ANDREW TANENBAUM, RAJA APPUSWAMY, HERBERT
我，我，我，我，我，我，我，我，
TOMAHRUBY, JORRIT HERDER, ERIK VANDER
KOUWE, AND DAVID VAN MOOLENBROEK

MINIX 3：状态报告和当前研究

尽管事实上，安德鲁·塔南鲍姆
已经为30生产开放源代码
我在自由大学担任了多年的教授，
不知怎么的，他找到了时间去（共同）作者18
书籍和150篇论文，并成为一名研究员
ACM和IEEE的成员。他被授予
2008年获得的USENIX火焰奖。他相信
电脑应该像电视机一样
插上电源，它们就能完美地为
接下来的10年。

ast@cs.vu.nl

拉贾·阿普斯瓦米是Vrije的一名博士生
大学。他的研究兴趣包括
文件和存储系统，以及操作
系统可靠性他收到了来自
印度安娜大学，和他的女士来自
佛罗里达大学，盖恩斯维尔大学。

rappusw@few.vu.nl

赫伯特·博斯从
位于荷兰的特文特大学
以及他在剑桥大学获得的博士学位
计算机实验室（英国）。他目前是
自由大学的副教授
在阿姆斯特丹，有着浓厚的研究兴趣
在操作系统、高速网络中，
和安全。

herbertb@cs.vu.nl

洛伦佐是一名博士后研究员
阿姆斯特丹自由大学
加入教授。塔南鲍姆和他的团队合作-
研究系统的可靠性和安全性。
洛伦佐对系统安全的热情是
进一步受到了加州大学圣巴巴拉分校工作的启发
和石溪大学。洛伦佐收到
计算机科学的硕士学位和博士学位
意大利米兰大学。

l.cavallaro@few.vu.nl

克里斯蒂亚诺·朱夫里达是Vrije的一名博士生
阿姆斯特丹大学。他的研究间
测试包括自修复系统和安全系统
以及可靠的操作系统。他收到了
他和我都来自罗马大学
Vergata，“意大利。

c.giuffrida@few.vu.nl

他拥有阿姆斯特丹自由大学计算机科学硕士学位（以优等成绩毕业），并将于9月在那里获得博士学位。2010.他的研究重点是操作系统的可靠性和安全性，并密切参与了MINIX 3的设计和实现。他现在在

谷歌在悉尼。

jnherder@gmail.com

Toma ˇ s Hrub ‘y拥有布拉格查尔斯大学和自由大学的硕士学位。毕业后，他决定去美国，并在新西兰的奥塔哥大学和澳大利亚的NICTA待了一段时间。他目前是自由大学的一名博士生，正在研究如何将多服务器操作系统与多核芯片相匹配。

thruby@few.vu.nl

埃里克·范德·库维在自由大学获得了硕士学位，现在是自由大学计算机科学专业的博士生。他致力于虚拟化和遗留驱动程序支持。

vdkouwe@cs.vu.nl

大卫·范·穆伦布鲁克拥有阿姆斯特丹自由大学的计算机科学硕士学位，目前在那里攻读博士生。他的研究兴趣包括文件和存储系统以及操作系统的可靠性。

dcvmoole@few.vu.nl

STPEOPLEWANTTHEIRCOMPUTER
他们的电视机一样：你买它，插上电源，它在接下来的10年里非常工作。我只想说，当前的计算机——尤其是它们的操作系统——甚至还不够接近。我们将考虑当普通用户在其一生中从未经历过系统崩溃，并且没有计算机有重置按钮时所完成的工作。在MINIX项目中，我们正在尝试

通过提高可靠性，可用性，和操作系统的安全性。

始于1987年的MINIX 1，一个教学生学习操作系统的工具，现在已经变成了MINIX 3，一个更成熟的操作系统，其内部结构促进了高可用性，同时保留了面向应用程序和用户的成熟的POSIX界面。虽然名称被保留了下来，但这两个系统非常不同，正如Windows 3和Windows 7都被称为Windows，但也非常不同。在本文中，我们将简要描述MINIX 3的体系结构和它现在的样子——作为2007年2月的更新；登录：文章[1]——以及目前正在进一步开发它的工作。

这项工作的动力是授予我们之一（塔南鲍姆）从荷兰皇家艺术与科学学院100万欧元开发一个高度可靠的操作系统，四年后，250万欧元拨款从欧洲研究委员会继续这项工作。这笔资金主要支持博士生、博士后和一些程序员参与这个项目，这导致了一系列的发布，其中3.1.7是最新的一个。

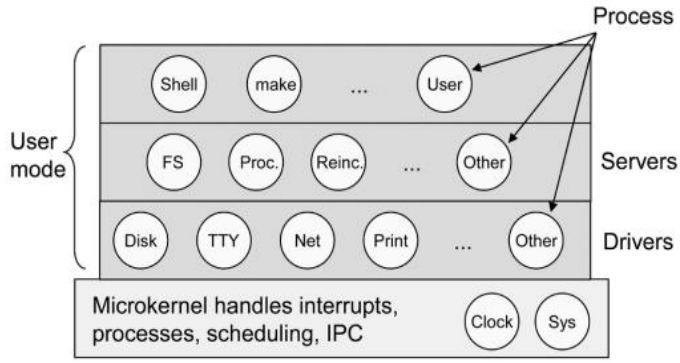
MINIX 3的愿景一直遵循着许多核心原则：

- **关注点的分离：将操作系统分割成彼此之间具有良好隔离性的组件。**
- **最小的权力：只授予每个组成部分它的工作所需要的权力，仅此而已。**
- **容错性：承认错误存在，并计划在继续运行的同时从中恢复。**
- **动态更新：计划一直保持不变，即使是面对重大的软件更新。**
- 符合标准：外部符合posix标准，但不要害怕内部发生变化。

我们相信，我们在生产通用、兼容的操作系统方面取得了良好的开端。随着硬件速度在过去二三十年里的飙升，我们并不认为大多数用户真正关心从硬件中挤出最后一滴性能。对于考试，在MINIX 3中，在现代电脑上构建整个操作系统（大约120个编译和12个链接）只需要10秒。很好。

如果您从www.minix3.org获得MINIX 3.1.7光盘并安装了ROM，对用户来说，它看起来像其他UNIX系统，尽管移植的应用程序较少（到目前为止），因为这不是我们的重点。但在内部层面上，它是完全不同的。它是一个基于小微内核的多服务器操作系统，可以处理中断、低级流程管理和IPC，等等。操作系统的大部分都作为用户模式进程的集合运行。这里的关键概念是“多服务器操作系统”——将系统设计为具有独立故障模式的用户模式驱动程序和服务器的集合。虽然术语“微内核”得到了很多的关注——微内核被广泛应用于手机、航空电子、汽车和其他嵌入式系统，其中可靠性是至关重要的[2]——但我们关注的是系统的多服务器方面。

微内核以内核模式运行，但几乎所有其他操作系统组件都以用户模式运行。用户模式进程的最低层由I/O设备驱动程序组成，每个驱动程序完全隔离在一个由MMU保护的单独进程中，并通过一个简单的API与内核进行通信，并通过消息传递与其他进程进行通信。下一层由服务器组成，包括虚拟文件服务器、MINIX文件服务器、进程管理器、虚拟内存管理器和转世服务器。在这一层的上面是正常的用户进程，如X11、shell和应用程序（参见图1）。



F I G U

转世服务器是该设计中最不寻常的部分。它的工作是监视其他服务器和驱动程序，当它检测到问题时，它会用从磁盘（或者从磁盘驱动程序，从RAM）获取的干净版本替换有故障的组件（驱动程序或服务器）。由于大多数错误都是短暂的，即使在驱动程序崩溃之后（例如，在解引用坏指针后由于分割错误），在许多情况下，即使知道系统的一部分已经被替换，系统的用户进程也可以继续运行。我们运行了一个故障注入测试，其中240万个故障被故意注入到驾驶员中，尽管我们发生了数千起驾驶员崩溃，但系统在所有试验[3]中继续正常运行。它连一次也没撞坏过。

MINIX 3的当前状态

MINIX 3并不是静止不动的。MINIX 3网站已被访问170万次，光盘图像已被下载超过30万次。我们已经被选中参加了2008年、2009年和2010年的谷歌夏季代码活动。它有一个wiki，一个推特提要，一个RSS提要，和一个活跃的谷歌新闻组。

自2007年的论文；登录：，MINIX 3有许多改进，大大小小。以下是对该系统现状的简要总结。

- 兼容posix的操作系统，具有虚拟内存和TCP/IP网络
- 用户界面通常是X11，尽管也有一个简单的GUI（EDE）
- 各种设备驱动程序（例如，千兆以太网、OSS音频框架）
- 支持各种文件系统的虚拟文件系统（如MFS、ISO、HGFS）
- 三个C编译器（ACK、gcc、LLVM），以及C++、Perl、Python、PHP等等
- 各种外壳（bash、pdksh、sh）
- 选择BSD、GNU或V7实用程序（awk、grep、ls、make、sed和所有其他工具）
- 许多包(e。g., Apache，Emacs，幽灵本，m播放器，后gresQL，QEMU，vi)
- 类似raid的软件层，保护完整性，甚至从错误的磁盘驱动程序
- 大量的正确性、一致性、代码覆盖率和性能测试套件

此外，还计划在不久的将来对该系统进行其他更改，包括：

；在：２０１０年6月，在I X 3：状态，在端口和状态

- 移植DDEkit [4]，这将为我们提供许多新的Linux设备驱动程序
- 异步消息传递（这意味着有故障的客户端无法挂起服务器）
- 内核线程

简而言之，虽然这个系统不如Linux或FreeBSD那样完整，但它也不是一个玩具内核。它是一个成熟的UNIX系统，但内部具有完全不同的、高度模块化的、可靠的结构。这也使它成为轻松测试新的操作系统想法的一个很好的研究工具。

目前的研究

我们有各种正在进行的研究领域，都集中于根据上述关于现代硬件的原则，开发一个高度可靠的、模块化的系统。我们还致力于生产一个可用的原型，清楚地表明您可以使用我们的想法构建一个真正的系统。以下是其中五个项目的简要概述。

实时更新

由于它的模块化结构，我们希望能够动态地更新系统的大部分内容，而无需重新启动。我们相信，例如，在系统运行时，可以用较新的版本替换主文件系统模块，而不需要重新启动，也不影响正在运行的进程。虽然Ksplice [5]可以动态地为Linux制作小补丁，但如果不重新启动（因此会停机），它就不能更新到一个全新的版本。

我们的实时更新的起点是，组件的编写者知道它有一天会被更新，并考虑到这一点。特别是新旧组件积极配合，使更新过程顺利进行。几乎所有其他的实时更新工作都假设该更新是突然出现的，并且必须立即完成，无论旧组件所处的状态有多复杂。我们认为这是一种错误的方法，通过将更新延迟几秒钟，我们通常可以使它更容易、更可靠。

实时更新MINIX 3比实时更新单片内核要容易得多，因为每个组件都作为一个单独的进程运行。要更新组件，转世服务器将向组件发送一个更新消息。然后，组件完成当前工作和队列，但不会启动在完成时进入的任何新请求。然后，它小心地将其状态保存在数据存储中，以便新组件以后可以找到它。在新版本启动后，它将转到数据存储以获取保存的状态，根据需要重新格式化和转换它，并开始处理排队的请求。其他组件甚至不应该注意到这次升级——当然，也不是那些正在运行用户程序的组件。

在新版本中，完全有可能有一些数据结构被重新组织。例如，以前作为列表存储的信息现在可能在哈希表中，因此新版本的工作是在运行之前先将存储的数据转换为新的格式。通过这种方式，系统可以运行数月或数年，通过许多重大升级，而不需要重新启动。

碰撞 恢复 的 有状态的 服务器

当前的系统可以处理从无状态驱动程序和服务器的崩溃中恢复，但不能透明地恢复具有大量内部状态的组件，这些状态在崩溃中丢失。我们需要确保在崩溃后可以恢复内部状态。检查指向对于像文件系统这样的组件来说不是一种好方法，它们有大量的状态，每秒钟会变化数千次。相反，我们正在试验一些技术，以实时复制每个进程的内部状态，并校验和它的变化。要做到这一点，我们需要更改编译器来插入代码来做到这一点，这就是为什么我们切换到LLVM，它有钩子（ACK和gcc没有）。

其想法是，在一个有状态的组件崩溃后，一个清除进程进入，并检查现已死亡的组件的核心图像。使用复制的数据和校验和，它可以确定哪些项是有效的，哪些项不是，从而恢复好的项。例如，如果我们能判断inode表中的哪些条目已经损坏（如果有的话），以及哪些是正确的，那么我们就能更好地恢复完全可恢复的文件。在某种程度上，日志记录也可以实现这一目标，但要付出更大的代价。

支持多核芯片

我们正在致力于以一种可扩展的方式支持多核芯片。英特尔已经展示了一个80核的CPU，而一个更大的CPU可能即将问世。我们不认为目前将这些芯片视为多处理器的方法是正确的做法，因为未来的芯片可能没有缓存一致性，或者可能会浪费太多的时间来争夺缓存线和软件锁。相反，我们打算将每个核心视为一个单独的计算机，而不是在它们之间共享内存。实际上，我们的方法是把多核芯片或多或少像一个由以太网连接的独立pc的机架，只是更小。但我们还不确定这将如何实现，所以我们可能会允许一些限制性的共享。我们确实相信（以及大麦鱼[6]的设计者），不跨内核共享内核数据结构将更好地扩展到未来预期的大型芯片。

虽然有些人正在研究如何并行化应用程序，但很少有人在研究如何并行化操作系统。在多服务器MINIX 3的情况下，处理器有许多核心，操作系统由许多进程组成，似乎自然适合将每个进程放在自己的专用核心上，而不是与任何其他进程竞争资源。这些资源包括L1和L2缓存线、TLB条目、CPU分支预测表中的条目等等，这取决于硬件约束。换句话说，当工作开始时，组件都已经设置好并准备就绪，没有过程切换时间。如果内核基本上是免费的，并且多服务器/多核设计将操作系统的速度提高了20%、50%或100%，即使它使用3、5或10核来实现，这是你不会有的收益。拥有大量核心闲置是毫无价值的（除了稍微低的能源成本）。

新文件系统

几乎所有当前的文件系统都是来自1965年的多重文件系统[7]，但是在45年里发生了很多变化。现代磁盘驱动器会出现部分故障，例如没有写入数据块或将其写入与预期的不同位置，但仍可能报告成功。新的设备类别，如ssd和OSD(基于对象的Stor-

；1在：２０１０年6月，在I X 3：状态代表和状态代表，11

年龄设备)正在被引入。这些设备在多个方面不同于磁盘驱动器，具有不同的价格/性能/可靠性权衡。卷管理器和其他工具破坏了“每个磁盘一个文件系统”的绑定，但也使存储管理非常复杂。

当引入RAID时，它看起来像“只是另一个磁盘”，向后兼容现有系统兼容。将RAID放置在存储堆栈的底部，导致了一些可靠性、异构性和灵活性问题。在存在部分故障时，块级RAID算法可能会传播损坏，导致不可恢复的数据丢失。块级卷管理与新的设备访问粒度（如面向字节的flash接口）不兼容。即使是简单的任务，如向现有安装添加新设备，也涉及一系列复杂、容易出错的步骤。

为了解决这些问题，我们正在为一个新的存储堆栈设计。与网络堆栈类似，新的存储堆栈具有具有定义良好的功能的层，即：

- 命名层（处理名称和目录处理）
- 高速缓存层（处理数据高速缓存）
- 逻辑层（提供RAID和卷管理）
- 物理图层（提供特定于设备的布局方案）

这些层之间的接口是一个标准化的文件接口。由于新的堆栈破坏了与传统堆栈的兼容性，因此我们在虚拟文件系统下运行它，因此操作系统可以挂载包含遗留文件系统的磁盘分区和挂载包含新文件系统的分区。因此，新旧程序可以同时运行。

新的存储堆栈解决了上述所有问题。通过在物理层中执行校验和，所有请求都要进行验证，从而提供端到端数据的完整性。通过将具有特定于设备的布局方案隔离到物理层上，就可以跨不同类型的设备使用RAID和卷管理算法。通过提供一个类似于ZFS的存储池的设备管理模型，新的堆栈自动化了设备管理的几个方面。此外，由于逻辑层是对文件感知的，因此可以根据每个文件提供RAID算法。例如，用户可以将关键文件自动复制到他自己的电脑上，在部门或主家庭电脑上，以及在远程云上，但甚至没有将编译器临时文件写入磁盘。

虚拟化

关于虚拟机监视器的最初工作可以追溯到35年前的[8]时代，它专注于生产底层IBM 360硬件的多个副本。现代虚拟化工作基于管理程序，客户操作系统可以多次调用来访问服务并完成工作。实际上，它们更像是微内核，而不是虚拟机监视器。我们相信微内核和虚拟机之间的边界还远远没有确定，并且正在探索虚拟机监控程序到底应该做什么的空间。

我们正在考虑的一个想法是有一个双核。除了用于处理中断、来自驱动程序和服务器的服务请求以及传递消息的普通微内核之外，还有一个以内核模式运行的组件来处理VM退出。然而，这两个部分——微内核和系统管理程序——在不同的地址空间中运行（但都在内核模式中），以避免相互交互。根据这个想法的逻辑结论，我们可能有一个微内核和像虚拟机一样多的虚拟机管理程序

相互保护。希望这种安排能给我们最好的好处，并让我们探索将功能放在不同地方的优势和权衡。此外，我们将尽可能多地将管理程序功能移动到用户模式。

我们也有一些关于如何使用这种技术来重用一些遗留软件的新想法，比如设备驱动程序。

结论

MINIX 3是一个正在进行的研究和开发项目，旨在生产一个具有灵活和模块化结构的高度可靠的开源操作系统。虽然这些资助用于资助博士生、博士后和少数程序员，但就像大多数开源项目一样，我们的大部分工作都依赖于志愿者。如果你想帮忙，请去www.minix3.org查看wiki上的愿望列表，并阅读谷歌MINIX 3新闻组。但即使你没有时间做志愿者，也去拿光盘图片试试。你会感到惊喜的。

确认

我们要感谢本·格拉斯、菲利普·霍伯格、基斯·范·勒维克、阿伦·托马斯、托马斯·维尔曼和德克·沃格特，感谢他们在编程系统的各个部分方面所做的伟大工作。如果没有他们的努力，我们就会有一个纸质的设计，但却不是一个实际的工作系统。他们也对这篇论文进行了反馈。

参考文献

[1] J.N.赫尔德，H。Bos，B.下午。Homburg和。S.塔南鲍姆，“故障弹性操作系统路线图”；登录：，卷。32岁，没有。2月1日。2007，pp.14－20.

[2] G.海瑟，“安全的嵌入式系统需要微内核，”；登录：，卷。30岁，没有。12月6日。2005，pp.9－ 13.

[3] J.N.赫尔德，H。Bos，B.下午。Homburg和。S.塔南鲍姆，“设备驱动器的故障隔离”，第39届可靠系统与网络国际会议论文集，2009年，第3页。33－ 42.

[4] http://wiki.tudos.org/DDE/DDEKit.

[5] J.阿诺德和M。F.开华克，“K拼接：自动无引导内核更新”，2009年ACM SIGOPS EuroSys计算机系统会议论文集，2009年，页。187－198.

[6] A.鲍曼，P。Barham，P.-E.Dagand，T.哈里斯，R。艾萨克斯，S。皮特T.罗斯科。舒普巴赫和A。“多核：可扩展多核系统的新操作系统架构”，ACM SIGOPS第22届操作系统原理研讨会论文集，2009年，第3页。29－ 43.

[7] R.C.戴利的变体和P。G.诺伊曼，“二次存储的通用文件系统”，非洲秋季联合计算机会议论文集，1965年，页。213－229.

[8] R.P.帕梅利，T。I.彼得森，C。C.蒂尔曼和D。J.《虚拟存储与虚拟机概念》，IBM系统杂志，第1卷。11，1972，pp.99－ 130.

；l在：２０１０年6月，在I X 3：状态代表和状态代表，13