论集中在 internal-discussion，所有 deepin 即将做的事情，以及要怎么做都会在这里面的 issue 中集中讨论；另外一个是很久以前便存在的 develop-center，专门供外部贡献者提交意见和 Bug 反馈。

为什么要作此区分呢？其实稍微关注一点 internal-discussion 你就会发现，deepin 每次系统发布的过程中会产生 300-400 个 issue，如果全部都放在 develop-center 中会对用户报 Bug 的体验造成不好的影响。有没有很感动呢？

推出 SDK

在 V15.8 发布之前，朋友圈曾流传一篇文章《如何为 DTK 编写文档》，如果你没有看过这篇文章，那也不要紧，因为这篇文章我最想突出的一句话就是：

“是时候给 DTK 添加文档了！”

当然，给 DTK 添加文档这只是第一步，推出 SDK 才是关键。

何为 SDK？在我看来其实就是写代码、调试、打包和发布一条龙服务，至少包含开发库、开发文档、工具和 IDE 等项。

深度又要造轮子了么？可能要让大家失望了，答案是否定的。以我们对 Qt 的喜爱，怎么可能抛弃 QtCreator 呢……至少现阶段我们还不会抛弃 QtCreator 去造轮子的。但是也不用那么失望，等我们完善了文档、改进完 QtCreator，让 DTK 跟 QtCreator 无缝集成，你甚至都不用会写代码就能编译、运行、甚至打包一个“Hello world!”应用来！

持续的技术输出

深度像大部分公司一样，有老鸟也有新手，但是与绝大部分公司不一样的是，深度的高手感觉牛到可以戳上天，有那种对谷歌的 offer 都要装逼回绝掉的那个类型、也有手撸窗口管理器的同时还能顺带开发一个深度影院的、还有精通所有程序语言和内核开发的……在这样的公司里混开发，生存真是倍感压力。

所以，我一直有一个梦想，就是每天上班不用顶着压力慌忙地敲代码，而是可以跟这些大牛一起摸着键盘谈笑风生。终于……我最近过上了这种生活……才怪，人总要意识到，梦想还是要有的，但是真不见得会实现……唉。

docs.deepin.io

当然，上面只是玩笑话了，真实的情况是我长期蒙受各位大牛的指导，心里倍感感激的同时，想着其实社区还有那么多搞技术的小伙伴，学校还有那么些想来深度实习而不得的实习生，他们其实对深度开发者的经验和技能不能说瞻仰吧，有点夸大





其词了，还是非常好奇和希望能学到的。所以，从今年3月份开始，我们内部一直在推进 <https://docs.deepin.io> 这个技术博客的建设，除了希望能随着 deepin 的成长有所技术积累外，也希望把这些技术积累分享给社区，大家共同成长和进步。

持续的技术输出没有什么弯弯绕，就是这么直接的东西。

结语

说了这么多，其实我的想法挺简单，就是通过逐步开放我们内部的一些软硬件资源，希望能让更多的开发者能更加方便地参与到我们的平台建设中来，而不必一个人摸瞎；通过刻意透明所有的讨论过程，包括需求讨论、方案设计、设计稿等等，让所有社区的小伙伴不仅可以看到我们在做什么，甚至可以参与讨论，“干涉”我们要做什么，让所有对操作系统设计有热情的用户的意见都得到公正对待。

最后，希望开放和透明，可以给深度带来一点不一样的色彩。d

深度·讲坛 征稿啦

是什么让你豁然开朗？是什么让你灵感迸发？又是什么让你百思不解？

这里汇聚了深度大神，专门凑在这里搞技术，从今往后，这里专治各种疑难杂症，你可以来提问，可以来传授技术，总之，这里都是爱凑热闹的技术控。

分享专业知识，方便学习交流，内刊“深度·讲坛”栏目长期向各位技术控们征集技术稿件啦。

字数：1000+，图文并茂

内容：必须为原创

投稿邮箱：deepin-magazine@deepin.com

注明“部门+姓名”





深度 Linux 已足够好，尤其对于一般办公者和开发者

danlong / 2018-9-3 14:08

过去办公室的办公电脑全被我给换成了 deepin 系统，之后开了个饭店，把美团外卖的主机也换成了 deepin 系统，成功安装了美团外卖商家版。已经习惯了用这款系统，感觉比 win 要好用太多太多，我还会一直支持下去，也期待 deepin 能尽快壮大。

yuan0306 发表于 2018-9-3 16:58:07

赞，日常主力快一年了，不过我是程序员，这不算什么，你就很厉害了。



用了这么多年的微软，感谢 deepin，deepin 系统工作使用一周有感

mojobs / 2018-10-18 17:09

deepin 系统工作使用一周有感。单位配置的电脑，是差劲的 dell 小型台式，那个性能，装个 Windows10，能用，就是慢半拍的感觉很难受。突然想用用 Linux 系统，虽然家里的 dell7537 也是深度系统，但是单位的不是啊。有着大学的计算机专业的底子，1 个小时配置好系统。直接 deepin 单系统，我可不要什么双系统。

使用一个礼拜来说，舒服，我还以为我工作就不再使用 linux 系统了，我的记忆还在原来 Ubuntu7、debian5 的时候，这些年下来我也不知道这些系统进化到什么地方了，不过应该也很好吧，可那丑丑的界面我还是接受不了，别说安装什么主题，得配置很久。今天用虚拟就跑了一下 debian9.5，那个配置安装的时候的煎熬，一如既往。

我就一工作的，不需要使用那么多的软件，所以在我看来，deepin15.7 就可以了，简单使用来工作。关于打印机这一块，别纠结什么打印机驱动了，直接一个深度云打印就搞定，这是最简单的。服务器装一个客户端就行。

给深度官方提个小意见，手动安装的软件，哪怕修改了源，手动升级没什么，可是刚刚系统来个补丁升级，又把我安装的软件给全部降级回去，呵呵呵，这个 ... 还得重新安装一次，虽然简单，可是能不能给我一个提示，系统补丁对什么软件有影响，我也好手动屏蔽啊。

在此，也感谢国产软件的跟进，有你们的存在，让我使用更方便。从 msoffic 切换到 wps 是没问题的，换成 libreoffice 还是 openoffice，那个界面我真心不适应。

successfully 发表于 2018-10-18 20:04:10

15.8 你会更爱！



感谢深度技术猿

nukes_lintao / 2018-11-19 22:49

深度更新到 15.8，感到热血沸腾，第一时间下载安装。

今天又看到与华为的 15.2 服务器版新闻，祝愿深度越来越好！

深度的有 linux 信仰的大批技术猿、工程师为此付出了艰辛的努力，使中国的 linux 环境越来越好，感谢你们！！！！

jkjk612 发表于 2018-11-20 12:28:08

deepin 真的把 linux 做得很傻瓜化了，希望国家提倡支持 linux 桌面使用，省得 microsoft 一家独大。



15.8 确实不错，优化很好，速度流畅！

xy_god / 2018-11-17 16:16

在我的最低配置的笔记本上，居然开全特效，完全流畅！用起来，感觉并不比 lubuntu 差多少。

wcswcs4 发表于 2018-11-17 19:31:32

流畅了不少。

syitian 发表于 2018-11-17 22:21:36

用了 3 天，确实进步很大，没遇到什么大 BUG。



普通用户使用 deepin 比其他版本 linux 方便！！！！

sd117 / 2018-11-26 08:16

我是普通用户也是喜欢折腾的人，很多版本的 Linux 都安装过用过，但很多版本就安装软件也是把你折腾个半死，不太适合普通人用户使用，但现在装了 deepin 之后觉得用起来还是可以，但如果非要和 Win 比的话还是有很大差距，这个差距，大家懂的，如果有 deepin 的工程师或者管理看到这贴，希望他们以后发展要在普通用户角度方向发展，这才是占领市场的根本毕竟普通用户比那些 N B 的人多，最后说的是软件安装要容易，最好做个什么打包软件的不要输入一大堆命令等，希望我们国产系统越来越好！以打败微软为目标，前进！！

wcswcs4 发表于 2018-11-26 08:30

支持 deepin，用自已作的系统，扬国人的志气！

jyxfld 发表于 2018-11-26 15:03

这几天刚接触 deepin 系统，感觉确实不错，正在逐渐熟悉适应之中。



deepin 很好用

jjewi / 2018-11-27 16:15

以前装过 Ubuntu。从搜狗下载了一个 deb 想安装，双击却是把文件展开了。就像打开了一个压缩包。那时候想真**Linux 就这鸟样怎么和 Windows 比。苹果改变世界不就是因为把一件简单的事做到极致，甚至融入了艺术思想。从纸带编程 > 到汇编 > 到面向对象，那不是进步？码农的编码进步带动用户体验进步那不是生产力的进步？我觉得 deepin 做的有意义，很好用。

希望 deepin 以后更新要简洁简单实用。因为现在用户都被手机上的简单操作惯了。

wcswcs4 发表于 2018-11-27 19:11

使用者追求的是相对体验 .deepin 作的不错，这是可以肯定的 . 但反映体验中的自我感觉，其质并无恶意！



浅谈操作系统在农信系统 实践国产化的意义

◎ 吴江波 周新超 李文昌 / 文

为了加强软件版权保护、鼓励软件创新，近年来国务院、国家版权局及国家相关部委一直不遗余力地推动操作系统的正版化（国产化）工作。2018年，河南省省联社进一步强调尊重软件知识产权、加强金融信息安全治理，要求年底前各市县行社必须完成软件正版化达标任务。

灵宝市农村信用合作联社严格按照省联社要求，在经过大量市场调查、严格内部评估、可靠使用测试的前提下，创新性地引入国产深度操作系统、国产 WPS 办公软件，既全面完成了省联社正版化任务，又探索了一条操作系统的国产化之

路，为全省农信系统操作系统国产化积累了一定的经验。

在国内桌面办公系统领域，一直被国外操作系统和办公软件产品垄断。但是近年来频繁爆出的“棱镜门”、“永恒之蓝勒索病毒”、“中兴被制裁”等事件，充分反映出过度依赖国外产品极易导致敏感信息和涉密数据被窃取、系统安全频受威胁和攻击、核心技术被人“卡脖子”等重大安全事故。

国内信息行业人士一致认为，在关键领域和关键业务范围进行国产化，以保证信息化设施的在自



主可控层面实现基础安全，已经成为了信息化的重要保障趋势。尤其是对我们金融系统来说，使用非国产软件可能对金融信息安全产生巨大隐患，所以说软件国产化是势在必行的。与此同时，国产深度操作系统等软件经过多年的研发和锤炼，已经达到国际主流水平，且在党、政、军等国家核心机构得到大量应用，已经从“可用”发展到“好用”。

我社在进行基础软件国产化方面做了大量的探索实践工作。

一是集体探讨、达成共识。联社党委高度重视使用国产深度操作系统的议题，组织专人进行专题讨论，对操作系统国产化的意义、使用国产操作系统的可行性、信息安全自主可控的重要性进行共同探讨，形成一致意见。

二是多维度考察、对比。对市场上主流的国产操作系统、办公软件进行自主版权、操作界面、用户数量、使用体验等多维度进行考察，确保能够胜任我社办公需求。特别需要指出的是，深度操作系统作为一款国内领先的自主创新操作系统，在流畅简洁的操作界面，高度集中的软件管理和友好的使用界面方面表现突出，给我社使用人员留下了深刻的映像。

三是真机安装测试。使用专用测试电脑安装国产操作系统和办公软件，对常用的文档、表格、浏览器、国产操作系统稳定性、操作简便性等指标的可靠性进行了持续性的测试，确保替换后不影响工作效率。

四是全员宣讲动员。普及操作系统国产化知识，宣传金融信息安全的重要性，宣讲国产操作系统发展现状和替换的可行性。

五是安装调试。取得深度科技厂商的正版授权后，对辖内全部近 150 台办公电脑逐台更换安装国产深度操作系统，并现场调试常用操作、连接打印机等，确保替换后员工体验更佳。

六是集中操作培训。对办公应用中的通用操作，集中进行演示培训，确保员工使用没有障碍。

七是总结反馈优化。收集员工使用过程中的问题和建议，及时反馈给深度科技解决，定期总结操作系统国产化的应用经验。这里也对深度科技对我社技术服务的大力支持表示感谢。

经过上述实践，我们总结了采用国产深度操作系统和国产办公软件的以下优势。

一是节省了成本投入。国产深度操作系统、国产办公软件均采用整体“场地授权”的方式进行正版化授权，总授权价格与微软相比，节省了 2/3。

二是国产深度操作系统在病毒防护、广告拦截、垃圾软件防护等方面具有天然优势，有利于打造纯净的网络办公环境。

三是信息安全自主可控，国产深度操作系统减少了国外软件“后门”隐患，有助于防范敏感信息、特别是金融信息泄密。

四是国产深度操作系统优化程度高，应用软件丰富，界面友好操作简便，对硬件兼容性好，员工接受程度高，推广使用门槛低。

我社在全面推广过程中也遇到的一些问题。一是个别特殊岗位使用的个别专业化软件还未适配。二是省联社核心系统客户端、需要安装插件的业务系统等暂时不能兼容。这些问题是下一阶段重点推进解决的。

我社下一步的工作规划。

一是向省联社提出基于国产深度操作系统的技术开发需求，使核心业务系统客户端及插件能支持国产深度操作系统，以期实现国产基础软件的全面替代。

二是我社拟在实现操作系统国产化的基础上，进一步探索国产服务器、国产龙芯终端替代的可行性，实现操作系统、应用软件、服务器、终端在我社的全面国产化应用。d



anything is possible

● 深度科技 北京公司 首席技术官 张磊 / 文

简介

什么是 anything 呢？按照项目文档简介来说，它是 Something like everything, but nothing is really like anything...

所以，anything 名字的由来是受到了 Windows 下 everything 这个软件的启发。在文件管理器里，针对文件名进行搜索一个很常见的需求。但是 GNU findutils 里的 find 搜文件太慢，locate 搜文件又会有很大的延迟，因此，anything 目标就是为 Linux 用户提供一个快速的文件名搜索的功能。如果一定要说，其实在 Linux 中也有一个类似的叫 rlocate 的程序，但是那个程序有点太老了，而重写一个更好的也不难，因此我们就重新造了一个轮子。

为了说明 anything 的特性，先来做一个简单但仍有典型意义的文件搜索对比测试。

首先说明下测试环境，测试环境物理机为 ThinkPad x230，8G 内存，机械硬盘。虚拟机为运行在 VirtualBox 里的 Deepin 15.4，内存为 2G，处理器为单核 2.9 GHz。在这里，使用虚拟机的主要原因是为了测试方便，因为 anything 涉及到内核开发，大家知道，内核编程很容易把系统挂掉的，使用虚拟机乃是内核开发的居家必备法宝之一。

虚拟机的文件系统排除 /sys、/proc、/dev 与 /run 目录后共有目录、链接与文件数约 38.7 万个，其中挂载了一个虚拟机外的文件系统以方便文件交换，在虚拟机内此文件系统类型为 vboxsf，大约有 11.5 万个文件与目录。

测试方法如下：

- 使用 `sysctl -w vm.drop_caches=3` 清空缓存，

然后运行 `find / -name "*hellfire*`，耗时约 39.9 秒；

- 带缓存再次运行 `find / -name "*hellfire*`，耗时约 10.1 秒，通过运行 `free` 命令对比 `cache` 项得知新增的文件缓存约占用 260MB 内存；

- 使用 anything 的基础索引搜索 hellfire，耗时约 6 毫秒，基础索引占用内存约 7MB，测试程序占用内存约 14MB；

- 使用 anything 的二级索引搜索 hellfire，耗时 0 毫秒，二级索引占用内存约 230MB，测试程序占用内存约 500MB。

经多次测试表明，使用基础索引比使用带缓存的 find 搜索要快大约 500 倍甚至更多。若使用全内存的二级索引，anything 的搜索速度是使用基础索引搜索速度的 20 倍以上，但是占用内存将增长 35 倍。若将二级索引存放在磁盘上，其内存占用将减少到近乎零，在大部分情况下仍然能够取得很好的搜索速度，仍比基础索引有明显的速度加快，但是在原始字符串较多的情况下，因为需要从索引文件中读取大量的数据，就会显得比基础索引搜索还要慢了，这个是典型的时间空间复杂度互换的问题。

下面对 anything 设计与实现进行一个简单的分析。

基础索引设计

简要分析

在 Linux 下，最常用的文件搜索软件是 find，它会递归遍历每个目录，针对每个目录与文件按照用户给出的参数过滤出符合条件的目录与文件并打印出来。

在命令行模式下，find 使用很广，功能强大，不仅可以根据文件名查找文件，还可以根据文件类型、权限、修改时间、大小，甚至与其它软件配合对文件进行过滤。但是在图形界面下，用户最常用的仍是根据文件名对文件进行查找，这正好是 anything 的用武之地。

使用 find 搜索时，如果已经搜索过一遍，则对应的文件与目录信息将会被缓存在内存中，可以大大加速搜索。当然，root 用户可以运行 `sysctl -w vm.drop_caches=3` 来清除这些缓存。但是，即使使用了缓存，在仅使用文件名搜索的前提下，find 的搜索速度仍比 anything 至少慢两个数量级。

anything 之所以这么快主要是因为：

- 相比于 find，它不依赖于额外的系统调用与函数调用，减少了大量的系统调用开销与内存复制开销；
- 它使用的文件系统存储结构的设计针对性极强，内容非常紧凑，使用也很高效。

find 会通过系统调用遍历每个目录的内容，读取其中的文件列表，再根据文件列表中的 inode 号找到对应的 inode 信息，接着读取 inode 类型为目录中的文件内容……如此递归查找。可以看到这个查找过程经过多重递归，会不断打开目录（需要系统调用与内存复制），而且文件名在其中与 inode 信息夹杂在一起，会严重减缓纯粹基于文件名的查找。

anything 的不同之处在于在其内部仅保存了文件（目录）名以及必要的归属关系，以便进行双向的查找。所有的数据都存放在用户态空间，没有额外的系统调用开销与内存复制开销。此外，考虑到数据访问的最大可能性，相邻数据应是最有可能被一起访问的数据，因此可以最大限度利用处理器的高速缓存，使得软件的运行性能更好。

内部线性存储

在经典设计中，文件系统应该使用数据结构中的树来存储。树中的每个节点有一个字符串作为名称，有 N 个指针指向其 N 个子节点，还有一个指针指向其父节点以支持从上至下以及从下至上的双向树遍历。这种结构的优点是修改很方便，不管是删除、添加还是重命名一个节点都很快。但是在遍历读取的时候却因为需要不停的指针跳转，会导致处理器的高速缓存频频失效，从而使得访问速度降低，而又因为各个节点都是分散的，无法体现出相邻节点的访问相关性，所以也难以进行有效的内存访问优化。

下面是上述树设计数据结构内存消耗的分析。简单起见，先假设文件系统是一棵深度为 D 、分叉为 N ($N > 1$) 的完全树，这样，在整棵树中将一共有叶节点 N^{D-1} 个，非叶节点 $N^0 + N^1 + N^2 + \dots + N^{D-2} = (N^{D-1} - 1)/(N - 1)$ 个。对于每个叶节点而言，忽略其文件名，其它部分的内存仅包括一个指向父节点的指针。对于每个非叶节点而言，忽略其文件名，其它部分的内存包含一个指向父节点的指针（假设根节点也有一个父节点指针，但是其值为空），以及 N 个指向子节点的指针，一共的指针数是 $P = N^{D-1} + (N+1) \times (N^{D-1} - 1)/(N - 1) = (2 \times N^D - N - 1)/(N-1)$ 个，对于 64 位系统而言，需要的内存是 $8 \times P$ 个字节。

在 anything 内部存储结构设计的早期阶段，使用树来保存文件系统结构也曾被考虑过，但是顾及高速缓存失效与指针过大的问题，显然在此处使用线性存储设计更好，假设某目录下有 $d1$ 与 $d2$ 两个文件夹， $d1$ 下有两个文件 $f1$ 与 $f2$ ，还有一个空的子目录 $d3$ 。而 $d2$ 下有一个文件 $f3$ 。如图 1 所示。

对应的 anything 内部的线性存储如图 2 所示。

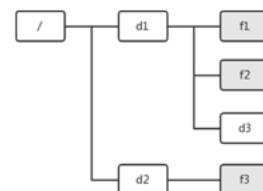


图 1 示例目录树

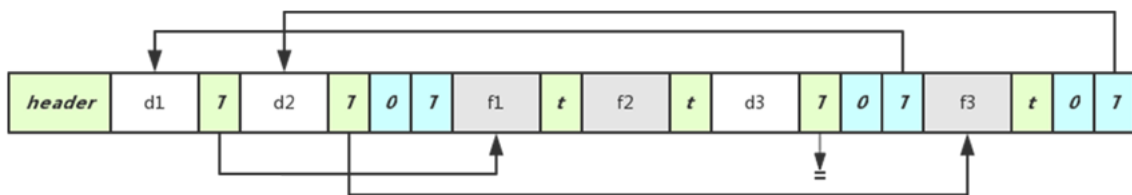


图 2 anything 目录树线性存储结构

从上述例子中可以看到，在 d1 的标签 (T) 中保存了一个偏移量，指向的是以 f1 开始的位置表示文件夹内容的开始，而在 d3 结束后，其标签也保存了一个偏移量，指向 d1，表示父目录的位置。对于空目录，例如 d3，其标签保存的偏移量为零。

由上述结构可以看出，相邻的节点，即同级目录下的文件，都存储在一起，这样有利于内存访问优化。而且将 8 字节指针优化为 4 字节偏移量，也节省了存储空间，减少了无效数据的存储与无效内存的访问，此外，线性存储也便于数据文件的存盘与读盘。

对于每个叶节点，需要耗费 1 个字节存储标签，对于每个非叶节点，需要消耗 4 个字节存储子节点的偏移量，还有 4 个字节用来存储其所有子节点指向其本身的偏移量，则一共需要 $N^{D-1} + 8 \times (N^{D-1} - 1) / (N-1) = (N^D + 7 \times N^{D-1} - 8) / (N-1)$ 个字节，对比之前树需要耗费的 $8 \times P = 8 \times (2 \times N^D - N - 1) / (N-1)$ 个字节，易知在最好情况下是其内存的 1/16，在最差情况下是其内存的 1/3。

当使用此结构进行搜索时，不需要再通过树状搜索，而可以使用线性搜索，从前至后进行字符串匹配，并跳过数据量较小的标签即可，这样也方便进行搜索分页。

当搜索到某个名称符合要求时，即可使用目录结尾的标签定位到父目录的偏移量，继而构成完整的路径，返回给调用者。

在这里也可以发现，其实对于搜索而言，程序

并不需要保存子节点的偏移量，只需要父节点的偏移量即可，这样还可以把额外的内存再节省约一半，但是这会导致文件系统更改（文件与目录删除、添加、重命名）时变更速度较慢。若仅需使用离线搜索，即不考虑文件系统改动的问题，则内存消耗确实还可通过使用上述方法进一步减少。

此外，由于需要至少一位来标识文件或者目录，因此最大的偏移量不可能是 2^{32} ，即最多能存储或 2G 内存的数据，为了可扩展性考虑，在内部其实保留了两为数据以进行文件标识，因此，最多可以使用 1G 内存作为内部存储。根据之前的测试估算，大约能保存 4000 万个文件或目录，在目前看来是足够的。

理论对比与实际测试

下面首先做理论对比，继而进行实际测试验证。测试环境仍包含 38.7 万个文件（目录），其中有大约 4 万个目录。

若使用经典的目录遍历方法，即使用 opendir/readdir/closedir 函数遍历目录，则需要调用大约 4 万次 opendir 与 closedir，38.7 万次 readdir，这些都是系统调用，而其中 readdir 将读取一个 struct dirent 大小的数据，即大约 274 个字节的数据。即使这些数据都在缓存中，也只是节省了从磁盘到内存的数据读取，但是仍然无法节省从内核态到用户态的数据复制开销。再接着，需要对这 38.7 万个数据进行 38.7 万次 strstr 调用。这就是 find 的大致开销，其中完全忽略了磁盘访问。

若使用树进行数据存储与搜索，在遍历过程中需要持续生成节点，在搜索过程中需要递归遍历（递归函数调用）各个节点，然后对每个节点调用 strstr 函数，即需要有 4 万次递归函数调用，以及 38.7 万次 strstr 函数调用。当然，由于数据存储在内存中不连续，因此高速缓存失效较多。在搜索过程中，由于不再需要系统调用，以及额外的从内核态到用户态的内存复制（这个开销相对较小），这种搜索方法应该会至少比 find 快一个数量级。

使用 anything 也首先需要进行目录树遍历建立基础索引。一旦基础索引建立完毕后，在搜索过程中，只需要从前向后，反复在线性存储空间中调用 strstr 即可，即大约需要 38.7 万次 strstr 调用。没有系统调用开销，没有内存复制开销，而且由于内存紧凑，又是单向内存顺序访问，因此对高速缓存的利用率更高，空间与时间效率比目录树状递归查找都要更好。此外，使用线性结构的基础索引也方便数据的保存与读取，但是其偏移量的管理会更复杂一些。

为了进一步确定上述猜测是否正确，下面可以开发三个程序，第一个程序模仿 find 的做法，通过 opendir/readdir/closedir 函数遍历目录，同时调用 strstr 进行搜索。第二个程序遍历目录的时候建立树形结构后，再基于树形结构进行搜索。第三个程序遍历目录的时候构建基础索引完成后，再基于基础索引进行搜索。

程序一：递归搜索（类 find）

```
void walkdir(const char* path, const char* query, int* count)
{
    if (is_special_mount_point(path))
        return;

    DIR* dir = opendir(path);
    if (0 == dir)
        return;

    struct dirent* de = 0;
    while ((de = readdir(dir)) != 0) {
        if (strcmp(de->d_name, ".") == 0 || strcmp(de->d_name, "..")
            == 0)
            continue;

        // DT_REG: regular file/hardlinks, DT_LNK: softlinks
        if (de->d_type != DT_DIR && de->d_type != DT_REG && de->d_type != DT_LNK)
            continue;

        if (strstr(de->d_name, query) != 0) {
            if (*count < 10) {
                printf("%d: %s/%s\n", 1 + *count, path, de->d_name);
            } else {
                printf(".");
                if ((*count) % 10 == 0)
                    fflush(stdout);
            }
            *count = *count + 1;
        }

        if (de->d_type == DT_DIR) {
            char fullpath[PATH_MAX];
            sprintf(fullpath, "%s/%s", path, de->d_name);
            walkdir(fullpath, query, count);
        }
    }

    closedir(dir);
}
```

```
continue;

// DT_REG: regular file/hardlinks, DT_LNK: softlinks
if (de->d_type != DT_DIR && de->d_type != DT_REG && de->d_type != DT_LNK)
    continue;

if (strstr(de->d_name, query) != 0) {
    if (*count < 10) {
        printf("%d: %s/%s\n", 1 + *count, path, de->d_name);
    } else {
        printf(".");
        if ((*count) % 10 == 0)
            fflush(stdout);
    }
    *count = *count + 1;
}

if (de->d_type == DT_DIR) {
    char fullpath[PATH_MAX];
    sprintf(fullpath, "%s/%s", path, de->d_name);
    walkdir(fullpath, query, count);
}
}

closedir(dir);
}
```

程序二：递归建立树结构后搜索

```
typedef struct __fs_tree_item__ {
    char* name;
    int kids_count;
    struct __fs_tree_item__* parent;
    struct __fs_tree_item__** kids;
} fs_tree_item;

void walkdir(const char* path, fs_tree_item* fti)
{
    if (is_special_mount_point(path))
        return;

    DIR* dir = opendir(path);
    if (0 == dir)
        return;

    struct dirent* de = 0;
    while ((de = readdir(dir)) != 0) {
        if (strcmp(de->d_name, ".") == 0 || strcmp(de->d_name, "..")
            == 0)
            continue;

        // DT_REG: regular file/hardlinks, DT_LNK: softlinks
        if (de->d_type != DT_DIR && de->d_type != DT_REG && de->d_type != DT_LNK)
            continue;

        fs_tree_item* kid = calloc(1, sizeof(fs_tree_item));
        if (kid == 0) {
            printf("kid malloc error, path: %s, name: %s, count: %d\n", path, de->d_name, *count);
            continue;
        }
        kid->parent = fti;
        kid->name = strdup(de->d_name);
        if (!kid->name) {
            printf("strdup error, path: %s, name: %s, count: %d\n", path, de->d_name, *count);
            free(kid);
            continue;
        }
        fti->kids[fti->kids_count] = kid;
        fti->kids_count++;
        walkdir(kid->name, kid);
    }

    closedir(dir);
}
```



```

%d\n", path, de->d_name, fti->kids_count);
    continue;
}

kid->name = strdup(de->d_name);
if (kid->name == 0) {
    free(kid);
    printf("kid strdup error, path: %s, name: %s, count:
%d\n", path, de->d_name, fti->kids_count);
    continue;
}
kid->parent = fti;

fs_tree_item** kids = realloc(fti->kids, (fti->kids_count+1) *
sizeof(void*));
if (kids == 0) {
    free(kid->name);
    free(kid);
    printf("kids realloc error, path: %s, name: %s, count:
%d\n", path, de->d_name, fti->kids_count);
    continue;
}

fti->kids = kids;
fti->kids[fti->kids_count] = kid;
fti->kids_count++;

if (de->d_type == DT_DIR) {
    char fullpath[PATH_MAX];
    sprintf(fullpath, "%s/%s", path, de->d_name);
    walkdir(fullpath, kid);
}
}

closedir(dir);
}

char fullpath[4096];

void print_path(int count, fs_tree_item* fti)
{
    int cur = sizeof(fullpath) - 1;
    while (fti) {
        int l = strlen(fti->name);
        cur -= l;
        memcpy(fullpath + cur, fti->name, l);
        cur--;
        fullpath[cur] = '/';
        fti = fti->parent;
    }
    printf("%d: %s\n", count+1, fullpath+cur);
}

void search_tree(fs_tree_item* fti, const char* query, int*
count)
{
    if (strstr(fti->name, query) != 0) {
        if (*count < 10)
            print_path(*count, fti);
        *count = *count + 1;
    }
}

```

```

for (int i = 0; i < fti->kids_count; i++)
    search_tree(fti->kids[i], query, count);
}

```

程序三：建立基础索引后对其进行搜索

```

static int match_str(const char* name, const void* arg)
{
    return strstr(name, (const char*)arg) != 0;
}

static uint32_t search_by_fsbuf(fs_buf* fsbuf, const char*
query)
{
    uint32_t name_offs[MAX_RESULTS], end_off = get_tail(fsbuf);
    uint32_t count = MAX_RESULTS, start_off = first_name(fsbuf);
    search_files(fsbuf, &start_off, end_off, query, match_str,
name_offs, &count);

    char path[PATH_MAX];
    for (uint32_t i = 0; i < count; i++) {
        char *p = get_path_by_name_off(fsbuf, name_offs[i], path,
sizeof(path));
        printf("%'u: %c %s\n", i+1, is_file(fsbuf, name_offs[i]) ? 'F' :
'D', p);
    }
    uint32_t total = count;
    while(count == MAX_RESULTS) {
        search_files(fsbuf, &start_off, end_off, query, match_str,
name_offs, &count);
        total += count;
    }
    return total;
}

```

在程序里单独在搜索前后调用 `gettimeofday` 获取时间戳，进行差分比较，在有缓存的情况下，三种方法搜索的耗时分别为：

- 边递归遍历目录边匹配（类 `find`）：10.1 秒；
- 递归遍历目录后用树存储目录结构再搜索：11 毫秒；
- 递归遍历目录后用基础索引存储目录结构再搜索：8 毫秒。

图 3 是程序一的 `google-perftools` 性能剖析图。

如果使用 `perf` 与 `google perftools` 进行性能剖析，可以看到在第一个程序里主要的程序耗时（60%）都花在了 `readdir` 函数上，17% 的时间花

```
./bin/debug/search_directly
```

```
Total samples: 181
```

```
Focusing on: 181
```

```
Dropped nodes with <= 0 abs(samples)
```

```
Dropped edges with <= 0 samples
```

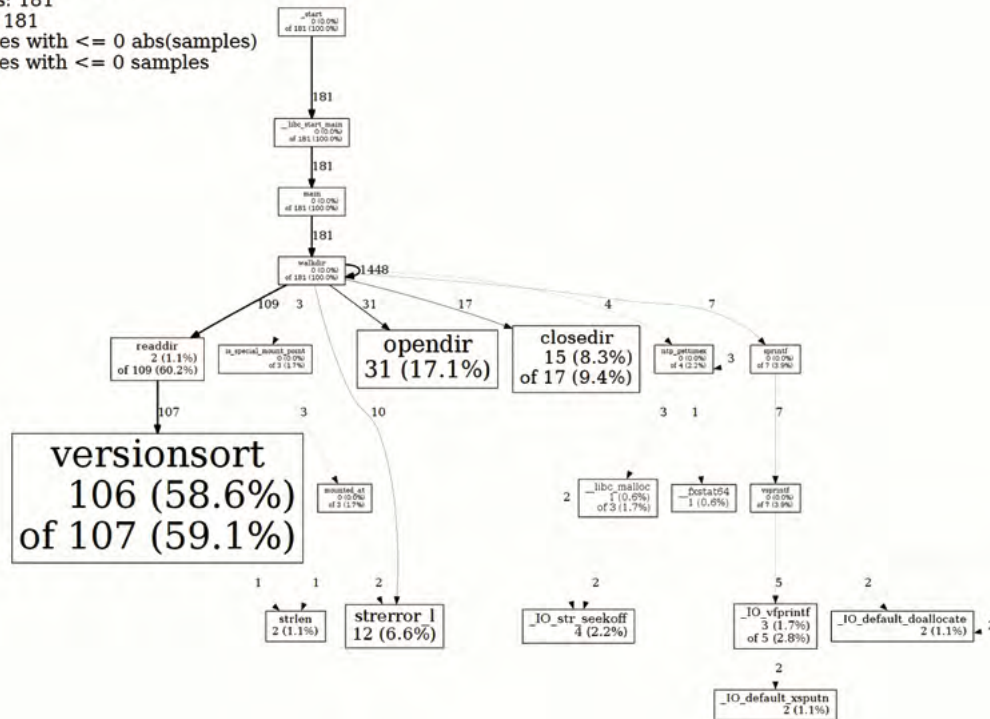


图3 边遍历目录边搜索的程序性能剖析图

在了 `opendir` 函数上。第二三个程序的主要耗时则是 `strstr` 函数与 `strlen` 函数，前者主要是为了进行字符串匹配，后者主要是为了遍历基础索引的线性存储空间中的文件名字符串，当然，`strlen` 函数的调用仅出现第三个程序中，因此，相对于树形结构的程序来说第三个程序也有自己额外的性能开销。

二级索引

上述基础索引的设计已经大大提高了基于文件名的搜索速度（相对于 `find` 而言），但这种搜索方法仍然需要从前至后遍历每个文件名进行搜索（即如性能剖析所示，调用 `strstr` 函数的开销）。从表面上看，因为需要对每个文件名进行检查，以得知其是否匹配搜索串，貌似很难更快了。但事实上，我

们还可以做得更好，那就是使用倒排索引（inverted index）。

倒排索引的原理是对原字符串进行切割，得到其所有的子字符串，再将这些子字符串存放对应的索引数据结构中，当用户输入对应的子字符串时能立刻找到相应的原始字符串。

例如原始字符串为 `china`，则可以得到 `c`、`h`、`i`、`n`、`a`、`ch`、`hi`、`in`、`na`、`chi`、`hin`、`ina`、`chin`、`hina` 与 `china` 一共 $5+4+3+2+1=15$ 个子字符串，假设 `china` 这个字符串的偏移量（或者指针）是 100，则可以得到 `{“c” → 100}`、`{“h” → 100}`、`……`、`{“china” → 100}` 等 25 个索引。当用户输入 `hi` 时，即可以立刻得到 `{“hi” → 100}` 这个索引，然后将 100 返回给调用者，由调用者通过这个偏移量或者指针得



```

./bin/debug/search_by_fsbuf
Total samples: 4
Focusing on: 4
Dropped nodes with <= 0 abs(samples)
Dropped edges with <= 0 samples

```

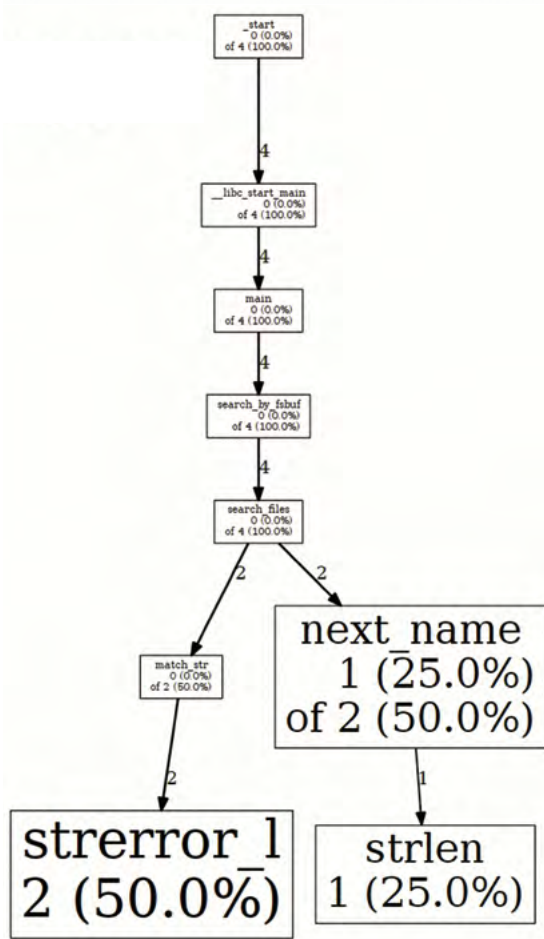


图 4 程序三的 google-perftools 性能剖析图

到” china” 这个字符串。

在简单的倒排索引设计中，一般使用哈希链表或者 Trie 树作为索引数据结构。anything 使用的是哈希链表。其工作原理是，当用户输入某个查询字符串（如 hi）时，anything 将首先对字符串进行一

次哈希运算，得到哈希数组中的下标值，然后根据下标值即可得到对应的索引数组，在索引数组里顺序查找即可得到对应的索引（即 hi），从而可以获取到所有的包含 hi 字符串的原始字符串的偏移量，将其返回给调用者。

关于倒排索引、哈希链表和 Trie 树的资料，网上和相关的书籍已经非常丰富了，在这里不再赘述。

下面是对二级索引内存消耗的分析，以及哈希链表设计问题的讨论。

对于长度为 L 的串，其生成的索引中，长度为 1 的子串有 L 个，长度为 2 的子串有 L-1 个，……，长度为 L 的子串有 1 个，则一共有 L + (L-1) + …… + 1 = L x (L+1) / 2 个子串，消耗的存储空间为 1xL + 2x(L-1) + 3x(L-2) + … + Lx1 = Lx(L+1)x(L+2)/6，若考虑到字符串指针（在 64 位系统上为 8 字节）与字符串结尾的 0 字符，则额外需要 Lx(L+1)/2x(8+1) 个字节，合计为 Lx(L+1)x(L+29)/6 个字节。当然，其中字符串是可以被重用的，但是不一定能被重用，因为越长的串重复可能性越低。而且另一方面来说，这里的串是包括了中文字符的串，转成 utf8 后索引还会更长，此处的分析也忽略了这一点。下面是几个典型的数据：

- 串长 5 时，占用 170 字节
- 串长 10 时，占用 715 字节
- 串长 20 时，占用 3,430 字节
- 串长 30 时，占用 9,145 字节
- 串长 40 时，占用 18,860 字节
- 串长 50 时，占用 33,575 字节
- 串长 60 时，占用 54,290 字节
- 串长 70 时，占用 82,005 字节
- 串长 80 时，占用 117,720 字节

可以发现一个串长 80 的文件名产生的空间占用就等效于 164 个串长为 10 的文件名，或者 692 个串长为 5 的文件名，虽然 80 仅是 10 的 8 倍，以及 5 的 16 倍。此外，真实的文件名还是有可能挺长的，

比如 .git 目录下的文件名，以及 debian 的 patch 文件名等。在测试环境下，38.7 万个文件（目录）中一共有 12 个长度超过 80 个字符的文件（目录）名，有 1888 个长度为 67 个字符的文件（目录）名，以及 4013 个长度为 70 个字符的文件（目录）名。

对于 38.7 万个文件与目录遍历的实测表明，其索引内存占用将超过 2G，这完全是不可接受的。因此，二级索引需要进行优化。

anything 使用的优化是对索引的最大长度加以限制。这是因为在实际使用中，用户真正输入的字符串不会很长，而且在索引串长到一定值之后，得到的原始字符串已经非常少，完全可以使用原始字符串进行二次匹配了。这样优化后，首先，索引数可以减少，其次，每个索引的长度也会减少，再加上使用的是 4 字节的偏移量，而不是 8 字节的指针，因此可以进一步减少内存消耗。下面是根据上述算法得到的内存消耗结果计算：

- 串长 5 时，占用 110 字节
- 串长 10 时，占用 452 字节
- 串长 20 时，占用 1,212 字节
- 串长 30 时，占用 1,972 字节
- 串长 40 时，占用 2,732 字节
- 串长 50 时，占用 3,492 字节
- 串长 60 时，占用 4,252 字节
- 串长 70 时，占用 5,012 字节
- 串长 80 时，占用 5,772 字节

可以发现，内存消耗有了很大的减少，在对上述含有 38.7 万个文件与目录的遍历表明，其索引有大约六百万个，索引占用内存约 230M，程序占据内存不超过 500M（因为动态内存分配需要大量的额外内存空间），基本可以接受。

下面再来看哈希链表的设计。

在基础索引中，如果是 38.7 万文件与目录，则一共需要进行 38.7 万次 strstr 的调用以完成一次完

整的查找。在二级索引中，如果没有哈希链表，则需要对所有生成的索引进行从前至后的 strcmp 调用，一共有约六百万个索引，则需要进行六百万次 strcmp 调用，如果考虑到还对索引进行了最大长度限制，还需要进行一定数量的 strstr 调用以进行二次确认。这样二级索引就完全成了一个笑话了，不仅耗费内存多，还会导致搜索慢，简直一无是处。

如果使用了哈希链表，可以将具有相等哈希值的索引存放到一个数组中，然后将这些数组再存放到一个更大的哈希数组中，使得这六百万个索引尽量均匀地分布开。这样每次查找的时候就可以大大减少 strcmp 的次数了。在这里，anything 采用了质数模的方法作为哈希函数，程序里使用的质数模是一个梅森数，即 131071，如果此哈希函数能将索引值完全均匀分布的话，大约每个哈希数上将会有 $6000/131 \approx 46$ 个索引，当然，在实际中哈希函数不可能绝对均匀，假设其较差的值为 5000 个，那也可以将 strcmp 的次数减少 1200 倍，是一个极大的性能提高了。在较差的情况下，假设有 100 个文件满足要求，则只需要进行一次哈希计算，5000 个 strcmp 函数调用和 100 多个 strstr 函数调用即可找到满足条件的文件名了。对比基础索引搜索需要的 38.7 万个 strstr 调用，确实提高了数十倍的性能。

下面是在测试环境中四个搜索方法的简单对比：

* 类 find：10.1 秒，系统缓存增加 260 MB；

* 树搜索：11 毫秒，存储数据内存需要 30 MB；

* 基础索引搜索：8 毫秒，存储数据程序内存需要 7 MB；

* 二级索引搜索：0.4 毫秒，存储数据程序内存需要 230 MB。

当然，二级索引的问题在于其占用内存实在太，对于 38.7 万个文件与目录的文件数来说，基础索引约占用 7 MB 内存，而二级索引则需要占用约 230 MB 内存。其次，当发生文件系统变更的时候，



基础索引的变更比较容易实现，但是二级索引的更新就相当繁复了，需要遍历所有的索引，找到对应的原始字符串偏移量进行修改。此外，二级索引的文件保存与载入也较为麻烦，因为需要将一整个哈希链表序列化与反序列化，而且当文件系统变更时，一旦变更了索引，则需要进行数百兆甚至上 G 数据的重新序列化，也会增大磁盘压力。

因此，除非是在对文件名搜索速度非常关键的场所或者离线搜索的场景下，不然使用基础索引进行文件名搜索即可满足要求了。

文件系统监控

在上面可以看到搜索提速最重要的原因是省去了系统调用，但是文件系统是会不停变化的，因此 anything 需要对文件系统进行监控，在其发生变化时对内部的基础索引进行修改，以保证搜索的正确性与实时性。

everything 对于文件系统变更的追踪相对简单一些，因为它直接依赖于 Windows 下 NTFS 文件系统的日志。但是在 Linux 下，首先文件系统繁多，例如 RHEL 7/CentOS 7 采用的缺省文件系统是 XFS，但是 Debian 与 RHEL 6/CentOS 6 使用的缺省文件系统却是 ext4，而且这两者的文件系统日志都没有 NTFS 全，除此之外还有 btrfs 等一系列的其它文件系统。其次是在 Linux 下，管理员或者用户对于文件系统的选择较多，并且可以深入调整文件系统的参数，例如去掉日志，因此仅依赖于文件系统日志检查文件系统的变更是不太可靠的。

另外，由于 Linux 本身没有全局的文件创建、删除与重命名的用户态监听接口（比较接近的 inotify 无法递归监控目录），因此要解决此问题，只能使用独立的模块监听文件的创建、删除与重命名，并通知相应的用户态程序更新文件系统数据与索引数据。

在 Linux 下，监控整个文件系统的变化可以采用用户态或内核态的方法，其中用户态的方法包括：

- 定时遍历文件系统，对整个文件系统的数据进行更新，这种方法需要每次都进行全量扫描，因此浪费极大，早期 GNU findutils 里的 updatedb 即提供了类似的功能；

- 使用 preload 库，监控 glibc 对应函数的参数与返回值，包括 open、fopen、creat、mkdir、rmdir、rename 等。这种方式的优点是不依赖于内核，但是会对所有依赖于 glibc 库的程序都造成影响，影响面大，工作量也不小（函数较多，参数处理较多），而且具有较大的安全隐患，此外，对于静态编译 libc 库的程序，以及不依赖于 libc 库的程序（虽然这种程序很少）无法监控；

- 使用 inotify，递归监控所有子目录，监控资源消耗极大；

- 使用审计系统，加入对相应系统调用（open、creat、mkdir 等）的审计，通过审计日志检查系统调用的结果，根据结果更新文件系统数据。其优点是不依赖于内核，系统调用数量较少，但是缺点是需要依赖于审计系统，而且事后处理日志的方式会缺少关键的场景信息，例如之前的文件或者目录是否存在。

除此之外，还可以采取编写内核模块的方法来进行监控：

- 使用内核 tracepoint 特性对系统调用入口与出口进行事件跟踪，并记录处理结果，缺点是对所有的系统调用进行跟踪，因此需要自己过滤，而且系统调用路径较长，可能会导致较多的资源占用；

- 通过 kprobes 对内核函数进行挂钩，在函数入口和出口处进行参数与返回值进行检查，当发现满足要求的文件事件时将事件信息记录下来，其缺点

是对比 tracepoint 来说，kprobes 从理论上来说依赖的函数变化的可能性更大一些，当内核升级时可能会带来维护的问题；

- 通过 ftrace 对内核函数进行挂钩，ftrace 处理内核函数挂钩比 kprobes 更方便，没有架构相关的问题，而且不需要考虑 kprobes 中不能休眠的问题，但是 ftrace 较 kprobes 更新一些，需要更高版本的内核支持，部分架构的内核尚未提供完善的 ftrace 功能。

anything 现在选择的是 kprobes 功能，主要是考虑到 kprobes 的支持更广泛些，而且监控的内核函数也相对稳定。具体到需要跟踪的内核函数，主要就是 VFS 的文件系统变更函数，包括 vfs_create 等，为了确保只有当函数成功返回时才进行记录，anything 需要使用 kretprobes 进行函数跟踪。

需要注意的是，使用 kretprobes 有一个与架构相关的问题，那就是要在函数入口处得到函数各参数的值，而在这里，kretprobes 没有给开发者提供很多便利，需要根据内核的寄存器结构体 (pt_regs) 来根据架构的函数调用规约获取相关的参数值。例如对于 x86 来说，其 64 位系统的函数调用规约是从第一个参数开始分别使用 rdi、rsi、rdx、rcx、r8 与 r9 寄存器保存参数，从第七个参数开始使用栈来保存剩余的参数。这些代码都被封装到源码的 kernelmod/arg_extractor.c 里了，因此，当需要扩展架构支持时，主要在这里进行修改即可。此外，在实际使用中，anything 现在仅用到了前四个参数，因此只要处理 n 等于 1 到 4 的情况就够了。

内核模块对外提供了 /proc/vfs_changes 文件以向用户态的应用程序提供文件系统更新信息的访问接口，当应用程序通过此文件获取到文件系统更新信息后，即可更新对应的基础索引数据了。由于基

础索引的数据结构设计使得一个目录树被保存在连续的一段内存里，因此对文件系统的更改都非常方便，最大的开销就是 memmove 而已，这个变更速度在 x86 上，对于 10M 的内存，只需要耗费毫秒级的时间。

后记

anything 是一个小巧的软件，它的功能很单一，开发时间也不长，总共的开发时间大概只有四个多星期，中间还经历了项目的打扰。但是它也达到了自己的目标，即为文件名搜索提供一个高效快速的实现，而且对现有的系统侵入很小。不过，由于时间的限制，二级索引的更新没有做完，留了一个尾巴，希望以后能有机会完成吧。

展望未来，我们认为 anything 可以在今后考虑下面的改进：

- 使用 ftrace 代替 kprobes，以避免处理 kprobes 中无法休眠的问题，以及与架构相关的处理逻辑；

- 索引全部落盘，而不是放在内存中，以提高伸缩性，比如可以支持数千万个文件的同时内存仅需要十兆左右；

- 兼容 GNU findutils 的 locate 命令的语法；

- 使用 eBPF 等功能对 VFS 函数进行跟踪，以避免使用内核模块。

希望大家能喜欢这个开源软件，并能为其提交 PR。当然，anything 现在仅支持文件名搜索，如果需要支持使用其他文件元数据进行搜索，也是可行的。但如果是要支持全文检索，可能就需要使用全新的设计了，这就又是一个新的问题了。

到这里，就到这里吧。d



Hello World CUDA !

● 深度科技 武汉公司 研发部 / 文

引言

这是很久以前跟师弟了解 CUDA 时写的一个文章，还挺有意思的，涉及内容比较简单，一些皮毛。抛砖引玉，和大家分享一下。

我们经常开玩笑说自己精通十种语言的 Hello, World 的写法。这周因为要阅读研究 YOLO，在其中有一定的 CUDA 代码，所以写下了这篇报告“Hello, World 的 CUDA 写法”。什么是 CUDA 呢，来自百度百科的解释是 CUDA (Compute Unified Device Architecture)，是显卡厂商 NVIDIA 推出的运算平台。CUDA™是一种由 NVIDIA 推出的通用并行计算架构，该架构使 GPU 能够解决复杂的计算问题。

开始

安装过程就不在赘述，因为就是点击下一步。本机上使用到的版本是 VS2017&CUDA 9.0 正式版。

在开始学习 CUDA 的 Hello, World 写法之前我们先写一个 C 的 Hello, World。这个应该非常简单吧。

```
#include

int main()
{
    printf("Hello,World");
    return 0;
}
```

然后将这个程序转化成一个 CUDA 程序需要加上一些语句，加上之后的程序段就是这个样子的。

```
#include
#include "cuda_runtime.h"

__global__ void helloKernel()
{
}

int main()
{
    helloKernel <<>> ();
    printf("Hello,World");
    return 0;
}
```

这个程序段和之前的 c 语言的 Hello, World 相比多了一个空函数 helloKernel 并且这个函数带有 global 修饰符，对于这个空函数的调用，需要用 <<>> 来修饰。

这个函数就是我们在 CUDA 编程中会经常使用的核函数。这个核函数在官方的手册中的说明是这样的：

A kernel is defined using the global declaration specifier and the number of CUDA threads that execute that kernel for a given kernel call is specified using a new <<>> execution configuration syntax (see C Language Extensions). Each thread that executes the kernel is given a unique thread ID that is accessible within the kernel through the built-in threadIdx variable.

使用 __global__ 声明说明符定义内核函数，并使用新的 <<>> 执行配置语法指定为给定内核调用执行该内核的 CUDA 线程数。执行内核的每个线程都被赋予一个唯一的线程 ID，该内核可以通过内置的

threadIdx 变量在内核中访问。

对于一个 CUDA 初学者来说，官方的例子肯定是最好的学习手段。我们下面就从一个官方给出的矩阵相加的例子来初步的学习 CUDA 在 GPU 上的编程是怎么实现的。

sample 学习

在开始学习之前我先将整个代码贴出来，这个代码是用 vs2017 创建 CUDA 项目之后自动生成的代码。

```
#include "cuda_runtime.h"
#include "device_launch_parameters.h"

#include

cudaError_t addWithCuda(int *c, const int *a, const int *b,
    unsigned int size);

__global__ void addKernel(int *c, const int *a, const int *b)
{
    int i = threadIdx.x;
    c[i] = a[i] + b[i];
}

__global__ void helloKernel(void)
{
    printf("hello");
}

int main()
{
    const int arraySize = 5;
    const int a[arraySize] = { 1, 2, 3, 4, 5 };
    const int b[arraySize] = { 10, 20, 30, 40, 50 };
    int c[arraySize] = { 0 };
    // Add vectors in parallel.
    cudaError_t cudaStatus = addWithCuda(c, a, b, arraySize);
    if (cudaStatus != cudaSuccess) {
        fprintf(stderr, "addWithCuda failed!");
        return 1;
    }

    printf("[1,2,3,4,5] + [10,20,30,40,50] = {%d,%d,%d,%d,%d}\n",
        c[0], c[1], c[2], c[3], c[4]);

    // cudaDeviceReset must be called before exiting in order for
    // profiling and
    // tracing tools such as Nsight and Visual Profiler to show
    // complete traces.
    cudaStatus = cudaDeviceReset();
    if (cudaStatus != cudaSuccess) {
```

```
        fprintf(stderr, "cudaDeviceReset failed!");
        return 1;
    }

    return 0;
}

// Helper function for using CUDA to add vectors in parallel.
cudaError_t addWithCuda(int *c, const int *a, const int *b,
    unsigned int size)
{
    int *dev_a = 0;
    int *dev_b = 0;
    int *dev_c = 0;
    cudaError_t cudaStatus;

    // Choose which GPU to run on, change this on a multi-GPU
    // system.
    cudaStatus = cudaSetDevice(0);
    if (cudaStatus != cudaSuccess) {
        fprintf(stderr, "cudaSetDevice failed! Do you have a CUDA-
        capable GPU installed?");
        goto Error;
    }

    // Allocate GPU buffers for three vectors (two input, one
    // output) .
    cudaStatus = cudaMalloc((void**)&dev_c, size * sizeof(int));
    if (cudaStatus != cudaSuccess) {
        fprintf(stderr, "cudaMalloc failed!");
        goto Error;
    }

    cudaStatus = cudaMalloc((void**)&dev_a, size * sizeof(int));
    if (cudaStatus != cudaSuccess) {
        fprintf(stderr, "cudaMalloc failed!");
        goto Error;
    }

    cudaStatus = cudaMalloc((void**)&dev_b, size * sizeof(int));
    if (cudaStatus != cudaSuccess) {
        fprintf(stderr, "cudaMalloc failed!");
        goto Error;
    }

    // Copy input vectors from host memory to GPU buffers.
    cudaStatus = cudaMemcpy(dev_a, a, size * sizeof(int),
        cudaMemcpyHostToDevice);
    if (cudaStatus != cudaSuccess) {
        fprintf(stderr, "cudaMemcpy failed!");
        goto Error;
    }

    cudaStatus = cudaMemcpy(dev_b, b, size * sizeof(int),
        cudaMemcpyHostToDevice);
    if (cudaStatus != cudaSuccess) {
        fprintf(stderr, "cudaMemcpy failed!");
        goto Error;
    }

    // Launch a kernel on the GPU with one thread for each
```



```
element.  
addKernel<<<>>(dev_c, dev_a, dev_b);  
  
// Check for any errors launching the kernel  
cudaStatus = cudaGetLastError();  
if (cudaStatus != cudaSuccess) {  
    fprintf(stderr, "addKernel launch failed: %s\n", cudaGetErrorString(cudaStatus));  
    goto Error;  
}  
  
// cudaDeviceSynchronize waits for the kernel to finish, and returns  
// any errors encountered during the launch.  
cudaStatus = cudaDeviceSynchronize();  
if (cudaStatus != cudaSuccess) {  
    fprintf(stderr, "cudaDeviceSynchronize returned error code %d after launching addKernel!\n", cudaStatus);  
    goto Error;  
}  
  
// Copy output vector from GPU buffer to host memory.  
cudaStatus = cudaMemcpy(c, dev_c, size * sizeof(int), cudaMemcpyDeviceToHost);  
if (cudaStatus != cudaSuccess) {  
    fprintf(stderr, "cudaMemcpy failed!");  
    goto Error;  
}  
  
Error:  
cudaFree(dev_c);  
cudaFree(dev_a);  
cudaFree(dev_b);  
  
return cudaStatus;  
}
```

这个代码不算太长，但是实现了调用 GPU 进行矩阵运算的所有功能。

在进行 CUDA 编程之前，我们需要包含 CUDA_RUNTIME 版本的头文件，然后我们就可以定义我们的核函数了。在 CUDA 中定义核函数很简单，只需要将一个函数前面加上 global 修饰就可以将这个函数定义为一个核函数。CUDA 程序的编译是由 nvidia 公司提供的 nvcc 编译器进行编译的，所以我们在安装的时候一定要确定自己的 nvcc 编译器运行没有错误。

我们从 main 函数开始阅读这段代码，这样可以很清晰的看出代码的运行过程。

```
int main()  
{  
    const int arraySize = 5;  
    const int a[arraySize] = { 1, 2, 3, 4, 5 };  
    const int b[arraySize] = { 10, 20, 30, 40, 50 };  
    int c[arraySize] = { 0 };  
    // Add vectors in parallel.  
    cudaError_t cudaStatus = addWithCuda(c, a, b, arraySize);  
    if (cudaStatus != cudaSuccess) {  
        fprintf(stderr, "addWithCuda failed!");  
        return 1;  
    }  
  
    printf("{1,2,3,4,5} + {10,20,30,40,50} = {%d,%d,%d,%d,%d}\n",  
        c[0], c[1], c[2], c[3], c[4]);  
  
    // cudaDeviceReset must be called before exiting in order for profiling and  
    // tracing tools such as Nsight and Visual Profiler to show complete traces.  
    cudaStatus = cudaDeviceReset();  
    if (cudaStatus != cudaSuccess) {  
        fprintf(stderr, "cudaDeviceReset failed!");  
        return 1;  
    }  
  
    return 0;  
}
```

首先 main 函数中定义了我们要相加的两个数组的大小和里面的数据。然后调用了 addWithCuda 的函数，这个函数会返回一个状态，这个状态是 CUDA 使用的错误状态码，在调用完相加程序之后，我们可以用这个状态码来判断程序是否正确的运行了，并且得到了正确的结果。

```
cudaStatus = cudaDeviceReset();  
if (cudaStatus != cudaSuccess) {  
    fprintf(stderr, "cudaDeviceReset failed!");  
    return 1;  
}
```

这个函数是 CUDA 编程里面比较重要的一个函数，这个函数主要的作用就是在当前流程中销毁所有分配并重置当前设备上的所有状态。在官方手册中的解释是这样的：

When a host thread calls cudaDeviceReset(), this destroys the primary context of the device the

host thread currently operates on (i.e., the current device as defined in Device Selection). The next runtime function call made by any host thread that has this device as current will create a new primary context for this device.

当主机线程调用 `cudaDeviceReset()` 时，这销毁了主机线程操作的设备的主要上下文。任何以这个设备为当前设备的主机线程调用的运行时函数将为设备重新建立一个主要上下文。

CUDA 上下文类似于 CPU 的进程。所有资源和在驱动程序 API 中执行的操作都封装在 CUDA 上下文内，在销毁上下文时，系统将自动清理这些资源。

然后就到了我们的相加函数中，在这个函数中，有几个调用函数比较重要，分别是：

```
cudaStatus = cudaSetDevice(0);
cudaStatus = cudaMalloc((void*)&dev_c, size * sizeof(int));
cudaStatus = cudaMemcpy(dev_a, a, size * sizeof(int),
cudaMemcpyHostToDevice);
addKernel<<>>(dev_c, dev_a, dev_b);
cudaStatus = cudaGetLastError();
cudaStatus = cudaDeviceSynchronize();
cudaFree(dev_c);

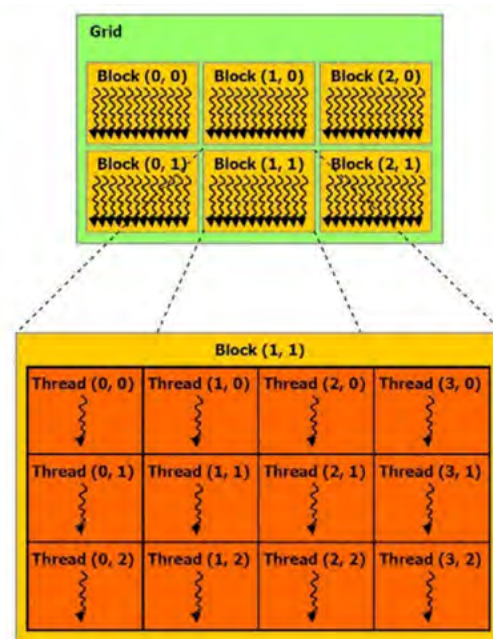
cudaStatus = cudaSetDevice(0);
```

在 CUDA 编程中，我们需要选择设备上的某一块显卡来执行我们的计算操作，这个代码就是选择设备商的第 0 块显卡来执行我们的程序。这个语句对于多 GPU 的设备非常重要，因为 CUDA 可以分别让不同的 GPU 执行不同的任务。

`cudaMalloc`、`cudaMemcpy` 函数和我们平时在 C 语言下使用的 `Malloc` 和 `Memcpy` 函数非常相似。为什么我们需要这两个函数呢。因为大家都知道的

是，我们使用 GPU 进行运算，这个运算的位置是位于 GPU 的显存上，那么又有人会去问什么是显存呢？显存全称显示内存，亦称帧缓存，它是用来存储显示芯片处理过或者即将读取的渲染数据。如同计算机的内存一样，显存是用来存储图形数据的硬件。显存在 GPU 中相当于电脑中的内存，GPU 进行运算的时候是由 GPU 处理核心从显存中提取数据然后进行运算，然后将运算的结果再存放到显存中。那么为什么我们需要将内存中的内容拷贝到显存中呢？因为，显卡和内存之间的带宽并不是想象中的那么大，在高速运算的情况下，这种带宽的限制可能就成了我们计算速度的瓶颈，所以，我们在使用 GPU 加速运算的时候，是将数据先拷贝到显存中，然后再利用 GPU 的运算特性来进行运算加速。

下面就要讲到编程中最重要核函数了。在讲解核函数之前我们要先了解一下关于 CUDA 的一些基本的架构。





志愿者装机活动

深度操作系统的发展离不开社区的支持，同时我们也会及时响应社区用户的需求。为了更好的服务、推广、宣传深度操作系统，让更多的社区用户参与线下互助和传播，通过社区用户反馈的建议，我们推出了免费装机地图活动。

本着奉献、友爱、互助、进步的志愿者精神，志愿者们利用业余时间，帮助希望了解 deepin 操作系统的爱好者们，免费上门安装系统，过程中不仅丰富了自己的专业知识，也结识了更多 Linux 爱好者。

活动开始至今，已覆盖全国 31 个省市，参与志愿者近 300 余人。

捐助渠道

深度操作系统的发展离不开社区的支持，为了进一步完善深度操作系统社区生态环境的建设，深度科技对外正式开通捐助渠道。

捐助秉承着完全自愿的原则，不管你是个人还是组织、来自何处、捐助金额多少，都是对社区的发展贡献出自己的力量，我们对你的捐助深表感谢。

捐助定位：

- 深度操作系统是一个致力于为全球用户提供美观易用、安全可靠的 Linux 发行版。
- 深度操作系统的发展离不开社区的支持，为了进一步完善深度操作系统社区生态环境的建设，深度科技内部通过决策，对外正式开通捐助渠道。
- 捐助秉承着完全自愿的原则，不管你是个人还是组织、来自何处、捐助金额多少，都是对社区的发展贡献出自己的力量，我们对你的捐助深表感谢。

捐助目的：

- 调查各个项目参与捐助的金额和人数，决定项目开发的方向和优先级
- 持续和深入的开发
- 社区活动和建设（例如：内测活动奖品、志愿者奖励以及社区礼品）

 Deepin Talk ¥ 1480.15 已捐助 98 位支持者		
 驱动中心 ¥ 6495 已捐助 337 位支持者		
 Deepin Live 修复系统 ¥ 1192.66 已捐助 80 位支持者		
 Deepin ID 云平台 ¥ 1115.65 已捐助 66 位支持者		
 深度开发者平台 ¥ 3286.58 已捐助 193 位支持者		



国产办公软件，从涅槃重生到跨越发展

● 中国电子报 葛珂 / 文

中文办公软件时代的开启

2018年，这是个非常有意义的时间点，它不仅是我国改革开放40周年，也是金山成立和WPS产品研发的30周年。

在30年前，金山创始人之一求伯君在没有编程环境的条件下，用了128万行接近计算机语言的汇编语言创造性地做出了WPS 1.0，用一种极其传奇的方式开启了整个计算机中文办公软件的使用元年。

在WPS 1.0问世之前，大家都是用纸、笔、打字机处理文件，先进一点的用富通或联想的电子打字机处理文件，而带有图文混排和表格功能的WPS 1.0的问世，使得中文信息的呈现方式更加鲜活，也意味着我们真正进入了中文排版、中文办公的时代。

经历过那个年代的人都知道，当时的电脑培训非常盛行，培训的内容主要有两项：一项是五笔字型，另一项就是WPS。如果那时的你不会使用五笔，或者不会使用WPS，都不好意思说会使用电脑。

WPS是那个时代办公软件的象征，从1988年开始到1995年，WPS占有当时90%的市场。在那个辉煌的时代，还有很多其他软件公司成长发展起来，比如做操作系统的、做汉化DOS的、做汉化Windows的等等，大家共同支撑了我国软件产业初期的发展。

重压困境下的涅槃重生

事物的发展不是一帆风顺，而是前进性和曲折性的统一。从1995年至2001年，在跨国公司和盗版的双重压力之下，国内软件企业的生存空间受到极大的挤压，很多软件企业开始走下坡路，有的走向没落，有的走向转型，软件产业规模一度萎缩，许多企业不得不退出竞争，离开这个行业。当时，

WPS变成了一个小众的、非主流的产品。

2001年，国家修订了著作权法，公布了《计算机软件保护条例》，加强了打击盗版侵权的力度，同时推动了从操作系统到应用软件的整体规划。特别是针对办公软件，国家在政务采购中开始积极支持国产软件，2002年，金山WPS、永中Office等国产软件公司拿到了当时相当大的政府订单份额。

在国家支持的同时，我们也加紧自主创新的研发。那时，当WPS软件被用户使用时，最直观的反馈就是没法用，不会用，文档兼容不了。为了解决这些问题，2003年，我们将账面上的3000万元与国家拨付的800万元全部投入到了研发，历时两年，重新编写了WPS，推出了WPS Office 2005。同年，我们又郑重承诺WPS Office个人用户永久免费。正是这一系列的动作，让WPS重获新生。在当时，WPS成为了PC时代用户最喜欢下载的几款软件之一，并成为当时的百度下载排行榜前十名中唯一的一款商业软件。2005年，对于整个WPS来讲成为关键的一年。

从1988年到2005年之间，中国软件产业经过了17年的跌荡起伏，对金山WPS来讲，曾经一度占有90%的市场，然后变为小众，又重生回到舞台。这些年中，WPS始终抱着理想主义者的心态，一直坚守着初心与信念，坚信中国软件业一定能成功，执着地在办公软件领域耕耘。

移动互联网时代的跨越腾飞

随着移动互联网大潮的来临，2011年，在移动办公行业前景还不明晰的情况下，作为国产办公软件的代表，我们集中研发力量，投入到办公软件移动版的开发。先后做出了WPS Android版、WPS

iOS 版，并成为当时全球领先的移动 Office 产品。

在研发初期，我们也彷徨过，觉得 PC 盛行的年代做一个办公软件放在手机上，似乎没什么用，但随着移动互联红利的爆发，WPS 以每年 300% 的速度高速增长时，充分验证了我们决策的正确。目前，WPS 全线产品累计月度活跃用户已经超过 3 亿户，覆盖了全球 200 多个国家和地区，远超其它同类产品。WPS Office 移动版成为国产办公软件把握移动互联网发展趋势的很好的例证，也成为办公软件发展历程中一次重要的产品创新，同时也成为金山办公软件在移动互联网的大潮中取得先机、实现弯道超车的标志。

在开发移动版产品的同时，WPS 也代表国产办公软件走向海外，推动与“一带一路”沿线国家和地区的软件合作。2017 年，WPS Office 泰文版在泰国首都曼谷发布。这是我国办公软件产品首次落地泰国，也是国产通用软件进入国外政府采购序列的一次成功尝试。现今 WPS 已拥有 46 种语言，在 200 多个国家软件下载排行榜中居于前列，既包括印度、泰国等发展中国家，也包括美国、法国、德国、西班牙等一系列发达国家。移动互联时代，WPS 正以一种崭新的方式实现着跨越腾飞。

挥别传统 Office 开启办公新未来

未来办公的方向是什么？在我看来，“云、多屏、内容、AI”将会成为未来办公创作的“四大件”。因为他们让用户摆脱了格式的束缚。在办公服务阶段，我们产生了大量的数据，这些海量的数据有用户的消费数据、使用数据等。数据的处理，不再只是通过计算机处理，而是需要运用人工智能等新技术，所以大数据、云计算、人工智能等将是所有软件企业面对的革命性变化。今年 7 月，金山 WPS 结合“云、多屏、内容、AI”的特点发布了 WPS Office 2019 等新品，并依托国内最大的文档库资源积极探索新技

术在办公软件中的应用。

对于文字创造者来说，错别字一直是很多用户最头疼却没有好办法解决的难题。未来，WPS 将推出基于机器学习的 AI 校对，帮助用户定位错字、措辞和自动修正，让繁琐的文字创造变得更加简单。在智能校对的过程中，用户也能参与训练 AI 办公助手的算法。随着时间的推移，我们希望 Office 助手能够做到理解用户的每一句话、每一段文字，通过理解语义来帮助大家创作。同时，WPS 还将通过与海量优质模板素材的匹配，自行完成智能图文排版，一键将 PPT 达到美颜效果，并可通过 AI 扫描自动完成照片转文档。让大家从格式中真正解脱出来，专注于创作。

今天的 WPS Office 已不再是大家原来想象中的文件、表格、邮件等传统组件化的办公软件，而是基于 Linux、Android、国内所有 CPU、电视投影等的多屏软件，是具有模板、素材、图片的内容软件，是基于协作办公的云端软件，是附有 AI 技术的智能软件。

WPS 最痛苦的时代已经过去，未来我们将利用技术门槛、品牌门槛，依托运营能力、产品技术能力、开发能力，实现人与人、人与设备、内容与创作的结合，使创作场景不再受时空的限制。只需一次下载、一个图标、一个账号，即可打开任何一种文档，只要打开 WPS，所处理文档即触手可及。为创造者服务，是我们最简单而又最有力的愿景。

30 年的金山，见证了改革开放以来我国软件产业的发展历程：从闭门造车、与狼共舞到涅槃重生、跨越腾飞再到我们开启新的未来。30 年对很多行业来讲是很短的历程，但对中国软件行业来讲，它是从无到有，从最初起步到今天辉煌的全过程。未来，金山 WPS 将会继续砥砺前行，不忘初心，不断创新，为大众打造更加满意的国产办公软件，让未来办公变得更加轻松、简单。d

文章节选自《中国电子报“改革开放 40 年·软件产业”特刊》



Linux 是否适合作为桌面系统使用？ 或许你该试一试 Deepin Linux

● 头条号作者 EmacserVimer / 文

目前来说，要将 Linux 作为桌面解决方案，对于大多数 PC 用户来说，当然是不现实的，毕竟 Linux 的主力用户群体依然是少数极客用户，说白了就是开发者。

造成这个现状的当然有很多历史原因，一个就是 Linux 自带极客血统，一个就是尽管现在有很多的桌面环境如 Gnome、KDE、XFCE、MATE 等等，但是比起 macOS、Windows 这样的成熟的消费级桌面解决方案依然有很大的差距。加上绝大多数 PC 用户已经习惯了 macOS、Windows 这样的桌面系统，Linux 的软件生态不够丰富，自然除了少数专业用户，普通用户不太可能会选择 Linux 作为自己的桌面解决方案。

不过现在发生了一些改变，很多出色的 Linux 发行版做得越来越好，最具代表性的就是来自国内深度科技开发的 Linux 发行版 Deepin Linux，这是我觉得目前最好的 Linux 桌面解决方案，基本上可以成为大多数用户的主力桌面系统。



深度系统安装

安装足够简单 + 精美的深度桌面

能作为主要的桌面系统，那么一定是要安装过程够简单，如果对于一个 Linux 入门用户，或者说是一个普通用户，给他推荐 Arch Linux/Gentoo Linux 自然是不现实的，估计 wiki 看了半天都不敢下手，而深度的安装过程就足够简单，甚至可以说比 Windows 和 macOS 的安装过程都要简单。你只需要做好启动盘，安装的时候设置好用户名、密码等简单步骤直接安装就可以了，即使是一个小白通过简单的查看文档也可进行安装。



深度文件管理器

就深度 Linux 的桌面环境来说，说是目前最酷的桌面环境一点都不夸张。深度桌面环境主要由桌面、启动器、任务栏、控制中心、窗口管理器等组成。

深度 Linux 系统的桌面风格加入了 HTML5 的风格，通过 H5、QT 技术让这个桌面非常美观，非常炫酷，

应用启动器的有两种模式，尤其是分类模式非常好用。在系统主题方面，深度团队做了两套窗口主题、五套图标主题、以及四套光标主题，可以说每一套都是深度团队用心做的。

深度系统的 Dock 栏也非常漂亮，比较接近 macOS 的 dock 栏风格，对于我这种 macOS 重度用户来说，学习成本、迁移成本也非常低。

基于 Debian、对开发者友好

基于 Debian，这就意味着 Deepin Linux 是对于开发者友好的。

这一点对于大部分 Linux 用户来说，都是重要的指标，毕竟对于大多数的 Linux 用户来说，目前还都集中在开发者群体，普通用户的比例还不够多，因此对于开发者友好就非常重要了。



深度系统监视器

深度 Linux 是基于 Debian 的发行版，Debian 可以说是 Linux 领域里面最老、最稳定、以及软件生态最为丰富的发行版之一了。Debian 是很多 Linux 发行版的基础，包括 Ubuntu、Linux Mint 等发行版的发展基石，Debian 的 Dpkg 和 Apt 软件包管理，可以说是最好的 Linux 软件包管理之一。

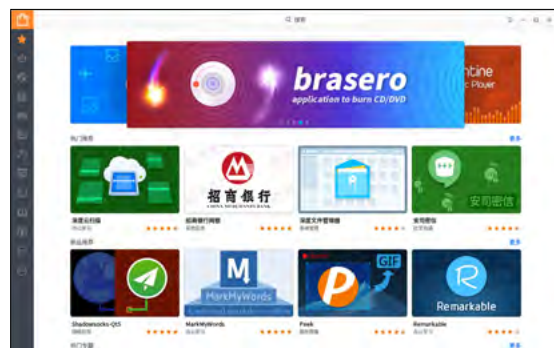
Debian 最大的优点易于使用、而且非常稳定靠谱，是很多系统管理员、软件工程师的首选，不论是作为桌面、服务器、测试、以及开发都是非常好的选择。

深度 Linux 很好的继承了 Debian 的血统，自然对于开发者也是友好的。

软件生态丰富、并且主流应用都已经上了应用商店

要想能成为好的桌面系统，软件生态简直是太重要了，而深度科技在这一点上做出的努力，可以说是非常有诚意。为了将深度 Linux 打造成用户友好的桌面系统，深度科技自己开发以及和第三方应用团队合作，做出了很多应用软件优化，而且本地化做得非常好。

当然基于 Debian，也就代表支持 Debian 的应用，你基本上可以没有障碍的拿过来就直接使用，这就是站在巨人肩上看看得远的道理。最关键的是，深度将 Linux 下安装软件的哲学简化了，毕竟对于大多数 Linux 发行版都提倡命令安装，目前大部分发行版的应用商店做得一团糟，但是深度商店绝对是诚意之作，说用户友好不为过。



深度商店



深度团队开发了 26 款原生应用

最具代表的，原深度团队 CTO 王勇在自己工作业余时间写了深度终端、深度系统监视器，深度总工程师张磊写了深度文件管理器。

深度团队还打造或者重写了深度云打印、深度云扫描、深度演示助手、深度商店、深度影音、深度录屏、深度编辑器、Deepin Emacs 等众多经典的应用。

开发者应用应有尽有，而且都已经在深度商店上线

对于大多数用户来说，应用生态其实很大程度上是用户选择的最终原因。而且几乎绝大多数常用的软件，不论是开发工具、娱乐工具、以及办公软件都已经在深度商店上线了，你不用再去一个个的敲命令，这对于初学者来说，这样的使用都是没有门槛的。

程序员最常用的文本编辑器，比如 Emacs、Vim、Visual Studio Code、Atom、Sublime Text 等等，这些文本编辑器你拿过来就可以直接使用；比如集成开发环境，Eclipse、IntelliJ IDEA、IntelliJ WebStorm、Android Studio、Navicat、Genymotion、Clion 等等众多的集成开发环境都已经在应用商店上线了。



深度终端

办公、娱乐软件非常全

要做思维导图怎么办？亿图、Xmind 深度商店有！

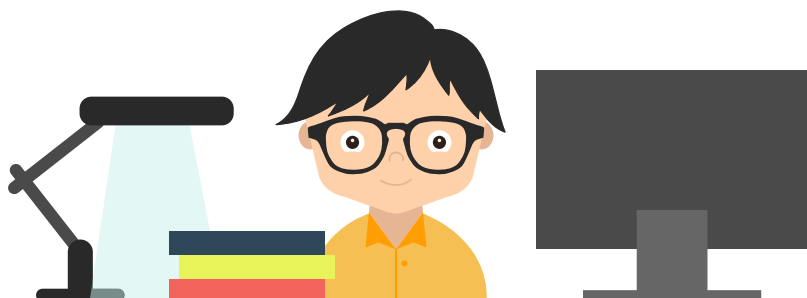
要发邮件怎么办？Foxmail、Thunderbird 深度商店也有！

要编辑、排版、记笔记怎么办？WPS、LibreOffice、永中 Office、为知笔记、蚂蚁笔记、OneNote 深度商店同样有！甚至你要用 Tex、Markdown 进行工作，TexStudio、Cmd Markdown 这些应用商店同样有！

除此之外，网易云音乐你想不想用？迅雷你想不想用？有道词典你想不想用？微信、QQ 你想不想用？钉钉你要不要你用？企业微信你要不要你用？搜狗输入法要不要？五笔输入法你要不要你用？..... 对的，这些通通都有！而且要么自带了要么都在应用商店上线了！

说了这么多，简单的总结一下就是，对于一个好的桌面系统，安装过程一定要足够简单，桌面体验一定要足够好，软件生态一定要足够丰富、并且有一个良好的应用商店。深度 Linux 真正做好了这些事情，而且在本地化方面做了很多优化，可以说诚意满满。

作为一个开发者，其实非常感谢有深度科技这样的团队还在默默地为中国的操作系统事业做出贡献，我曾经在很多场合都表述过，只要有这样的团队，我就愿意为这样的团队打 Call，毕竟国产操作系统、国产处理器，这是所有行业人的梦想！



deepin 集结 | 征稿启事

《deepin 集结》是深度科技内部刊物，它不仅是记录公司发展历程的一本“画册”，也是公司对外形象和企业建设的窗口。

自 2016 年伊始至今，《deepin 集结》已有定期发放给热爱深度的用户及合作伙伴，并得到了大家的认可。在收集各位读者的反馈意见后，我们开设了新栏目——深度伙伴，主要针对深度员工以外的人员投稿，使《deepin 集结》成为一个企业与用户沟通的刊物，彼此增加交流，分享开源技术。

内部征稿

投稿内容及要求：

形式：摄影作品

Ps. 不要叫你的单反在家睡大觉了，赶紧出来发挥作用吧！

绘画书法

Ps. 你不发出来怎么能知道原来你还这么有才！

诗歌散文、游记、人生感悟

Ps. 知道你原来话不多，但是肚子里面还是有墨水的！

经验技术分享

Ps. 专业达人们，把你们的专业知识拿来 show 一下吧，科普一下啦！

以及能落实到纸上的任何才艺

Ps. 还有多少是我意想不到的，快点告诉我！

外部征稿

投稿内容及要求：

1. 内容：a 技术分享

b 用户体验

c 项目评价

2. 要求：a 与行业相关

b 字数 1000-4000

c 文中图片需清晰

深度的同学可以告知身边爱好开源的发烧友积极投稿哦！

投稿邮箱主题需标明：外部投稿（字样）+ 姓名 + 手机号码

要求：我们很看重有图有真相哦，图片和文字说明一样重要。

稿酬：现金（微信红包）或精美礼品

投稿方式：qindi@deepin.com 邮件名称一定要注明“所在城市 - 部门 - 姓名”

诸位同学有任何问题，都可以立即马上咨询我们：

qindi@deepin.com



走吧，去太平洋看海

● 深度科技 北京公司 技术部 / 文

这次出游要从前段时间一次去幼儿园接娃说起。

话说这天下班后，我去幼儿园接娃，接到娃后跟老师简单聊了几句准备回家，老师突然说：“听孩子说你们要带他去塞班玩，多给孩子照点照片”。

我：“……？！啊，哦，呵呵，好的好的……”，匆匆跟老师再见。

出了学校，我问娃：“你跟老师说你要去塞班玩？”

娃说：“恩，是你告诉我的啊，要去塞班玩”

我使劲想了想，忽然想起前几天在家里跟老婆聊起来同事去塞班玩，感觉不错，娃听到后竟然顺势去幼儿园告诉老师和小朋友，爸爸妈妈要带他出去旅游，需要好几天不能来幼儿园……

于是，在娃制造的“舆论”中，我们收拾行囊，开启了这场深蓝之旅。

上午十点钟从家出发去机场，经过几乎一整天的折腾，五个小时的航班，终于在当地时间凌晨一点降落在塞班机场。然后又是将近两小时的入关、大巴、酒店入住，躺在床上已是凌晨三点，不可谓不累啊，倒头就睡。

早上，完全没有做好起床准备的我被一声清脆的惊呼叫醒，迷迷糊糊的来到阳台，一路的劳顿和早起的睡意瞬间被眼前的景象一扫而光，第一次见到这样的彩虹，仿佛一道架在海上的彩色拱桥，仿佛上帝随意的一笔彩绘……强烈的感到人类的辞藻和表达在自然面前显得如此苍白。

短暂的陶醉后，开始了第一天的行程，环岛北部游，参观了当年日军在岛上的司令部和著名的自杀崖，看着眼前太平洋海水惊涛拍岸的气势，遥想



当年数千人纵身跳海的惨烈，不禁感慨战争的残酷，和平的珍贵。

告别了令人沉重的自杀崖，来到鸟岛，这里是看日出的绝佳地点，也是情侣必须拍照打卡的景点，向往爱情想象力丰富的人们把这座 mini 小岛描绘成依偎在爱人臂弯里的少妇，的确是寓意美好，引得一众新老夫妇们争相在这里合影。

回来的路上，看到了当地独有的“好男人花”，仔细看会发现，这花没有花心，也就是“不花心”，不花心就是好男人。这时我的自然知识又开始捉急，没有花心就是没有花蕊，没有花蕊怎么繁殖，不能繁殖能算得上好男人么？

下午回到酒店后，迫不及待的和深蓝的大海进



行了一次亲密接触。海上遛娃，我还是第一体验，要点是，不管娃怎么踢你，你都要抓紧绳子。

第二天的行程主要是军舰岛，据说这个岛从空中鸟瞰形似一艘军舰，故而得名，且传闻当年美日争夺马里亚纳群岛的战役中，在夜袭时一度把这座小岛当成一艘军舰，投入大量火力轰炸而没能炸沉，第二天一看，整个岛成为一片焦土，是否属实无从考证，然而其美景确实实实在在摆在眼前，属于必去的地方，远看如上图。

靠近军舰岛时，海水蓝的让人无法相信这是自然的颜色，怪不得有深蓝的称谓，在这里海比天更蓝。

登上军舰岛，就来到了海水和沙滩的天堂，柔软细白的沙子，湛蓝温暖的海水，仿佛一位热情的少女，轻抚着你的每寸肌肤，啊呸，说偏了……应该是这样，在这样纯净的自然美景中，每个细胞都吸足了高浓度的氧气，脑子会彻底放空，完全的心无杂念，放眼望去，满眼的比基尼，好吧，我不会好好描述热带的海滩，换个话题。

玩沙子玩水几乎是一个小朋友所有的乐趣，陪着小朋友玩着最美的沙子和水，大人也会觉得心旷

神怡。

这里的天气完全看你赶上哪朵云彩了，看到一坨乌云飘过来，那就准备淋浴吧，不用躲，反正很快就飘走了，反正我也在水里泡着。似乎当你与自然这样零距离接触时，你看不到一丝人为的造作时，整个人也会变得顺其自然，自然而然。

一篇游记不能没有对美食的记录，塞班的美食用四个字形容，“包容并蓄”，这和她的历史分不开，来自世界各地的移民将不同地区的美食带到这里，中餐、日餐、西餐，哪一种吃起来都别具风味，都和这个小岛毫无违和感，这里要特别推荐下向日葵餐厅，很赞！

由于去程和返程在飞机上都是晚上，竟然没能从天空中一睹塞班岛的芳容，从地图上看马里亚纳群岛像是太平洋里的一串珍珠，塞班岛是其中最美的一颗，短短几天的行程让我对这个美丽的小岛倍感亲切，五星级的海景，淳朴平和的民风，一波又一波的旅客，你来我往，她就静静地在那里，不急不躁。

下次再见，塞班。d

deepin

we do we change...