申请代码	F0205
接收部门	
收件日期	
接收编号	6207070290

国家自然科学基金

申 请 书

（2020版）

资助类别：面上项目

亚类说明：

附注说明：

项目名称：大规模分布式并行fuzzing关键技术研究

申 请 人：周旭　　电 话：13755023003

依托单位：中国人民解放军国防科技大学

通讯地址：国防科技大学 计算机学院 665室

邮政编码：410073　　单位电话：0731-87000415

电子邮箱：zhouxu@nudt.edu.cn

申报日期：2020年01月30日

国家自然科学基金委员会

基本信息

申请人信息	姓　　名	周旭	性别	男	出生年月	1985年02月	民族	汉族
	学　　位	博士	职称	助理研究员		每年工作时间（月）		8
	是否在站博士后	否			电子邮箱	zhouxu@nudt.edu.cn		
	电　　话	13755023003			国别或地区	中国		
	个人通讯地址	国防科技大学 计算机学院 665室						
	工　作　单　位	中国人民解放军国防科技大学/计算机学院						
	主要研究领域	网络安全						
依托单位信息	名　　称	中国人民解放军国防科技大学						
	联　系　人	张维琦			电子邮箱	kejichu@nudt.edu.cn		
	电　　话	0731-87000415			网站地址	www.nudt.edu.cn		
合作研究单位信息	单位名称							
项目基本信息	项目名称	大规模分布式并行fuzzing关键技术研究						
	英文名称	Research on Key Technologies of Massively Parallel Distributed Fuzzing						
	资助类别	面上项目			亚类说明			
	附注说明							
	申请代码	F0205.网络与系统安全						
	基地类别							
	研究期限	2021年01月01日 -- 2024年12月31日			研究方向：漏洞检测与利用			
	申请直接费用	86.0000万元						
中文关键词		漏洞检测；模糊测试；并行与分布式计算						
英文关键词		Vulnerability detection; Fuzzing; Parallel and Distributing Computing						

科学问题属性

○ **“鼓励探索，突出原创”**：科学问题源于科研人员的灵感和新思想，且具有鲜明的首创性特征，旨在通过自由探索产出从无到有的原创性成果。

○ **“聚焦前沿，独辟蹊径”**：科学问题源于世界科技前沿的热点、难点和新兴领域，且具有鲜明的引领性或开创性特征，旨在通过独辟蹊径取得开拓性成果，引领或拓展科学前沿。

◉ **“需求牵引，突破瓶颈”**：科学问题源于国家重大需求和经济主战场，且具有鲜明的需求导向、问题导向和目标导向特征，旨在通过解决技术瓶颈背后的核心科学问题，促使基础研究成果走向应用。

○ **“共性导向，交叉融通”**：科学问题源于多学科领域交叉的共性难题，具有鲜明的学科交叉特征，旨在通过交叉研究产出重大科学突破，促进分科知识融通发展为完整的知识体系。

请阐明选择该科学问题属性的理由（800字以内）：

需求牵引：软件漏洞对于信息系统安全是一个致命威胁，可以对国民经济乃至国家安全造成重大损失。因此，软件漏洞检测技术是国家经济和国家安全领域的一个重大需求。迄今为止，Fuzzing（也叫模糊测试）是检测软件漏洞最有效的技术手段，然后fuzzing的效率还需要极大的提高才能满足漏洞检测在实效性方面的需求。

突破瓶颈：目前针对fuzzing的研究主要集中在算法优化层面，很难有重大性能突破，导致fuzzing的效率一直是一个瓶颈。为了突破这个瓶颈，本课题转变研究思路，将从另外一个角度去研究这个问题，即如何有效利用大规模并行计算资源来加速fuzzing，从而突破fuzzing的性能瓶颈，推进fuzzing技术在实际漏洞检测中的应用。

综上所述，本课题研究的科学问题属于国家重大需求，影响国家经济和安全，具有鲜明的需求导向、问题导向和目标导向特征，旨在突破fuzzing性能瓶颈这个核心科学问题，研究成果可以直接应用于实际生产中。因此本课题的科学问题属性属于“需求牵引，突破瓶颈”类型。

中文摘要	软件漏洞对于信息系统安全是一个致命威胁，可以对国民经济乃至国家安全造成重大损失。因此，如何快速发现软件漏洞十分关键。迄今为止，Fuzzing（模糊测试）仍然是检测软件漏洞最有效的技术手段。然而，Fuzzing需要消耗大量的计算资源和计算时间，需要极大提升性能以满足漏洞检测实效性方面的需求。目前针对Fuzzing的大多数研究都集中在算法优化的层面，性能提升很难有重大突破。考虑到大规模分布式并行计算技术已经较为成熟，云计算和超算等计算资源也较容易获取，本课题将重点研究Fuzzing的并行优化技术，即如何利用大规模并行计算资源来加速Fuzzing，包括并行fuzzing框架设计，任务调度和负载均衡以及可扩展性优化等，再辅助以算法方面的优化，利用并行和算法的叠加优化效果，大幅提升fuzzing的效率，帮助快速发现目标软件系统中存在的漏洞，缩短漏洞检测时间窗口，最终达到提升信息系统安全性的目的。
英文摘要	Software vulnerability is a great threat to information systems, it can cause great damage to economy and national security. Therefore, it is helpful to discover vulnerabilities in time. Fuzzing is an important technology to detect software vulnerabilities. However, fuzzing usually costs huge computing energy and many computing hours. It requires obvious performance improvement to meets the timeliness of vulnerability detection. Currently, most researches focus on improving the algorithms of fuzzing, which is difficult to make a breakthrough in performance. Considering massively parallel distributing computing is mature (it is easy to get massive computing resources by cloud computing or super-computing), we try to study how to utilizing massive computing resources to improve fuzzing performance. Our study will include distributed fuzzing framework design, fuzzing task scheduling and load balance, scalability analysis. Finally, we plan to combine algorithms methods with our parallel methods to further improve performance. In this way, we can rapidly detect vulnerabilities in target software, shortening the vulnerability discovering window greatly. By the effort of this project, we want to make the information systems and the Internet safer.

项目组主要参与者（注：项目组主要参与者不包括项目申请人）

编号	姓名	出生年月	性别	职 称	学 位	单位名称	电话	电子邮箱	证件号码	每年工作时间（月）
1	卢凯	1973-07-27	男	研究员	博士	中国人民解放军国防科技大学	13407316868	kailu@nudt.edu.cn	4**********X	2
2	王鹏飞	1988-09-07	男	助理研究员	博士	中国人民解放军国防科技大学	15526429185	pfwang@nudt.edu.cn	3**********7	8
3	宋丛溪	1994-11-29	女	博士生	硕士	中国人民解放军国防科技大学	15910917576	526905729@qq.com	1**********1	8
4	张根	1993-12-05	男	博士生	硕士	中国人民解放军国防科技大学	13548979424	zhanggen12@hotmail.com	5**********6	8
5	乐泰	1996-03-27	男	博士生	硕士	中国人民解放军国防科技大学	15664233668	yuetai17@nudt.edu.cn	4**********1	8
6	刘陈一帆	1995-09-28	男	硕士生	学士	中国人民解放军国防科技大学	18074682927	2060909445@qq.com	4**********2	10
7	尹启迪	1997-04-04	男	硕士生	学士	中国人民解放军国防科技大学	13755476638	1063751768@qq.com	3**********0	10
8	刘莹莹	1994-08-08	女	硕士生	学士	中国人民解放军国防科技大学	15273112081	15273112081@163.com	4**********8	8

总人数	高级	中级	初级	博士后	博士生	硕士生
9	1	2			3	3

国家自然科学基金项目资金预算表（定额补助）

项目申请号：6207070290　　项目负责人：周旭　　金额单位：万元

序号	科目名称	金额
	(1)	(2)
1	**项目直接费用合计**	86.0000
2	1、 设备费	0.0000
3	(1)设备购置费	0.00
4	(2)设备试制费	0.00
5	(3)设备升级改造与租赁费	0.00
6	2、 材料费	25.00
7	3、 测试化验加工费	0.00
8	4、 燃料动力费	3.40
9	5、 差旅/会议/国际合作与交流费	25.60
10	6、 出版/文献/信息传播/知识产权事务费	10.00
11	7、 劳务费	16.00
12	8、 专家咨询费	4.48
13	9、 其他支出	1.52

预算说明书（定额补助）

（请按照《国家自然科学基金项目预算表编制说明》等的有关要求，对各项支出的主要用途和测算理由，以及合作研究外拨资金、单价≥10万元的设备费等内容进行必要说明。）

材料费25万元：用于租用4个多核计算节点4年时间，每个节点费用5万元，小计4*5=20万元；项目研制过程中，购置用于现有仪器设备升级改造所需的计算机配件、耗材及抵值易耗品等，小计3万元；因本单位保密要求，在科研场所使用光盘、打印所需纸张、硒鼓等，小计2万元。总计：20万元+3万元+2万元=25万元。

燃料动力费3.4万元：主要用于科研设备运行所需分摊的电费等费用。

差旅/会议/国际合作与交流费25.6万元：项目研究计划每年参加国内调研和学术交流6人次，每人次0.4万元，小计0.4*6*4年=9.6万元；用于组织开展项目交流研讨会、学术研讨会、项目专家咨询会议3次，每次2万元，小计3*2=6万元；计划参加国际学术会议4人次，每人次2.5万元，小计4*2.5=10万元。总计：9.6万元+6万元+10万元=25.6万元

出版/文献/信息传播/知识产权事务费10万元：项目计划发表学术论文10篇，平均每篇论文0.5万元，小计0.5*10=5万元，计划申请发明专利2项，每项1万元，小计2*1万元=2万元。其他包括印刷、会议注册、查新、资料和软件购买、学术会费、通信、专利维护等其他费用，小计3万元。总计：5万元+2万元+3万元=10万元

劳务费16万元：项目计划聘用技术熟练的程序员1人，用于具体的软件编码工作。每名程序员年薪4万元，小计4万元*4年=16万元

专家咨询费4.48万元：用于项目研制过程中的支付给临时聘请专家的咨询、评审等费用，按照目前专家劳务费发放的相关文件标准，以高级专业技术职称为例：会议咨询劳务费税后标准0.24万元/人天、税金0.04万元，本项目计划每年聘请4人次，小计0.28*4*4年=4.48万元。

其他支出1.52万元：用于项目研究过程中必要的其他支出，小计1.52万元。

报告正文

参照以下提纲撰写，要求内容翔实、清晰，层次分明，标题突出。**请勿删除或改动下述提纲标题及括号中的文字。**

（一）立项依据与研究内容（建议 8000 字以下）：

1．项目的立项依据（研究意义、国内外研究现状及发展动态分析，需结合科学研究发展趋势来论述科学意义；或结合国民经济和社会发展中迫切需要解决的关键科技问题来论述其应用前景。附主要参考文献目录）：

计算机软件应用在信息时代人们生活的各个场景中，软件的安全性是软件质量的保障。软件漏洞是指软件中存在的具有逻辑错误的代码片段，这些逻辑错误可以被恶意攻击者所利用，达到控制或损害软件系统的目的。软件漏洞一般是由于开发人员在编写程序时疏忽所致，在软件系统中是普遍存在的[1]。软件漏洞的存在给软件带来了大量的安全问题，比如程序运行时出现非预期崩溃，程序正常的功能难以维持；更严重的是，黑客等恶意攻击者会利用软件漏洞对宿主机系统进行攻击，从而达到窃取用户隐私、破坏系统正常运行的目的。例如，2014 年爆发的“心脏滴血”漏洞是因为 TLS 缺乏对长度的校验，导致缓冲区过读[29]。加拿大税务局（CRA）因为这个漏洞，关闭了电子服务网站。2017 年席卷全世界高校、大型企业内网的勒索病毒 wannacry 就是利用 Windows 系统的 SMB 漏洞获得了系统的最高权限，使得系统中的文件和数据被加密，使用这些文件的用户需要支付高额的赎金才可以解密文件[2]。这些软件漏洞不仅造成了重大的经济损失，也在社会上产生了恶劣的影响。所以，如何快速有效的检测软件漏洞一直都是学术界和工业界紧密关注的一个重要课题。

Fuzzing（也叫模糊测试）是一种通过向目标程序提供非预期的输入并监视异常结果来发现软件漏洞的方法[3]。其中灰盒 fuzzing 技术因为可以较为有效地发现漏洞的特点备受研究人员青睐[7-13,16,22,23,25]，但是它仍然是计算密集型的，测试者需要等待大量的时间才能够完全测试程序。例如，一些难以覆盖的路径需要几天甚至几个月才能找到。因此，提升 fuzzing 的效率至关重要，例如，一个新开发的软件即将发布时，值得花费比平常更多的资源来保证软件的安全性和易用性，让软件及时且高质量地发布。为了提升 fuzzing 效率，现有工作主要从两个维度展开研究：一是通过改进 fuzzing 算法，我们称之为算法优化；二是利用

并行计算资源分解 fuzzing 任务，我们称之为并行优化。现如今主流的提升 fuzzing 效率的方法都是算法层面上的，比如改进变异策略，测试用例选择策略，利用静态分析技术辅助测试等方法。但是较少有研究者利用大规模的计算资源去提升 fuzzing 的效率，达到加速 fuzzing 的目的。

本课题的研究属于对 fuzzing 的并行优化方面。我们认为研究大规模分布式并行 fuzzing 技术具有重要的意义。首先，并行计算是现在应用十分广泛的技术[4][5]，海量廉价的计算资源可以轻松获得（例如，使用 Amazon spot instance[6]）。其次，并行优化具备更大的性能提升空间，在计算资源充足和系统可扩展性好的情况下，理论上并行优化可以无限度的提升性能。最后，并行优化和算法优化是两个正交的研究方向，理论上任何算法上的改进都可以很容易地应用在并行 fuzzing 框架中，比如将灰盒 fuzzing 的测试状态抽象共享给分布式并行系统。通过这种利用更多的计算资源来并行化 fuzzing 任务，而不是局限于改进单个计算节点中的算法的方式，可以使用大量的资源交换时间，以加速软件测试。下面我们将重点介绍国内外关于算法优化和并行优化方面的研究现状。

国内外研究现状一：算法优化方面

目前国内外针对软件 fuzzing 技术的研究主要还是针对算法的提升，涉及到 fuzzing 的多个方面，比如变异策略、种子选择方式、待测程序目标、与其他技术结合等等。首先，AFLFast[7]和 Fairfuzz[22]都是在 AFL 的种子选择策略上做出了改进，两个工作都将目光放在了覆盖到较难到达部分的种子上。AFLFast 的作者 Bohme 在之后又针对另一个方向对 AFL 做出了改进，设计了 AFLGo 工具[8]，该工具属于导向型 fuzzing 工具，可以使测试的方向更倾向于探索指定区域的代码。为了解决灰盒 fuzzing 技术中的 bitmap 中因为哈希值索引碰撞带来的冲突问题，Gan 等人在 2017 年设计了 CollAFL[9]，有效缓解了 AFL 的覆盖率碰撞问题。另外，考虑到待测程序的多样性，产生了一些针对不同应用场景的测试工具，比如 2017 年针对操作系统内核的测试工具 KAFL[10]。2018 年的 PTfuzz[11]是利用 Intel 的 processor trace(PT)记录待测程序的执行情况，将信息反馈给 AFL 进行新的变异，该方法应用于无法获得待测代码源代码，只有二进制文件的情况。与 AFL 对于二进制测试的 Qemu 模式对比，得到了很好的结果。一些测试用例文件具有高度结构化，随机变异难以产生有质量的测试用例通过语法检测，skyfire[23]为了解决这一问题通过统计的方法学习这些文件的词法语法特征并作用于种子生成，之后再将这些种子放入 AFL 的变异机制中。为了使 fuzzing 的覆

盖率进一步提高，一些研究尝试将 fuzzing 与其他技术结合。VUZZER[24]利用了一些静态分析技术分析待测程序，从而能够生成更具有针对性的测试用例。Angora[25]在灰盒 fuzzing 的基础上结合了污点分析以及梯度下降算法，能够较好的解决一些待测程序中难解的约束条件，在 LAVA-M 测试集上取得了比较好的效果。还有一些研究工作将 fuzzing 与符号化执行技术相结合，比如 Driller[26]，在 fuzzing 进程被难解的约束卡住的时候调用符号化执行引擎去尝试解决约束，类似的工作还有 QSYM[28]。为了尽可能避免符号化执行技术带来的时间开销，T-fuzz[27]先将待测程序中的完整性检查去除，在发现潜在崩溃的时候再调用符号化进行约束求解，从而避免多余的开销。

以上的工作都是从算法层面对 fuzzing 技术进行的改进，从而达到了提升 fuzzing 效率的目的。与这些研究不同，本课题重点从另外一个维度展开研究，即研究利用并行计算对 fuzzing 进行改进，从而提升 fuzzing 的效率。对 fuzzing 技术来讲，算法的优化解决了在固定计算能力的前提下，如何最大限度的提升性能；而并行优化的重点解决如何通过扩充计算能力来提升性能。这两种优化方式可以很好的融合，从而产生叠加的优化效果。因此，本课题的研究对算法的研究将是一个很好的补充和加强。

国内外研究现状二：并行优化方面

近年来国内外也有一些研究者尝试通过使用大规模计算资源来提升漏洞挖掘的效率。针对符号化执行技术的路径爆炸问题，亚马逊的 Cloud9[12]将路径搜索范围划分成多个部分，每个计算资源处理其中的一或多个部分。为了保证各个计算资源之间的负载均衡，Cloud9 采用了一定周期重新划分子任务的策略。北京邮电大学的梁洪亮等人[13]提出了一种针对符号化执行应用时路径爆炸的问题，采取路径取反算法以及将随机产生新的测试用例的方法，这些取反得到的结果利用并行资源去运行，将求解约束产生的不同输入分配给不同资源，并基于此设计了“谛听”系统。Xie 等人[14]在 2010 年借助网格计算提升 fuzzing 的效率，用户将待测任务上传到服务器，一个核心调度器将任务分配给多个主机进行测试,利用多个主机的计算资源提升测试效率。但是，该方法的缺点是在测试开始时，就将测试任务静态划分给各个节点，没有考虑到在测试过程中资源分配的问题，所以该工作在测试中会出现任务分配不均衡的现象。以上研究都是对漏洞挖掘做并行优化方面中较早的尝试。然而，这些研究解决的主要问题是分布式系统中的资源调度问题，即如何让多个不同的任务合理的使用分布式系统中的计算资源。在

对一个测试任务做并行优化方面，这些工作一般采用了一种静态任务划分的策略，在负载均衡方面还有很大的改进空间。并且由于研究时间比较早，这些研究采用了一些目前已经不再流行的技术，如网格计算。

AFL 并行模式是灰盒 fuzzing 工具 AFL 自带的并行模式[16]，所有核运行 AFL 得到的状态都储存在一个共享文件夹中。它在单机系统的多个核上运行 AFL 工具实例，通过扫描共享文件夹在各个实例之间共享测试用例队列，从而提升 AFL 的运行效率。该工作可以在多核计算机上运行多个 AFL 实例，加速 fuzzing，但是其缺点在于不能跨多机测试，缺乏可扩展性。Roving [17]和 DisAFL[18]是两个在 AFL 的基础上并行化的工具。它们通过维持服务器客户端架构实现多机并行。每个客户端上部署了 AFL 实例，每经过一定的时间间隔，客户端将测试中产生的新的测试用例和崩溃信息上传至服务器，服务器将这些更新的信息同步给各个客户端，通过这样共享的方式实现灰盒 fuzzing 工具的提速。但是这两个工作的缺点在于没有共享一些必要的 fuzzing 信息，导致产生冗余工作。

PAFL 是清华大学 2018 年的一个并行化 fuzzing 工作[20]，该工作将 AFL、AFLFast、Fairfuzz 工具并行化，每个客户端会共享 fuzzing 信息到全局的共享模块中。同时，共享模块按照 bitmap 分支将测试任务分成多个子任务交给多个客户端中的 fuzzing 引擎进行并行化工作，这样使得每一个客户端专注于自己的一部分子任务，实现了从单客户端低效测试到并行高效测试的迁移，但是该工作不支持分布式并行，且只是针对 AFL 系列的两个工作进行并行，如果更换一个新的优化策略，需要共享的信息不同并行算法就需要重新设计。Enfuzz[21]是 2019 年该组所做的延伸工作，该工作新增的亮点在于分析了不同 fuzzing 工具的差异性和它们擅长处理的待测程序，根据此标准选取了多样性的 fuzzing 工具。在人工总结出多样性标准后，Enfuzz 按照标准选择了几个具有代表性的工作。比如混合符号化执行工具 QSYM、黑盒工具 Radamsa、以基本块覆盖率作为反馈的 Libfuzz，以及 AFL 系列的工具等，将这些工具并行起来，得到一个全面可扩展的 fuzzing 平台。但同样地，Enfuzz 不支持分布式并行，只能使用文件系统共享种子，不能利用海量的计算资源加速 fuzzing 的效率。

谷歌提出了一个大规模的 fuzzing 框架 Clusterfuzz[19]，它利用了谷歌云的海量计算资源，可以支持 AFL 和 LibFuzzer 这两种以覆盖率驱动的 fuzzing 工具以及黑盒测试工具。由于 Clusterfuzz 的庞大计算规模以及测试的持续性，它已经挖掘出 Chrome 中大约 16000 个漏洞，其他 160 个开源程序中大约 11000 个漏洞。

但是 Clusterfuzz 也缺乏必要 fuzzing 信息的同步，而且在各个计算节点上运行的是相同的 fuzzer，缺乏算法的多样性。

总的来讲，现有 fuzzing 并行优化方面的工作存在着以下三个方面的问题：（1）早期的方法大都没有解决 fuzzing 节点之间存在的任务冲突问题，这时，多个计算节点会对同一个种子进行变异、测试，大概率会生成相同的测试用例，造成计算浪费，如 AFL 的并行模式等[16-18]；（2）另一种改进的方法是将种子分给不同的节点去 fuzzing，这样虽然节点之间不会有任务冲突，但是因为种子不能在节点之间调度，很容易造成负载不均衡——有些种子会被过度 fuzz，而有些却存在 fuzzing 不饱和的情况，如 PFuzz 等[30]。这种情况在种子重要程度存在差异和节点数量较多的情况下尤为突出；（3）在分布式环境中，可能存在着计算节点之间计算能力不均衡的情况，相同的调度策略同样容易造成负载不均衡，引起资源浪费，这个问题目前还没有任何工作考虑到。

综上所述，我们认为有必要针对 fuzzing 的并行优化技术开展深入研究，力争解决分布式并行 fuzzing 中的系统设计、任务调度以及负载均衡等关键技术问题，将分布式并行 fuzzing 的可扩展性推进到一个新的高度，从而能够利用海量计算资源来极大提升 fuzzing 的效率；在此基础上，研究并行优化与算法优化的融合技术，通过叠加算法维度上的优化来进一步提升 fuzzing 的效率。

参考文献

[1] 李舟军, 张俊贤, 廖湘科, 等. 软件安全漏洞检测技术[J]. 计算机学报, 2015, 38(4): 717-732.

[2] CVE-2017-0143.https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143. 2015.

[3] Sutton M, Greene A, Amini P. Fuzzing: brute force vulnerability discovery [M].Pearson Education, 2007.

[4] Almasi G S, Gottlieb A. Highly parallel computing [J]. 1988.

[5] Wilson G V. The history of the development of parallel computing [J]. URL:http://ei. cs. vt. edu/history/Parallel. html. 1994.

[6] Amazon spot instance. https://aws.amazon.com/ec2/spot/. 2015.

[7] Böhme M. AFLFast. new. 2017.

[8] Böhme M, Pham V-T, Nguyen M-D, et al. Directed greybox fuzzing [C]. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications

Security. 2017: 2329–2344.

[9] Gan S, Zhang C, Qin X, et al. Collafl: Path sensitive fuzzing [C]. In 2018 IEEE Symposium on Security and Privacy (SP). 2018: 679–696.

[10] Schumilo S, Aschermann C, Gawlik R, et al. kAFL: Hardware-Assisted Feedback Fuzzing for OS Kernels [J]. 2017: 167–182.

[11] Zhang G, Zhou X, Luo Y, et al. PTfuzz: Guided Fuzzing With Processor Trace Feedback [J]. IEEE Access. 2018, 6: 37302–37313.

[12] Amazon spot instance. https://aws.amazon.com/ec2/spot/. 2015.

[13] LIANG H, Xiaoyu Y, Yu D, et al. Parallel smart fuzzing test [J]. Journal of Tsinghua University (Science and Technology). 2015, 54 (1): 14–19.

[14] Yan X. Using grid computing for large scale fuzzing [D]. [S.l.]: Lisbon: Universidade Nova de Lisboa, 2010.

[15] Zalewski M. American fuzzy lop. http://lcamtuf.coredump.cx/afl/. 2015.

[16] Zalewski M. American fuzzy lop. http://lcamtuf.coredump.cx/afl/docs/parallel_fuzzing.txt. 2015.

[17] Roving. https://github.com/richo/roving/.2015.

[18] DistributedFuzzingfor AFL.https://github.com/MartijnB/disfuzz-afl.2015.

[19] Serebryany K. OSS-Fuzz-Google’s continuous fuzzing service for open source software [J]. 2017.

[20] Liang J, Jiang Y, Chen Y, et al. Pafl: extend fuzzing optimizations of single mode to industrial parallel mode [C]. In Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. 2018: 809–814.

[21] Chen Y, Jiang Y, Ma F, et al. Enfuzz: Ensemble fuzzing with seed synchronization among diverse fuzzers [C]. In 28th fUSENIXg Security Symposium (fUSENIXgSecurity 19). 2019: 1967–1983.

[22] Lemieux C, Sen K. Fairfuzz: Targeting rare branches to rapidly increase greybox fuzz testing coverage[J]. arXiv preprint arXiv:1709.07101, 2017.

[23] Wang J, Chen B, Wei L, et al. Skyfire: Data-driven seed generation for fuzzing[C]//2017 IEEE Symposium on Security and Privacy (SP). IEEE, 2017: 579-594.

[24] Rawat S, Jain V, Kumar A, et al. VUzzer: Application-aware Evolutionary Fuzzing[C]//NDSS. 2017, 17: 1-14

[25] Chen P, Chen H. Angora: Efficient fuzzing by principled search[C]//2018 IEEE Symposium on Security and Privacy (SP). IEEE, 2018: 711-725.

[26] Stephens N, Grosen J, Salls C, et al. Driller: Augmenting Fuzzing Through Selective Symbolic Execution[C]//NDSS. 2016, 16(2016): 1-16.

[27] Peng H, Shoshitaishvili Y, Payer M. T-Fuzz: fuzzing by program transformation[C]//2018 IEEE Symposium on Security and Privacy (SP). IEEE, 2018: 697-710.

[28] Yun I, Lee S, Xu M, et al. {QSYM}: A practical concolic execution engine tailored for hybrid fuzzing[C]//27th {USENIX} Security Symposium ({USENIX} Security 18). 2018: 745-761.

[29] Heartbleed http://heartbleed.com

[30] Song C, Zhou X, Yin Q, et al. P-Fuzz: A Parallel Grey-Box Fuzzing Framework[J]. Applied Sciences, 2019, 9(23): 5100.

2. 项目的研究内容、研究目标，以及拟解决的关键科学问题（此部分为重点阐述内容）：

2.1 研究内容

本课题将围绕着分布式并行 fuzzing 系统的设计与优化，开展四个方面的研究内容，分别是（1）分布式并行 fuzzing 系统框架设计和优化研究，（2）分布式并行 fuzzing 的任务调度与负载均衡技术研究，（3）分布式并行 fuzzing 的性能瓶颈及可扩展性分析，（4）分布式并行 fuzzing 中的多样性算法协同优化技术研究。如图 1 所示，这四个研究内容的设置从基础到深入，层层递进，可分为一个框架和三个优化。通过这些内容的逐次研究，本课题最终将产生一个优化的分布式并行 fuzzing 系统，该系统能利用大规模并行计算资源快速的对软件进行安全性测试。

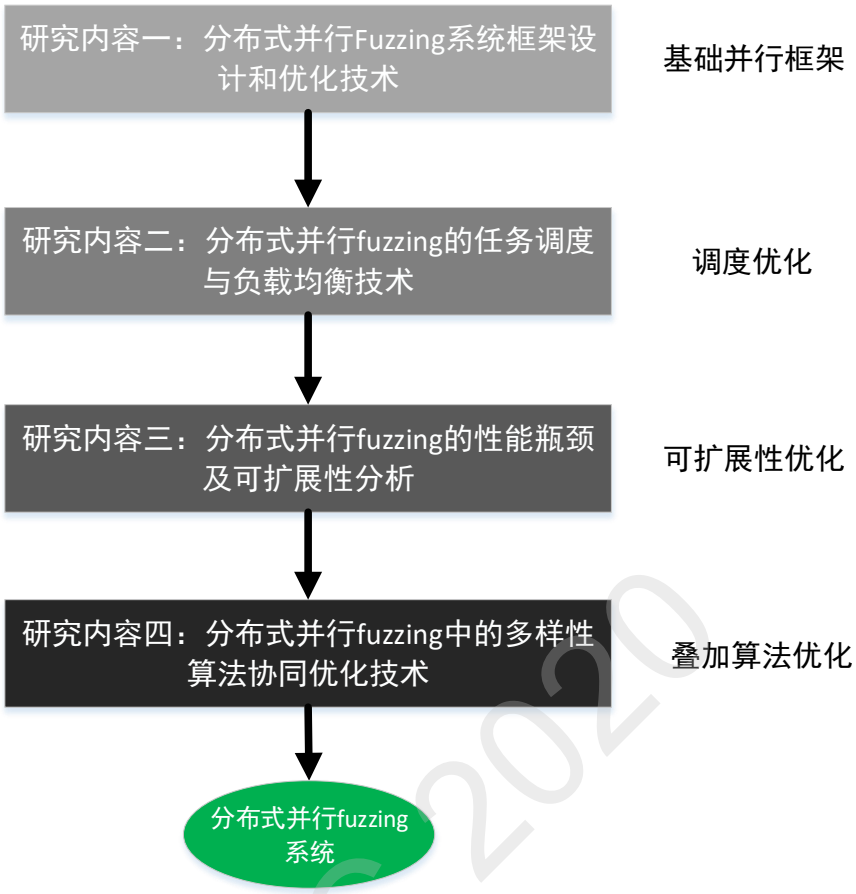

图 1 研究内容设置

一、分布式并行 fuzzing 系统框架设计和优化研究

一个分布式并行 fuzzing 框架需要合理的调度计算资源和分配 fuzzing 任务，以便最大化计算资源的效能。传统的并行 fuzzing 框架在处理调度的时候展现了如下缺点：计算资源使用不均衡，无法充分利用多核和多机的优势；缺乏合理的、动态的任务分配，存在许多冗余计算。本课题将首先针对 fuzzing 任务的特点，研究分布式并行 fuzzing 系统的框架设计与优化，拟采用以数据库为中心的架构，使得系统内的工作节点都可以通过访问一个中心节点获得系统整体的测试状态，从而能够完成最优的任务调度。

二、分布式并行 fuzzing 的任务调度和负载均衡技术研究

现有并行或者分布式 fuzzing 系统多在每个节点中采用了独立的任务调度策略。在缺乏全局 fuzzing 状态信息的情况下，调度是盲目的，容易引起重复计算和负载不均衡的情况。针对这个问题，本课题将研究一种集中式的分布式 fuzzing 调度策略，该策略基于以数据库为中心的系统框架设计，将各个节点本身循环的种子队列和 fuzzing 状态信息整合到服务器中心节点，实现中心调度器，将调度从计算中分离。中心调度器通过综合考虑全局的任务信息、节点计算能力信息等，

实现计算任务与计算能力之间的动态最优匹配，从而解决分布式环境中的fuzzing 任务冲突、负载均衡以及动态节点增删等问题。

三、分布式并行 fuzzing 的性能瓶颈及可扩展性分析

本课题主要研究利用分布式计算资源对 fuzzing 进行并行加速，根据并行系统的设计经验，这种加速一定存在一个可扩展性瓶颈，即在系统规模增加到一定程度时，整个系统的性能增长开始显著变慢，甚至出现系统性能下降的情况。引起可扩展性瓶颈的原因可能是网络通信、数据库、任务之间的数据依赖等多种原因。本课题将研究分布式并行 fuzzing 的性能和可扩展性瓶颈问题，分析测试系统能够扩展到多少个计算节点，分析引起瓶颈的原因，并对系统结构和算法进行针对性优化。

四、分布式并行 fuzzing 中的多样性算法协同优化技术研究

为了提升 fuzzing 的效率，学术界研究了很多不同的 fuzzing 算法，这些算法在不同的软件测试上会有不同的表现，没有任何一个算法能够对所有程序都表现出最优的效果。针对一个 fuzzing 任务，利用算法的多样性可以帮助弥补单个算法的不足和瓶颈，从而提升测试效率。然而，算法和算法之间由于采用了不同的策略和数据结构，很难融合。幸运的是，并行加速和算法的改进是两个正交的技术途径，可以很好的融合。为此，本课题将研究分布式并行 fuzzing 中的多样性算法协同优化技术，课题将设计统一的标准来描述 fuzzing 状态，使得不同的算法可以实现跨平台的测试数据共享。另外，在测试前智能化提取待测程序的信息，在测试到相应程序片段的时候，调用适合的 fuzzing 工具。由此，利用分布式资源协同优化同一个 fuzzing 任务，提升 fuzzing 效率。

2.2 研究目标

课题针对大规模分布式环境下的并行 fuzzing 技术展开研究，通过研究分布式并行 fuzzing 系统框架，分布式环境中的 fuzzing 任务调度和负载均衡策略，分布式并行 fuzzing 在大规模并行条件下的性能和可扩展性瓶颈分析，以及多样性 fuzzing 算法的协同优化等关键技术和问题，从而能够利用大规模分布式计算资源对传统 fuzzing 进行加速优化，极大提升 fuzzing 的效率，帮助快速发现目标软件系统中存在的漏洞，缩短漏洞挖掘时间窗口，最终达到提升软件系统安全性的目的。

2.3 拟解决的关键科学问题

一、分布式环境中 fuzzing 状态的快速同步问题

利用分布式并行资源进行 fuzzing 时，需要系统内的计算资源之间共享一些必要的 fuzzing 状态，以达到整个系统的协同作业。这里涉及到两个问题：一、哪些 fuzzing 状态需要被同步？二、最佳的同步时机是什么？通过分析一般的 fuzzing 过程，我们可以得出一些基本的 fuzzing 状态信息：如测试用例、种子、种子的状态信息、覆盖率等。这些信息哪些需要同步以便共享，哪些不需要？同步信息过多会造成性能开销，同步信息过少会造成信息缺失，不利于任务调度。同时，同步时机也是一个关键问题，同步太频繁会引起性能开销，同步不及时会影响调度器的决策。本课题需要综合考虑 fuzzing 算法，系统结构，节点性能等因素，解决分布式环境下 fuzzing 状态快速同步这个关键问题。

二、集中式的 fuzzing 任务调度和负载均衡机制

任务冲突（会造成重复计算）和负载不均衡是目前多数并行 fuzzing 系统的共同问题。本课题拟通过集中式的 fuzzing 任务调度策略来解决这个问题。然而，分布式环境还引入了额外的复杂性：一、不同节点的计算能力存在差异性；二、系统中的节点可能随时会死掉或者引入新的节点。这就对系统的设计提出了更高的要求。首先，中心调度器需要综合考虑 fuzzing 任务的差异性，计算节点的差异性，通过合理的负载均衡机制，将任务划分到不同的计算节点。其次，中心调度器需要能够在系统节点发生变化时及时响应，动态的调整任务划分。任务调度机制和负载均衡是分布式并行 fuzzing 的一个关键技术问题。

三、什么是分布式并行 fuzzing 的性能和可扩展性瓶颈

在现有的并行框架下，分布式并行 fuzzing 必然存在性能和可扩展性瓶颈。那么什么是分布式并行 fuzzing 的性能和可扩展性瓶颈就是一个关键的科学问题，是网络通信、数据库还是任务间的数据依赖？为了回答这个问题，我们需要对系统进行可扩展性测试，找到扩展性拐点，通过系统分析找到引起瓶颈的关键因素，通过针对性的优化突破瓶颈，进一步提升系统性能。

3. 拟采取的研究方案及可行性分析（包括研究方法、技术路线、实验手段、关键技术等说明）；

本课题将采用层层递进的研究方法，依次对各个内容展开研究。首先，本课题将针对分布式并行 fuzzing 设计合理的系统软件框架，根据对已有研究 fuzzing 工作的分析和我们的研究经验，本课题拟采用以数据库为中心的分布式框架，初步解决 fuzzing 技术的并行化问题；在此基础之上，本课题将重点开展分布式并

行 fuzzing 的任务调度和负载均衡机制研究，解决其中的关键技术问题：即在分布式环境中由于共享信息缺失、不一致导致的重复测试问题，从而确保分布式环境中的计算资源能够被 fuzzing 任务有效利用；通过以上两个内容的研究，我们预期将会获得一个较为优化的分布式并行 fuzzing 系统。然而，并行系统都会存在一个可扩展性瓶颈的问题。因此接下来，本课题将重点去分析分布式并行 fuzzing 系统的性能和扩展性瓶颈，通过不断增加计算资源来对系统的可扩展性进行压力测试，分析可扩展性瓶颈原因，研究可扩展性瓶颈的突破方法；最后，本课题将研究一个非常有意思的问题，即算法多样性对分布式并行 fuzzing 的优化问题。算法多样性已经被证明有助于提升 fuzzing 效率，在分布式环境中，由于存在大量并行的 fuzzing 节点，实际上更有利于实现多样性的 fuzzing 算法，因此这个研究内容将会成为 fuzzing 并行优化的一个强有力的补充。这四个研究内容层层递进，难度逐渐增加，具备研究的可行性。下面将针对每个研究内容详细阐述我们拟采用的研究方法和技术路线，以及如何突破关键技术的思路。

一、分布式并行 fuzzing 系统框架设计优化方案及可行性分析

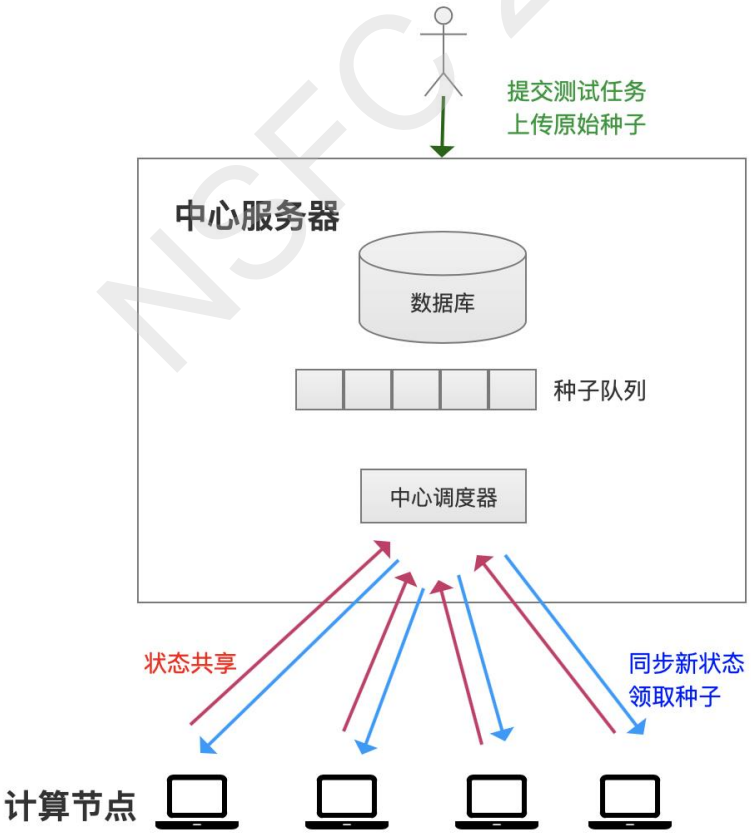

图 2 分布式并行 fuzzing 系统框架

为了均衡地利用分布式并行计算资源发挥多核和多机的优势，合理的、动态

地分配任务，并且避免进行冗余的 fuzzing 工作，本课题将设计以数据库为中心的分布式并行 fuzzing 框架，以集中式的架构存储 fuzzing 的必要状态，并且以集中式的调度策略来分配任务。框架的整体设计如图 2 所示。

（1）系统框架

整个系统框架分为两大部分，第一部分是计算节点。计算节点由大规模集群中的计算资源构成，每个节点代表一个计算核，每个核作为框架中进行运算的一个基本单位可运行一个 fuzzing 进程。第二部分是中心服务器，中心服务器是整个框架中的核心部分，其中包括了一个中心数据库用来存储所有的 fuzzing 状态，中心调度器执行集中式调度策略，负责将测试任务合理地调度给各个计算节点，种子的实例存储在数据库中，但中心调度器将所有种子维护成一个全局种子队列方便动态调度。计算节点通过访问中心服务器，通知中心调度器为计算节点分配测试任务。另外，计算节点获取系统的测试状态并同步更新本机的状态至中心服务器，避免冗余操作。服务器的数据库保存了 fuzzing 整体状态，维护了系统框架的整体性。系统在触发崩溃之后，利用自动漏洞检测技术，实现漏洞的验证。系统框架通过统一的 web 接口与用户交互，用户上传测试任务和初始种子到服务器中心节点的数据库中，中心调度器即可启动调度。所以用户可以通过 web 接口实时与中心服务器交互获取测试进程和结果。

在明确了框架的各个部分和功能后，我们需要解决两个问题：一是哪些 fuzzing 状态需要被同步？二是同步的时机是什么？

（2）需要同步的状态

我们分析了一般的灰盒 fuzzing 过程，基本的 fuzzing 状态信息包括测试用例、种子、种子的状态信息、覆盖率等。同步全部测试信息将会避免冗余测试，但会带来不必要的开销，因此，我们评估了各个测试状态对 fuzzing 过程的作用，并选择必要的状态进行同步。第一，因为 fuzzing 的过程是由种子驱动，通过不断变异种子产生测试用例输入给待测程序，所以种子是需要被同步的。第二，各个节点在 fuzzing 的变异过程中会产生新的测试用例，这些测试用例会产生新的覆盖率，可以驱使 fuzzing 探索到更多地方。有些节点产生了过多的测试用例，这些测试用例也应当被上传到中心服务器中从而共享给其他计算节点。第三，以灰盒 fuzzing 技术 AFL 为例，AFL 采用了一种遗传算法渐进式探索策略，为了将该特性从单机迁移到分布式并行环境中，需要同步种子的一些附加信息，比如标志种子是否值得变异的 favored 字段等，有了这些信息，中心调度器就可以跨越

多机之间的屏障，利用 AFL 遗传算法的优势去调度选择种子。第四，覆盖率信息是以覆盖率为反馈 fuzzing 工具的反馈信息，是该类 fuzzing 机制的核心状态，存储在 bitmap 数据结构中。但是，bitmap 占用空间大，对比复杂度高，所以本研究在初步实验过后将不存储覆盖率信息在中心服务器中，采用另一种“复现”的方式去同步覆盖率信息，这样一来节约了时间、空间资源，也达到了同步覆盖率的目的。

（3）同步时机

同步周期既不能太频繁会增加开销，也不能太稀少会影响调度器的决策，这就需要确定一个合理的同步时机。本课题设定在各个计算节点发现新种子的时候立即上传到中心服务器中，将信息在系统中共享。但为了保证同步时机不能过于频繁影响网络通信的效率，将各节点从中心服务器同步新的种子的时间设定在每一次 cull_queue()之前。这样的时间周期设定既保持了中心服务器保存了最完整的系统状态，又不会太多占用通信资源，保证了同步的及时性。

二、分布式并行 fuzzing 的任务调度和负载均衡技术研究方案及可行性分析

Fuzzing 的任务调度冲突和负载均衡是目前并行优化研究的共同问题，前人将 fuzzing 迁移至多核环境中时，没有修改原先的调度策略使得节点之间存在着任务冲突，首先会有多个节点对同一个种子进行变异、测试的情况，这种情况大概率会生成相同的测试用例。另外，即使将种子分给不同的节点去 fuzzing，虽然节点之间不会有任务冲突，但是因为种子不能在节点之间调度，很容易造成负载不均衡——有些种子会被过度测试，而有些却测试不饱和，这种情况在种子重要程度存在差异和节点数量较多的情况下尤为突出。以上两种情况造成重复计算，浪费计算资源；另外，节点之间因为机器性能等因素制约造成了计算资源存在差异性，系统中的节点可能随时会死掉或者引入新的节点也是需要被系统考虑的情况。比如前人的工作采用静态的任务调度方法，将测试任务在开始就划分给各个节点，没有考虑节点在 fuzzing 过程中的动态变化，这就是一种系统负载不均衡的情况，这样的调度方式将会造成计算资源的严重浪费。

（1）集中式调度策略

为了进行合理的任务调度，消除冲突，并均衡系统负载，本课题拟研究一种集中式调度器，部署在中心服务器上。集中式调度器采用集中式的 fuzzing 调度策略，对 fuzzing 任务进行动态调度。在这种策略下，由调度器统一分配测试任务，计算节点仅仅完成分配到测试任务。对于计算节点所需的共享 fuzzing 状态

信息，采用“复现”、“同步”两种方式同步给各个节点：

“复现”方式指将从数据库下载的种子使用恢复模式，dry_run 一遍（初次运行某程序时只 dry_run 初始测试用例），这样一来，能够使当前节点快速更新本地的一些 fuzzing 状态，减少了“同步”时的开销。对于存储占用空间大、对比复杂度高的覆盖率信息，本课题采用“复现”的方法。

“同步”方法通过从中心服务器下载新的种子，以及其他 fuzzing 状态信息，将那些无法通过“复现”得到的状态同步给各个节点，

通过以上的“复现”、“同步”方式进行状态共享，以及集中式调度器对整个系统的 fuzzing 进程进行引导，fuzzing 相同种子的情况不会出现。调度器将种子以及 fuzzing 必要状态共享给各个节点，就可以保持一些已经没有 fuzzing 价值的种子不会被过度测试，避免了 fuzzing 不饱和的现象，从而规避了大量的冗余工作。

（2）负载均衡

集中式 fuzzing 调度策略保证了任务调度的动态性，当一个节点完成一个种子的 fuzzing，需要更换种子时，中心调度器将按照维持的全局种子队列调度种子给节点。如果出现了计算节点间性能有性能差异的情况，这种抢占式的调度策略可以保证高性能的节点在快速完成任务后能够及时领取到新的任务，不会出现空闲状态，浪费计算资源，动态性的调度在各个节点的性能不均衡的时候体现出较大的优势，性能好的节点可以处理更多任务。当系统中退出节点或加入新节点时，也不影响系统整体的 fuzzing 进程，因为 fuzzing 状态都维持在中心服务器中，集中式调度器将调度任务给系统中活跃的节点。这种调度方法可以大规模环境中有效防止了资源浪费，保持了负载均衡，提升系统的测试效率。

三、分布式并行 fuzzing 的性能瓶颈及可扩展性分析研究方案和可行性分析

在实现了分布式并行 fuzzing 框架后，我们会对框架进行可扩展性测试，找到扩展性拐点，从而发现其中一些潜在的性能瓶颈，为日后进行大规模分布式并行 fuzzing 提供保障。我们将选择在超算中心的大规模计算节点上进行扩展性试验（目前我们已经在湖南超算部署了 8 台多核服务器，共计 256 个计算核心用于 fuzzing 实验，未来还会继续扩展），实验内容将围绕着三种潜在的性能瓶颈进行，分别是：网络通信、数据库、数据依赖。

（1）网络通信

本课题拟衡量的第一个潜在扩展性瓶颈是网络通信问题。在系统利用大规模节点后，节点数目的增多会占用大量通信带宽，系统的通信处理速度可能会受到

制约。为了探测这一扩展性瓶颈，我们设计了以下实验：

我们将节点按照一定数目分组，隔一定时间开启一组新的节点加入系统进行fuzzing，并保持对网络吞吐量的追踪，记录数据。最后将得到的数据进行分析，得出加入多少节点时出现扩展性拐点。

（2）数据库

因为关系型数据库诸如 Mysql 等严格执行数据库的 ACID 约束限定，因此普遍存在海量并发数据处理效率相对低下的问题。本课题的中心数据库拟采用非关系型数据库 MongoDB，在适量级内存的 MongoDB 的性能是非常迅速的，它将热数据存储在物理内存中，使得热数据的读写变得十分快，但是也存在占用内存大等劣势。为了探究数据库的性能瓶颈，我们设计了以下实验：

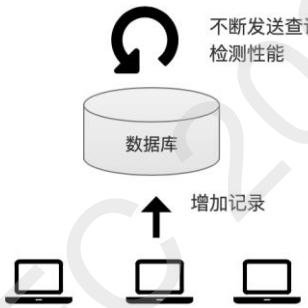

图 3 数据库性能瓶颈实验

实验的基本框架和操作流程如上图 3 所示，

我们设计采用 128 个节点进行数据库的扩展性实验，在每个节点上运行 mongodb-performance-test 工具，模拟 fuzzing 中的数据读写操作，使数据库中快速增长大量记录。同时，在服务器端也运行 mongodb-performance-test 工具，不断进行查询操作。在实验中需要检测的指标有：服务器的内存、mongodb 的处理速度。我们通过观察这两个指标分析 mongodb 的扩展性拐点。

（3）数据依赖

在分布式并行环境中，计算节点间对中心服务器中的共享资源进行不当竞争可能会导致一些错误。通过初步实验和分析，我们得出系统可能存在以下数据依赖：

（a） 数据库中存在重复种子

（b） 多个节点同时更新全局状态，导致状态丢失。

为了探究是否存在以上依赖以及依赖程度，我们设计了以下实验：

对服务器的使用 64 个节点进行 48 小时的实验，每 30 分钟，统计数据库中的相同种子数目，判断是否存在相同种子这种情况如果有则分析增长趋势。对于多个节点同时更新全局状态，根据每个状态的特性，追踪是否存在状态丢失的情况。如果存在以上数据依赖情况，那么我们将对调度策略进一步改进优化，加入相同种子筛选以及同步锁等机制避免数据冲突，进一步提升系统的效率。

四、分布式并行 fuzzing 中的多样性算法协同优化技术研究

学术界对 Fuzzing 算法的研究具有多样性，因为这些研究针对不同的优化方向，关注了不同的应用场景或基于不同的技术，产生了具有多样性的 fuzzing 算法。例如，FairFuzz 能够针对稀少分支提高进行高效探索；AFLFast 能够在更短时间内达到较高覆盖率；Matryoshka 则主要致力于高效生成通过嵌套语句的测试用例。如果能够利用分布式并行平台的条件，协同各类 fuzzing 算法在深层次路径探索、能量节约、嵌套分支处理等不同方面的优势，测试效果将得到大幅度提升。为了实现这一目的，本课题拟开展 fuzzing 算法多样性的选择、fuzzing 状态的统一标准、fuzzing 算法的智能调度等研究。

（1） fuzzing 算法的特点研究

首先我们需要确定不同的 fuzzing 技术适合解决的程序类型及应用场景。为此，我们结合相关论文及技术报告，对现有的 fuzzing 技术进行整理、收集、归类，并结合静态分析技术，通过大量的程序测试，收集不同 fuzzing 技术对不同类型程序在分支覆盖率、代码覆盖率、漏洞检测数量等指标上的实验数据，进而通过人工推理分析得出不同 fuzzing 工具适合解决的程序类型与应用场景。在获取这一知识后，我们将对现有的 fuzzing 工具按照应用场景与适应类型进行筛选评估，从重遴选出在各个方面具有很高代表性的 fuzzing 工具，作为多样性算法协同框架的基础。根据筛选，初步拟定选择的工具有覆盖率反馈的灰盒模糊测试工具 AFL、混合符号化执行工具 QSYM、黑盒模糊测试工具 Radmsa 等，作为工具集部署在大规模分布式并行 fuzzing 系统中。

（2） fuzzing 状态的统一标准

因为不同的工具分属于不同的技术类别，采用了不同的策略、数据结构、覆盖率指标和反馈信息，对目标程序的类别、有无源码等需求也不尽相同，所以会产生具有差异的 fuzzing 状态信息，这些信息由于标准不一致，难以融合。为了实现进一步地协同多样性 fuzzing 技术进行加速，本课题将设定统一的标准，对选取的 fuzzing 算法的状态信息抽象出同一个模板，作为全局数据存储在数据库

中，不同的 fuzzing 算法在本地维持自己的数据结构，在同步和共享的时候采用全局的模板，将不同 fuzzing 算法的数据结构转化成统一的数据结构，这一过程由中心调度器统一调度转化，使得不同的算法可以实现跨平台的测试数据共享。

（3） fuzzing 的智能调度

由于所采用的 fuzzing 算法具备多样性，使得它们在对不同的软件测试条件下，在不同指标上的效果均有所差异。正如 Lv 等人[MOPT]指出，不同的变异策略对不同的应用程序有不同的效果，通过利用离子群加速算法等智能算法对现有策略进行实时评估，进而侧重于使用评估效果好的策略，更有利于提升测试效果。为此，我们将通过静态分析以及一些智能化手段提取待测程序的特征信息，并设计 fuzzing 节点的初始匹配与调度算法，在测试开始前根据提取的信息分配带有相应 fuzzing 算法的节点，并合理配置节点数量。在测试过程中，中心调度器还可以根据反馈信息，对不同 fuzzing 节点的效率进行评估，并结合静态分析，同时对全局 fuzzing 的进度与下一步目标进行描述，根据现有策略的效率及全局测试进度，及时调整 fuzzing 算法的选择策略。由此，利用分布式资源协同优化同一个 fuzzing 任务，提升 fuzzing 效率。

4. 本项目的特色与创新之处；

本项目的特色和创新之处包括以下几点：

一、利用大规模分布式计算资源大幅提升 fuzzing 的效率。我国现如今超算技术发展迅速，海量计算资源可以与漏洞挖掘这一计算密集型任务结合，能够对测试过程提供强大增益。分布式并行 fuzzing 具有很强的可扩展性能力，可以支持超算中心中的大规模计算资源同时测试。这一扩展应用可以持续性的测试软件，为大量软件上线工作、漏洞检测工作提供有力保障，具有广阔的应用前景。

二、分布式并行资源对 fuzzing 提速和传统从算法层面提升 fuzzing 效率是两个正交的优化方向，系统可以结合多样性算法设计协同优化策略，从而叠加 fuzzing 的优化效果。

三、现如今的并行 fuzzing 工作的任务调度策略存在着大量的冗余工作，而且节点之间的性能差异引起任务分配不均衡，造成计算资源的浪费。本系统采用中心化调度，并结合“复现&同步”的信息共享方式，可以大幅减少冗余计算、实现负载均衡。

5. 年度研究计划及预期研究结果（包括拟组织的重要学术交流活动、国际合作与交流计划等）。

年度研究计划

2021 年 1 月-2021 年 12 月

- 全面调研现有并行 fuzzing 工作，深入总结、分析和对比
- 设计以数据库为核心的分布式并行 fuzzing 系统框架
- 基于 AFL 实现分布式并行 fuzzing 系统，并做测试和初步优化
- 组织国内外专家进行分布式并行 fuzzing 的学术讨论

2022 年 1 月-2022 年 12 月

- 研究集中式的任务调度策略，解决分布式并行 fuzzing 中的负载不均衡和重复任务问题
- 对集中式任务调度策略进行测试、对比和分析
- 根据实验结果对集中式任务调度策略进行改进和优化
- 组织学生到国外知名大学联合培养

2023 年 1 月-2023 年 12 月

- 对分布式并行 fuzzing 系统进行可扩展性测试（到上千并行度的规模）、分析性能和扩展性瓶颈
- 研究突破扩展性瓶颈的方法，进一步优化
- 参加高水平国际学术交流会

2024 年 1 月-2024 年 12 月

- 研究并行优化与算法优化的融合技术
- 研究利用算法的多样性对分布式并行 fuzzing 进行叠加优化
- 整理研究成果，撰写研究报告

预期研究成果

发表高水平学术论文 6 篇（SCI 论文或者 CCF B 类会议以上论文）。实现分布式并行 fuzzing 软件系统一套。培养博士生 3 人，硕士生 5 人。

（二）研究基础与工作条件

1. 研究基础（与本项目相关的研究工作积累和已取得的研究工

作成绩）；

本课题组隶属于国防科技大学计算机学院系统与安全实验室，课题组成员长期从事操作系统、并行计算和信息安全方面的研究工作。

在并行计算方面，课题组成员多来自于天河超级计算机研究团队，在系统软件、并行计算和大规模分布式计算等方面都拥有深厚的技术底蕴。课题组前期的研究包括分布式并行计算技术，虚拟化技术和容器技术等方面，相关研究成果发表在 PPoPP、VEE 等顶级国际学术会议上，并已在天河超级计算机中得到应用，课题组在相关研究上累计获得 2 项自然科学基金项目资助。

在信息安全方面，课题组长期从事软件漏洞自动挖掘方面的工作,已有 12 年的技术积累，累计发现超过 100 个 0-day 漏洞,在 CCF A 类学术会议上发表论文多篇。课题组自主独立研发了基于符号化执行的软件漏洞发掘系统，先后自动发掘典型各类桌面应用软件漏洞 27 处。其中，为了证实漏洞的真实存在性，上报微软 Windows 2000/2003 Server 系统中的 2 处安全漏洞，均被微软官方承认并列入最高级别高危漏洞: CVE-2009-1923、CVE-2009-1924。另外，系统自动发现百度公司 Baidu HI 即时通讯软件多处漏洞，其中，报告的一处高危缓冲区溢出漏洞被国际 CVE 公共漏洞库收录，编号 CVE-2008-6444。

2017 年，课题组从 Linux 内核中发掘 9 个未知 double fetch 漏洞（CVE-2016-5728，CVE-2016-6130，CVE-2016-6136，CVE-2016-6156，CVE-2016-6480，CVE-2017-8831，CVE-2017-9984，CVE-2017-9985，CVE-2017-9986），有的漏洞已潜伏 10 年以上。所撰写的论文“How Double-Fetch Situations turn into Double-Fetch Vulnerabilities: A Study of Double Fetches in the Linux Kernel”**发表在安全领域顶级会议 USENIX Security 2017 上，引起了领域内对 double fetch 漏洞的关注**，并迅速得到其他顶级安全团队的跟进研究和引用。其他团队的跟进成果分别发表在安全领域顶级会议 CCS’18 和 S&P’18 上，进一步提高了此问题的关注度。**该项研究在 2019 年获得自然科学基金资助**。

自 2019 年开始，课题组探索了利用分布式并行计算资源加速 fuzzing 的技术，利用该技术发掘 0-day 漏洞 30 多个，目前已上报其中 13 个并被 CVE 漏洞库所收录，发表相关学术论文一篇。课题组 2019 年的一篇关于 fuzzing 算法优化的文章“EcoFuzz: Adaptive Energy-Saving Greybox Fuzzing as a Variant of the Adversarial Multi-Armed Bandit”最近刚刚**被安全类顶级会议 USENIX Security 2020 所接收**。

2．工作条件（包括已具备的实验条件，尚缺少的实验条件和拟解决的途径，包括利用国家实验室、国家重点实验室和部门重点实验室等研究基地的计划与落实情况）；

课题组隶属于国防科技大学计算机学院，依托软件安全智能并行分析湖南省重点实验室，与国家超算中心长沙中心（湖南超算）建立了密切的合作关系。目前，课题组已配备了各种档次的计算机设备（如服务器、高档工作站和微机等）和相关软件，并于 2019 年 12 月在湖南超算采购了 20 个多核并行计算节点，已累计投入使用其中的 8 个计算节点，共计 256 个计算核心，用于分布式并行 fuzzing 的技术研究。2020 年，课题组计划继续在湖南超算扩充 16 个多核计算节点，用于分布式并行 fuzzing 的可扩展性研究。目前，这些软硬件设备均处于良好运行状态，它们为本项课题研究工作的顺利开展提供了良好的实验环境。学院科研学术氛围浓厚，与国内外许多知名大学、研究机构和企业建立了良好的学术交流和合作关系。这些条件使得我们开展该项目的研究具有了很好的研究和技术保障。

3．正在承担的与本项目相关的科研项目情况（申请人和项目组主要参与者正在承担的与本项目相关的科研项目情况，包括国家自然科学基金的项目和国家其他科技计划项目，要注明项目的名称和编号、经费来源、起止年月、与本项目的关系及负责的内容等）；

国家重点研发子课题：固件 fuzzing 技术；编号：2019QY0502；经费来源：科技部；研究周期：2019 年 7 月-2020 年 6 月；和本课题的关系：该课题主要是 fuzzing 算法层面上的研究，是针对特定目标（固件）的 fuzzing 技术，而本课题主要是 fuzzing 并行优化方面的研究，与该课题的研究是同一个技术（fuzzing 技术）的两个正交的研究方向，该课题可以利用本课题的研究成果进行性能优化，本课题可以利用该课题的研究成果拓展应用范围；申请人是该子课题的项目负责人。

4．完成国家自然科学基金项目情况（对申请人负责的前一个已结题科学基金项目（项目名称及批准号）完成情况、后续研究进展及与本申请项目的关系加以详细说明。另附该已结题项目研究工作总结

摘要（限 500 字）和相关成果的详细目录）。

申请人已完成自然科学基金青年基金项目一项：项目名称为确定性并行技术研究，批准号：61402492。该项目于 2017 年 12 月完成结题工作。

该项目主要研究在分布式并行计算中，并行程序的竞争执行所引起的不确定性问题，通过控制竞争消除不确定性。该项目的研究属于分布式并行计算领域，为分布式并行计算领域的基础性研究；而本项目的研究属于安全领域，与该项目在分布式并行计算这个技术点上有交叉。该项目为本项目的研究提供了一定的技术基础。

确定性并行技术研究工作摘要：

课题在已有工作的基础之上，进一步提出了面向多核并行程序的确定性懒惰释放一致性实现方法和一种快速的全系统模拟器确定性回放方法，对于解决确定性运行时的时空开销问题以及兼容性问题起到了关键性的作用。课题围绕着确定性运行时技术在程序分析、硬件支持技术和编程模型等方面进一步展开研究，在静态数据竞争检测的误报剔除技术，基于程序切片的并行错误检测技术、基于程序例化的面向强确定性执行的程序分析技术，无全局同步的弱确定性执行技术以及基于硬件虚拟化的进程级确定性执行环境等方面也取得了进展和突破。通过这些技术的研究，在一定程度上推动了确定性运行时系统的实用化，完成了课题制定的研究目标。

确定性并行技术研究相关成果：

会议和期刊论文：

[1] 周旭，卢凯，陈沉，确定性并行技术，计算机学报，2015，38(5)：973~986，EI

[2] Chen Chen，Kai Lu，Xiaoping Wang，Xu Zhou，Zhendong Wu，DRDet: Efficiently Making Data Races Deterministic， IEICE TRANSACTIONS on Information and Systems，2014，97(10)：2676~2684

[3] Chen Chen，Lu Kai，Wang Xiaoping，Wu, Zhendong，Zhou Xu A Load-Balanced Deterministic Runtime for Pipeline Parallelism， IEICE TRANSACTIONS ON INFORMATION AND SYSTEMS，2015.2，E98D(2)：433~436，SCIE

[4] Kai Lu，Xu Zhou，Tom Bergan, Xiaoping Wang. An Efficient and Flexible Deterministic Framework for Multithreaded Programs，Journal of Computer Science and Technology，2015.1.1，30(1)：42~56，SCIE

[5] Wu Zhendong，Lu Kai，Wang Xiaoping，Surveying concurrency bug detectors based on types of detected bugs，SCIENCE CHINA-INFORMATION SCIENCES，2017.3，60(3)，SCIE

[6] Zhang Wenzhe，Lu Kai，Lujan Mikel，Wang Xiaoping，Zhou Xu，Write-Combined Logging: An Optimized Logging for Consistency in NVRAM， SCIENTIFIC PROGRAMMING，2015，12(2)，SCIE

[7] Zhang Wenzhe，Lu Kai，Lujan Mikel，Wang Xiao-ping，Zhou Xu，Fine-grained checkpoint based on non-volatile memory， FRONTIERS OF INFORMATION TECHNOLOGY & ELECTRONIC ENGINEERING，2017.2，18(2)：220~234，SCIE

[8] Zhu Guoliang，Lu Kai，Wang Xiaoping，Zhang Yiming，Zhang Pengfei，Mittal Sparsh，SwapX: An NVM-Based Hierarchical Swapping Framework，IEEE ACCESS，2017，5：16383~16392，SCIE

[9] Zhang Wenzhe，Lu Kai，Wang Xiaoping，Jian Jie，Fast Persistent Heap Based on Non-Volatile Memory，IEICE TRANSACTIONS ON INFORMATION AND SYSTEMS，2017.5，E100D(5)：1035~1045，SCIE

[10] Pengfei Wang，Jens Krinke，Kai Lu，Gen Li，How Double-Fetch Situations turn into Double-Fetch Vulnerabilities: A Study of Double Fetches in the Linux Kernel，USENIX Security，Vancouver, BC,Canada，2017.8.16-2017.8.18

[11] Lu, Kai，Zhang, Wenzhe，Wang, Xiaoping，Lujan, Mikel，Nisbet, Andy，Flexible Page-level Memory Access Monitoring Based on Virtualization Hardware，13th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE)，Xian, PEOPLES R CHINA，2017.4.8-2017.4.9

专利：

[1] 中国专利，周旭，卢凯，杨灿群，李根，王睿伯，王小平，迟万庆，唐宏伟，刘勇朋，冯华，蒋洁，樊葆华，面向多核并行程序的确定性懒惰释放一致性实现

方法，授权，2016.2.24，CN201510898408.X

[2] 中国专利，周旭 ，卢凯，迟万庆，李根，唐宏伟，刘勇鹏，冯华，王小平，蒋杰，王睿伯，樊葆华，张英，高颖慧，王双喜，陈沉，一种快速的全系统模拟器确定性回放方法，授权，2014.10.17，ZL 2014 1 0551840.7

（三）其他需要说明的问题

1. 申请人同年申请不同类型的国家自然科学基金项目情况（列明同年申请的其他项目的项目类型、项目名称信息，并说明与本项目之间的区别与联系）。

无。

2. 具有高级专业技术职务（职称）的申请人或者主要参与者是否存在同年申请或者参与申请国家自然科学基金项目的单位不一致的情况；如存在上述情况，列明所涉及人员的姓名，申请或参与申请的其他项目的项目类型、项目名称、单位名称、上述人员在该项目中是申请人还是参与者，并说明单位不一致原因。

无。

3. 具有高级专业技术职务（职称）的申请人或者主要参与者是否存在与正在承担的国家自然科学基金项目的单位不一致的情况；如存在上述情况，列明所涉及人员的姓名，正在承担项目的批准号、项目类型、项目名称、单位名称、起止年月，并说明单位不一致原因。

无。

4. 其他。

周旭 简历

中国人民解放军国防科技大学，计算机学院，助理研究员

教育经历（从大学本科开始，按时间倒序排序；请列出攻读研究生学位阶段导师姓名）：

（1）2010-3至2013-12，国防科技大学，计算机科学与技术，博士，导师：卢锡城

（2）2007-9至2009-12，国防科技大学，计算机科学与技术，硕士，导师：卢凯

（3）2003-9至2007-7，国防科技大学，计算机科学与技术，学士

科研与学术工作经历（按时间倒序排序：如为在站博士后研究人员或曾有博士后研究经历，请列出合作导师姓名）：

（1）2013-12至现在，国防科技大学，计算机学院，助理研究员

曾使用其他证件信息（申请人应使用唯一身份证件申请项目，曾经使用其他身份证件作为申请人或主要参与者获得过项目资助的，应当在此列明）

主持或参加科研项目（课题）（按时间倒序排序）：

国家自然科学基金委员会，青年基金项目，61402492，确定性并行技术研究，2015-01至2017-12，26万元，已结题，主持

军委科技委，973项目，6132540302，面向脆弱性分析的控制流并行优化理论与方法，2014-01至2019-12，180万元，已结题，主要参与者

科技部，国家重点研发子课题，2019QY0502，固件fuzzing技术，2019-07至2020年6月，150万元，在研，主持。

代表性研究成果和学术奖励情况

（请注意：①投稿阶段的论文不要列出；②对期刊论文：应按照论文发表时作者顺序列出全部作者姓名、论文题目、期刊名称、发表年代、卷（期）及起止页码（摘要论文请加以说明）；③对会议论文：应按照论文发表时作者顺序列出全部作者姓名、论文题目、会议名称（或会议论文集名称及起止页码）、会议地址、会议时间；④应在论文作者姓名后注明第一/通讯作者情况：所有共同第一作者均加注上标“#”字样，通讯作者及共同通讯作者均加注上标“*”字样，唯一第一作者且非通讯作者无需加注；⑤所有代表性研究成果和学术奖励中本人姓名加粗显示。）

按照以下顺序列出：

一、代表性论著（包括论文与专著，合计5项以内）

(1) 宋从溪; **周旭***; 尹启迪; 何兴陆; 张航玮; 卢凯; P-fuzz: a parallel grey-box fuzzing framework, *IEEE Access*, 2019, 9(23): 0-5100. **(期刊论文)**

(2) Song, Congxi; Yu, Bo*; **Zhou, Xu***; Yang, Qiang*; SPFuzz: A Hierarchical Scheduling Framework for Stateful Network Protocol Fuzzing, *IEEE Access*, 2019, 7: 18490-18499. **(期刊论文)**

(3) Zhang, Gen; **Zhou, Xu***; Luo, Yingqi; Wu, Xugang; Min, Erxue; PTfuzz: Guided Fuzzing With Processor Trace Feedback, *IEEE Access*, 2018, 6: 37302-37313. **(期刊论文)**

(4) Wang, Pengfei*; Krinke, Jens; **Zhou, Xu**; Lu, Kai; AVPredictor: Comprehensive prediction and detection of atomicity violations, *Concurrency and Computation-Practice & Experience*, 2019, 31(15): 0-e5160. **(期刊论文)**

(5) Kai Lu; **Xu Zhou***; Tom Bergan; Xiaoping Wang; Efficient Deterministic Multithreading Without Global Barriers, *Proceedings of the 19th ACM SIGPLAN symposium on Principles and practice of parallel programming(PPoPP '14)*, Orlando Florida USA, 2014-2-22至2014-2-26. **(会议论文)**

二、论著之外的代表性研究成果和学术奖励（包括专利、会议特邀报告等其他成果和学术奖励，请勿在此处再列论文和专著；合计10项以内）

(1) **周旭**; 卢凯; 迟万庆; 李根; 唐宏伟; 一种快速的全系统模拟器确定性回放方法, 2017-5-17, 中国, ZL201410551840.7. **(专利)**

(2) 周旭; 卢凯; 杨灿群; 李根; 王睿伯; 王小平; 迟万庆; 唐宏伟; 刘勇朋; 冯华; 蒋洁; 樊葆华; 面向多核并行程序的确定性懒惰释放一致性实现方法, 2017-10-27, 中国, ZL201510898408.X. **(专利)**

除非特殊说明，请勿删除或改动简历模板中蓝色字体的标题及相应说明文字

卢凯 简历

中国人民解放军国防科技大学，计算机学院，研究员
软件安全智能并行分析湖南省重点实验室，主任

教育经历（从大学本科开始，按时间倒序排序；请列出攻读研究生学位阶段导师姓名）：

(1) 1995.9－1999.12, 国防科技大学，计算机，博士，导师：金士尧

(2) 1991.9－1995.7, 国防科技大学，计算机，学士，导师：

科研与学术工作经历（按时间倒序排序；如为在站博士后研究人员或曾进入博士后流动站（或工作站）从事研究，请列出合作导师姓名）：

(1) 2007.12-至今，国防科技大学，计算机学院，研究员

(2) 2001.12-2007.12，国防科技大学，计算机学院，副研究员

(3) 2001.1-2001.11，国防科技大学，计算机学院，助理讲师

曾使用其他证件信息（申请人应使用唯一身份证件申请项目，曾经使用其他身份证件作为申请人或主要参与者获得过项目资助的，应当在此列明）

主持或参加科研项目（课题）及人才计划项目情况（按时间倒序排序）：

1. 国防科技卓越人才，2017-2022，500 万元，主持
2. 银河-B 超级计算机，2017-2020，98880 万元，主持
3. E 级计算机关键技术验证系统，2016-2018，8882 万元，主持
4. 万人计划，2015 年，30 万元，主持
5. 科技部青年领军人才，2012 年，主持
6. 国家自然科学基金面上项目，61272142，面向大规模高性能并行计算系统的容错计算模型研究，2013-2016，72 万元，完成，主持
7. 国家 863 计划项目，2012AA01A301，天河新一代高效能计算机系统研制（天河二号），2012-2017，180000 万元，完成，副总设计师
8. 总部计划项目，（涉密项目），银河-Z 计算机系统，2012-2015，68000 万元，完成，副总设计师
9. 国家 863 计划项目，2012AA010303，并行程序确定性执行技术与工具，2012-2015，143.3 万元，完成，主持
10. 教育部新世纪优秀人才支持计划，2012-2014，50 万元，完成，主持
11. 武器装备预先研究基金，（涉密项目），高可靠军用操作系统自治管理技术，2011-2015，120 万元，完成，主持
12. 武器装备预先研究基金，（涉密项目），（军队预研项目），2011-2015，270 万 元，完成，主持

13. 武器装备预先研究基金，（涉密项目），新型高性能计算机体系结构

重大基础研究， 2011-2014，1000 万元，完成，主持

14. 国家 863 计划项目，2009AA01A128，千万亿次高效能计算机系统研制（天河一号），2009-2010，60000 万元，完成，副总设计师

15. 总部计划项目，（涉密项目），银河-Y 计算机系统，2007-2009，30800 万元，完成，副总设计师

16. 国家自然科学基金创新群体项目，60621003，千万亿次高性能计算关键技术创新 群体，2007-2009，360 万元，完成，参与

17. 霍英东教育基金，111072，2007-2009，8 万元，完成，主持

18. 武器装备预先研究基金，（涉密项目），支撑能耗管理的安全嵌入式操作系统，2006-2010，80 万元，完成，主持

19. 武器装备预先研究基金，（涉密项目），（军队预研项目），2006-2010，200 万元，完成，主持

20. 总部计划项目，（涉密项目），银河-X 计算机系统，2003-2006，30970 万元，完成，主任设计师

代表性研究成果和学术奖励情况（每项均按时间倒序排序）

（请注意：①投稿阶段的论文不要列出；②对期刊论文：应按照论文发表时作者顺序列出全部作者姓名、论文题目、期刊名称、发表年代、卷（期）及起止页码（摘要论文请加以说明）；③对会议论文：应按照论文发表时作者顺序列出全部作者姓名、论文题目、会议名称(或会议论文集名称及起止页码)、会议地址、会议时间；④应在论文作者姓名后注明第一/通讯作者情况：所有共同第一作者均加注上标“#”字样，通讯作者及共同通讯作者均加注上标“*”字样，唯一第一作者且非通讯作者无需加注；⑤所有代表性研究成果和学术奖励中本人姓名加粗显示。）

按照以下顺序列出：①10篇以内代表性论著；②论著之外的代表性研究成果和学术奖励。

一、 近 5 年内发表的 5 篇代表性论著

(1) Lu, Kai (#)(*); Wang, Xiaoping; Li, Gen; Wang, Ruibo; Chi, Wanqing; Liu, Yongpeng; Tang, Hongwei; Feng, Hua; Gao, Yinghui, Iaso: an autonomous fault-tolerant management system for supercomputers , Frontiers of Computer Science, 2014.6, 8(3): 378~390 (期刊论文)

(2) Kai Lu (#); Xu Zhou; Tom Bergan; Xiaoping Wang, Efficient Deterministic Multithreading Without Global Barriers, 19th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP), 2014.2.15-2014.2.19 (会议论文)

(3) Pengfei Wang (#); Jens Krinke; Kai Lu; Gen Li; Steve Dodier-Lazaro, How Double-Fetch Situations turn into Double- Fetch Vulnerabilities: A Study of Double Fetches in the Linux Kernel, USENIX Security Symposium, 2017.8.16-2017.8.18 (会议论文)

(4) Kai Lu (#); Wenzhe Zhang; Xiaoping Wang; Mikel LUJAN; Andy Nisbet, Flexible Page-level Memory Access Monitoring Based on Virtualization Hardware, ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE 2017), 2017.4.8-2017.4.9 (会议论文)

(5) Songlei Jian (#); Longbing Cao; Guansong Pang; Kai Lu; Hang Gao, Embedding-based Representation of Categorical Data by Hierarchical Value Coupling Learning, International Joint Conference on Artificial Intelligence 2017 (IJCAI), 2017.8.19-2017.8.25 (会议论文)

二、 近 5 年内发表的其余论著

(1) Songlei Jian (#); Liang Hu; Longbing Cao; Kai Lu Metric-based Auto-Instructor for Learning Mixed Data Representation, AAAI Conference on Artificial Intelligence (AAAI-18), 2018.2.2-2018.2.7 (会议论文)

(2) Huijun Wu (#); Chen Wang; Jie Yin; Kai Lu; Liming Zhu, Sharing Deep Neural Network Models with Interpretation, WWW 2018 : International World Wide Web Conference, 2018.1.1-2018.1.1 (会议论文)

(3) Huijun Wu (#); Chen Wang; Yinjin Fu; Sherif Sakr; Kai Lu; Liming Zhu, A Differentiated Caching Mechanism to Enable Primary Storage Deduplication in Clouds, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS (TPDS), 2018.1, 1 (1) (期刊论文)

(4) Huijun Wu (#); Chen Wang; Yinjin Fu; Sherif Sakr; Liming Zhu; Kai Lu HPDedup: A Hybrid Prioritized Data Deduplication Mechanism for Primary Storage in the Cloud, MSST 2017 : IEEE Symposium on Massive Storage Systems and Technologies, 2017.5.15-2017.5.17 (会议论文)

(5) Kai Liu (#); Yi Zhang; Kai Lu(*); Xiaoping Wang, Restoration for Noise Removal in Quantum Images, Int J Theor Phys, 2017, 1(1): 1~20 (期刊论文)

(6) X Wang (#); J Song; K Lu (*); X Wang, Community detection in attributed networks based on heterogeneous vertex interactions, Applied Intelligence, 2017, 1(3):

1~12 (期刊论文)

(7) Kai Lu (#)(*), Let Data Tell the Truth: a Study on Automatically Error Message Reasoning of MilkyWay-2 Supercomputer, JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY (期刊论文)

(8) Wenzhe Zhang; Kai Lu; Xiaoping Wang, Hybrid Efficient Memory Access Monitoring, International Conference on Communication Systems and Computing Application Science, 2016.3.19-2016.3.20 (会议论文)

(9) Wenzhe Zhang; Kai Lu; Xiaoping Wang, Fault Tolerant Barnes-Hut Algorithm on Non-Volatile Memory, International Conference on Communication Systems and Computing Application Science, 2016.3.19-2016.3.20 (会议论文)

(10) Zhenwei Wu (#); Kai Lu (*); Xiaoping Wang; Wanqing Chi, Topology-aware network fault influence domain analysis, Computers and Electrical Engineering, 2016, 1(1) (期刊论文)

(11) Wenzhe Zhang; Kai Lu; Xiaoping Wang, High Efficient General Memory Allocator, The International Conference on Computer Science and Technology, 2016.1.8-2016.1.10 (会议论文)

(12) Wenzhe Zhang; Kai Lu; Xiaoping Wang, Optimizing Checkpoint with Record and Replay, The International Conference on Computer Science and Technology, 2016.1.8-2016.1.10 (会议论文)

(13) Wenzhe Zhang; Kai Lu; Mikel Luján; Xiaoping Wang; Xu Zhou, Write-Combined Logging: An Optimized Logging for Consistency in NVRAM, Scientific Programming, 2015.12.27, 2015(3): 1~13 (期刊论文)

(14) Zhendong Wu; Kai Lu; Xiaoping Wang, Identifying Repeated Interleavings to Improve the Efficiency of Concurrency Bug Detection, International Conference on Algorithms and Architectures for Parallel Processing, 2015.11.18-2015.11.20 (会议论文)

(15) Zhendong Wu; Kai Lu; Xiaoping Wang, Efficiently Trigger Data Races through Speculative Execution, IEEE International Conference on High Performance and Communications, 2015.8.24-2015.8.26 (会议论文)

(16) Zhendong Wu; Kai Lu; Xiaoping Wang; Xu Zhou; Chen Chen, Detecting harmful data races through parallel verification, Journal of Supercomputing, 2015.秋

季, 71(8): 2922~2943 (期刊论文)

(17) Zhendong Wu; Kai Lu; Xiaoping Wang; Xu Zhou, Collaborative technique for concurrency bug detection, International Journal of Parallel Programming, 2015.夏季, 43(2): 260~285 (期刊论文)

(18) Kai Lu (#); Zhendong Wu; Xiaoping Wang; Chen Chen; Xu Zhou, RaceChecker: Efficient Identification of Harmful Data Races, Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, 2015.3.4-2015.3.6 (会议论文)

(19) Yi Zhang; Kai Lu; Kai Xu; Yinghui Gao; Richard Wilson, Local Feature Point Extraction for Quantum Images, Quantum Information Process, 2015.春季, 14: 1573~1588 (期刊论文)

(20) Kai Lu (*); Xu Zhou; Xiaoping Wang; Tom Bergan; Chen Chen, An Efficient and Flexible Deterministic Framework for Multithreaded Programs, Journal of Computer Science and Technology, 2015.春季, 20(1): 42~56 (期刊论文)

(21) Zhendong Wu; Kai Lu; Xiaoping Wang; Xu Zhou; Chen Chen, Pfinder: Efficiently Detecting Bugs in Concurrent Programs through Parallelizing RaceVerification, International Conference on Computer Engineering and Systems, 2014.12.22-2014.12.24 (会议论文)

(22) Bo Yang (*); Kai Lu ; Ying-hui Gao; Kai Xu; Xiao-ping Wang; Zhi-quan Cheng, Improving vertex-frontier based GPU breadth-first search, Journal of Central South University, 2014.冬季, 21(10): 3828~3836 (期刊论文)

(23) Chen Chen; Kai Lu; Xiaoping Wang; Xu Zhou; Zhendong Wu, DRDet: Efficiently Making Data Races Deterministic, IEICE Transactions on Information and Systems, 2014.冬季, E97.D(10): 2676~2684 (期刊论文)

(24) Kai Lu (#); Yi Zhang; Kai Xu; Yinghui Gao; Richard Wilson, Approximate Maximum Common Sub-graph Isomorphism Based on Discrete-Time Quantum Walk, International Conference on Pattern Recognition, 2014.8.24-2014.8.28 (会议论文)

(25) 孙曼晖; 卢凯; 王小平, 基于NVRAM的函数式语言容错机制, 计算机工程工程与工艺年会, 2014.7.30-2014.8.3 (会议论文)

(26) Huijun Wu; Kai Lu; Gen Li; Design and Implementation of Distributed Stage DB: A High Performance Distributed Key-Value Database, International Asia

Conference on Industrial Engineering and Management Innovation, 2014.5.16-2014.5.18 (会议论文)

(27) Zhang Yi; Lu Kai; Gao Ying-hui, Fast image matching algorithm based on affine invariants, Journal of Central South University, 2014.夏季, 21(5): 1907~1918 (期刊论文)

(28) Xiaoping Wang; Yunhao Liu; Zheng Yang; Kai Lu; Jun Luo, Robust Component-Based Localization in Sparse Networks, IEEE Transactions on Parallel & Distributed Systems, 2014.夏季, 25(5): 1317~1327 (期刊论文)

(29) 李崇飞; 高颖慧; 卢凯; 曲智国, 基于结构相似度的视觉显著性检测方法, 计算机工程与科学, 2013.冬季, 35(10): 181~185 (期刊论文)

(30) 张毅; 卢凯; 高颖慧, 量子算法与量子衍生算法, 计算机学报, 2013.秋季, 36(9): 1835~1842 (期刊论文)

(31) Yi Zhang; Kai Lu; Yinghui Gao; Kai Xu, A novel quantum representation for log-polar images, Quantum Information Processing, 2013.秋季, 12(9): 3103~3126 (期刊论文)

(32) Yi Zhang; Kai Lu; Yinghui Gao; Mo Wang, NEQR: a novel enhanced quantum representation of digital images, Quantum Information Processing, 2013.秋季, 12 (8): 2833~2860 (期刊论文)

(33) 冯华; 卢凯; 王小平, 面向多核处理器的实时优化技术:基于独立实时域的实时优化方法, 计算机科学, 2013.夏季, 40(9): 159~162 (期刊论文)

(34) Kai Lu (#); Xu Zhou; Xiaoping Wang; Wenzhe Zhang; Gen Li, RaceFree: an efficient multi-threading model for determinism, ACM SIGPLAN Notices, 2013.秋季, 48(8): 297~298 (期刊论文)

(35) Chen Chen; Kai Lu; Xiaoping Wang; Xu Zhou; Li Fang, Pruning False Positives of Static Data-Race Detection via Thread Specialization, Advanced Parallel Processing Technologies, 2013.8.27-2013.8.28 (会议论文)

(36) Zhendong Wu; Kai Lu; Xiaoping Wang; Xu Zhou, ColFinder Collaborative Concurrency Bug Detection, 13th International Conference on Quality Software, 2013.7.29-2013.7.30 (会议论文)

(37) Zhou, Xu; Lu, Kai; Wang, Xiaoping; Zhang, Wenzhe; Zhang, Kai; Li, Xu; Li, Gen, Deterministic Message Passing for Distributed Parallel Computing , IEICE

Transactions on Information and Systems, 2013.5, E96D(5): 1068~1077 (期刊论文)

(38) Changfei Zhou; Kai Lu; Wanqing Chi; Xiaoping Wang; Zhenhai Xiong, MIC Acceleration of Saliency Detection Algorithm, International Workshop on Cloud Computing and Information Security, 2013.5.2-2013.5.3 (会议论文)

(39) Guoliang Zhu; Kai Lu; Xu Li, SCM-BSIM: A Non-Volatile Memory Simulator Based on BOCHS, Emerging Technologies for Information Systems, Computing, and Management, 2013.5.2-2013.5.5 (会议论文)

(40) Wang, Xiaoping; Liu, Yunhao; Yang, Zheng; Lu, Kai; Luo, Jun, OFA: An optimistic approach to conquer flip ambiguity in network localization , Computer Networks, 2013.4.22, 57(6): 1529~1544 (期刊论文)

(41) 卢凯(#)(*); 周旭, 确定性执行技术, 中国计算机学会通讯, 2013.春季, 9(2) (期刊论文)

(42) Tang Hong-wei; Cao Jian-nong; Sun Cai-xia; Lu Kai, REA-MAC: A low latency routing-enhanced asynchronous duty-cycle MAC protocol for wireless sensor networks , Journal of Central South University, 2013.3, 20(3): 678~687 (期刊论文)

(43) Li, Xu; Lu, Kai; Wang, Xiaoping; Dai, Bin; Zhou, Xu, Understanding the Impact of BPRAM on Incremental Checkpoint , IEICE Transactions on Information and Systems, 2013.3, E96D(3): 663~672 (期刊论文)

(44) Hong Zhu; Yongpeng Liu; Kai Lu; Xiaoping Wang, Self-Adaptive Power Management of Idle Nodes in Large Scale Systems, International Journal of Next-Generation Computing, 2013, 4(2) (期刊论文)

(45) Mo Wang; Kai Lu; Yi Zhang; Xiaoping Wang, FLPI: representation of quantum images for log-polar coordinate, The 5th International Conference on Digital Image Processing, 2013 (会议论文)

(46) Yi Zhang; Kai Lu; Yinghui Gao, Analysis of image thresholding segmentation algorithms based on swarm intelligence, Fifth International Conference on Machine Vision, 2013 (会议论文)

(47) 卢凯(#); 熊振海; 李崇飞; 李根, 基于全局结构相似度度量方法的显著性检测, 计算机工程与科学, 2013, 35(6): 113~117 (期刊论文)

(48) 蒋杰; 方力; 卢凯; 刘杰; 武林平, SSTD:基于栈帧分析的可扩展并行程序调试工具 , 计算机工程与科学, 2013.4.15, (04): 8~13 (期刊论文)

(49) 刘勇鹏; 卢凯; 迟万庆, 机群系统中空闲结点的功耗管理 , 计算机科学, 2013, 40(4): 59~63 (期刊论文)

三、 已发表的其余论著

(1) Wang Rui-bo; Lu Kai; Lu Xi-cheng, Aware conflict detection of non-uniform memory access system and prevention for transactional memory , JOURNAL OF CENTRAL SOUTH UNIVERSITY, 2012.8, 19(8): 2266~2271 (期刊论文)

(2) Zhou, Xu (#); Wang, Xiaoping; Li, Xu, Lu, Kai (*) Exploiting parallelism in deterministic shared memory multiprocessing , Journal of Parallel and Distributed Computing, 2012.5, 72(5): 716~727 (期刊论文)

(3) Liu, Yongpeng; Zhu, Hong; Lu, Kai; Liu, Yongyan, A Power Provision and Capping Architecture for Large Scale Systems , 26th IEEE International Parallel and Distributed Processing Symposium (IPDPS), Shanghai, PEOPLES R CHINA, 2012.5.21-2012.5.25 (会议论文)

(4) Kai Lu; Wenzhe Zhang; Xu Zhou, Strong Atomicity: An Efficient and Easy-to-Use Mechanism to Guarantee Atomicity, The 2nd International Conference on Computer Science and Service System, 2012 (会议论文)

(5) Yongpeng Liu; Hong Zhu; Kai Lu; Xiaoping Wang, Self-adaptive management of the sleep depths of idle nodes in large scale systems to balance between energy consumption and response times, CloudCom, 2012.5.1-2012.5.3 (会议论文)

(6) Xu Zhou; Kai Lu; Xicheng Lu; Xiaoping Wang; Baohua Fan, dMPI: Facilitating Debugging of MPI Programs via Deterministic Message Passing, IFIP International Conference on Network and Parallel Computing, 2012 (会议论文)

(7) Bo Yang; Kai Lu; Jie Liu; Xiaoping Wang; Chunye Gong, Hybrid Embarrassingly Parallel algorithm for heterogeneous CPU/GPU clusters, 7th International Conference on Computing and Convergence Technology, 2012 (会议论文)

(8) Bo Yang; Kai Lu; Jie Liu; Xiaoping Wang; Chunye Gong, GPU accelerated Monte Carlo simulation of deep penetration neutron transport, 2nd IEEE International Conference on Parallel, Distributed and Grid Computing, 2012 (会议论文)

(9) Xu Li; Kai Lu; Xiaoping Wang; Xu Zhou, NV-process: A Fault-Tolerance

Process Model Based on Non-Volatile Memory [C]. Proceedings of the Asia-Pacific Workshop on Systems, ACM SIGOPS Asia-Pacific Workshop on Systems, 2012 (会议论文)

(10) Xu Li; Kai Lu; Xu Zhou, NV-TS: A Fault Tolerance Transaction System Based on Persistent Memory, Proceedings of 2012 International Conference on Computer Science and Electronics Engineering, 2012 (会议论文)

(11) 张文喆; 卢凯; 周旭, 一种有效的检测Ad-hoc同步的方法, HPC China, 2012 (会议论文)

(12) Xu Li; Kai Lu; Xu Zhou, HG-ckpt: A tybrid granularity checkpoint frame, Proceedings of 2012 IEEE International Conference on Information Science and Technology, 2012 (会议论文)

(13) 刘勇鹏; 王锋; 卢凯; 刘勇燕, 面向异构并行计算系统的流水线式压缩检查点, 电子学报, 2012, (02): 223~229 (期刊论文)

(14) 李崇飞; 高颖慧; 卢凯; 曲智国, 基于相位谱和调谐幅度谱的显著性检测方法, 中国图象图形学报, 2012, (07): 821~827 (期刊论文)

(15) 刘勇燕; 刘勇鹏; 卢凯, 功耗管理透明虚拟化, 计算机工程与科学, 2012, (08): 75~80 (期刊论文)

(16) 胡昊; 卢凯; 李根; 王小平; 李旭; 周旭, 一个高性能Key/Value数据库XDB的设计与实现, 计算机工程与科学, 2012, (12): 140~143 (期刊论文)

(17) 陶玉龙; 卢凯; 王小平; 王继祖, 面向云服务的高性能计算柔性服务平台 , 信息网络安全, 2012.3, (06): 57~60 (期刊论文)

(18) Xu Zhou; Kai Lu; Xicheng Lu; Chen Chen; Xu Li, RReplay: A record & replay system based on restricted multi-threading, CSSS, 2012 (会议论文)

(19) 张毅; 卢凯; 高颖慧; 杨博, 量子图像存储技术综述, 计算机科学, 2012, 11: 203~208 (期刊论文)

(20) Xu Li; Kai Lu; Xu Zhou, TM-Dissector: an Overhead Model for Software Transactional Memory, Proceedings of 2012 IEEE International Conference on Information Science and Technology, 2012 (会议论文)

(21) Xu Li; Kai Lu; Xu Zhou, SIM-PCM: A PCM Simulator Based on Simics, International Conference on Computational and Information Sciences, 2012.8.17-2012.8.19 (会议论文)

(22) Xu Li; Kai Lu; Xu Zhou, SIM-PCM: A PCM Simulator Based on Simics, Proceedings of the 4th International Conference on Computational and Information Sciences, 2012 (会议论文)

(23) Zhang, Wenzhe; Lu, Kai; Zhou, Xu, DetSF: A Deterministic Scheduling Framework, International Conference on Computer Science and Network Technology (ICCSNT), Harbin, PEOPLES R CHINA, 2011.12.24-2011.12.26 (会议论文)

(24) 章婧; 卢凯; 周旭, Java程序内存行为研究, 小型微型计算机系统, 2011, (08): 1617~1621 (期刊论文)

(25) Yang, Xue-Jun; Liao, Xiang-Ke; Lu, Kai; Hu, Qing-Feng; Song, Jun-Qiang; Su, Jin-Shu, The TianHe-1A Supercomputer: Its Hardware and Software, Journal of Computer Science and Technology, 2011.5, 26(3): 344~351 (期刊论文)

(26) 卢凯(*); 迟万庆; 高颖慧; 冯华, MIOS:面向大规模CCNUMA系统的多实例操作系统, 计算机研究与发展, 2011, (09): 1693~1703 (期刊论文)

(27) 王鹤; 卢凯, 可满足性求解器算法基于GPU的加速研究, 计算机应用与软件, 2011, (10): 58~61 (期刊论文)

(28) 王睿伯; 卢锡城; 卢凯; 王绍刚, 面向CC-NUMA体系结构的事务内存冲突规避方法, 计算机学报, 2011, (04): 676~683 (期刊论文)

(29) 周旭; 卢凯; 李根, 一种安全高效的Java操作系统IPC机制的设计与实现, 计算机应用与软件, 2011, (01): 137~141+149 (期刊论文)

(30) 李崇飞; 曲智国; 卢凯; 高颖慧, 基于侧抑制频谱调谐的显著性检测方法, 计算机科学, 2011, (12): 258~262 (期刊论文)

(31) Hongwei Tang; Caixia Sun; Kai Lu; Yong Peng Liu, MPT-MAC: A Multiple Packets Transmission MAC Protocol for Wireless Sensor Networks, The Fifth International Conference on Sensor Technologies and Applications, 2011 (会议论文)

(32) 李根; 卢凯; 张英; 卢锡城; 冯华; 张巍, Hunter:一种指令集体系结构无关的二进制级动态测试用例生成技术, 计算机工程与科学, 2011, (04): 69~74 (期刊论文)

(33) 冯华; 唐宏伟; 卢凯, 一种新的Hypervisor逻辑域通道设计, 计算机工程与科学, 2011, (09): 52~56 (期刊论文)

(34) Xuejun Yang; Xiangke Liao; Weixia Xu; Junqiang Song; Qingfeng Hu;

Jinshu Su; Liquan Xiao; Kai Lu; Qiang Dou; Juping Jiang; Canqun Yang, TH-1: China’s first petaflop supercomputer, Frontiers of Computer Science in China, 2010. 冬季, 4(4): 445~455 (期刊论文)

(35) 刘勇鹏; 朱鸿; 卢凯; 迟万庆; 刘勇燕, 基于NInO模型的大规模计算系统功耗控制, 软件学报, 2011, 22(2): 199~207 (期刊论文)

(36) Canqun Yang; Feng Wang; Yunfei Du; Juan Chen; Jie Liu; Huizhan Yi; Kai Lu, Adaptive Optimization for Petascale Heterogeneous CPU/GPU Computing, CLUSTER, 2010-2010 (会议论文)

(37) Ruibo Wang; Kai Lu, Using Transactional Memory on CC-NUMA systems. 6th International Conference on Networked Computing, INC, 2010 (会议论文)

(38) Ruibo Wang; Kai Lu; XIcheng Lu, Brief announcement: NUMA-aware Transactional Memory, Annual ACM Symposium on Principles of Distributed Computing, 2010 (会议论文)

(39) 冯华; 唐宏伟; 卢凯; 刘勇鹏, OpenSparc T2处理器虚拟化技术研究, 计算机工程与科学, 2010.7.15, (07): 72~75 (期刊论文)

(40) 卢凯(*); 周旭; 迟万庆, FLSP:一个高效的系统级垃圾收集算法, 计算机工程与科学, 2010, 32(11): 119~124 (期刊论文)

(41) 卢凯(*); 迟万庆; 刘勇鹏; 唐宏伟, 高效能计算机系统虚拟化技术研究, 计算机工程与科学, 2010.7.15, (07): 53~57 (期刊论文)

(42) 卢锡城; 李根; 卢凯; 张英, 面向高可信软件的整数溢出错误的自动化测试, 软件学报, 2010.2.15, (02): 179~193 (期刊论文)

(43) Li, Gen; Lu, Kai; Zhang, Ying; Lu, Xicheng; Zhang, Wei, Decoupling Dynamic Test Generation from Specific Operating System Details Based on Whole System Virtual Machine , 4th International Conference on Frontier of Computer Science and Technology, Shanghai, PEOPLES R CHINA, 2009.12.17-2009.12.19 (会议论文)

(44) Li, Gen; Lu, Kai; Zhang, Ying; Lu, Xicheng; Zhang, Wei, Decoupling Binary-Level Dynamic Test Generation from Specific Architecture Details , 4th International Conference on Computer Sciences and Convergence Information Technology, Seoul, SOUTH KOREA, 2009.11.24-2009.11.26 (会议论文)

(45) Li, Gen; Lu, Kai; Zhang, Ying; Li, Xicheng; Zhang, Wei, Architectureand

OS-Independent Binary-Level Dynamic Test Generation , 11th International Conference on Information and Communications Security, Beijing, PEOPLES R CHINA, 2009.12.14-2009.12.17 (会议论文)

(46) Ruibo Wang; Kai Lu; Xicheng Lu, Two-phase conflict detection for transactional memory on clusters, CLUSTER, 2009 (会议论文)

(47) Ruibo Wang; Kai Lu; Xicheng Lu, Investigating transactional memory performance on ccNUMA machines, 18th ACM International Symposium on High Performance Distributed Computing (HPDC 2009), 2009 (会议论文)

(48) Ruibo Wang; Kai Lu, Hierarchical Conflict Detection for Cluster's Transactional Memory, 5th International Joint Conference on INC, IMS and IDC (NCM2009), 2009 (会议论文)

(49) Gen Li; Kai Lu; Ying Zhang; Xicheng Lu; Wei Zhang, Mixing Concrete and Symbolic Execution to Improve the Performance of Dynamic Test Generation, 3rd International Conference on New Technologies, Mobility and Security, NTMS 2009, 2009 (会议论文)

(50) 刘勇鹏; 卢凯; 刘勇燕; 武林平; 陈娟, 高性能计算中处理器功耗特征的评测与分析, 计算机工程与科学, 2009.11.15, (11): 102~105 (期刊论文)

(51) 史佩昌; 王怀民; 蒋杰; 卢凯, 面向云计算的网络化平台研究与实现, 计算机工程与科学, 2009.10.15, (S1): 249~252 (期刊论文)

(52) 王睿伯; 卢凯; 卢锡城, 集群上软件事务内存的层次化冲突检测, 计算机工程与科学, 2009.10.15, (S1): 136~137+190 (期刊论文)

(53) Lu, Kai; Chi, Wanqing; Liu, Yongpeng; Tang, Hongwei, HPVZ: A High Performance Virtual Computing Environment for Super Computers 8th International Symposium on Advanced Parallel Processing Technologies, Rapperswil, SWITZERLAND, 2009.8.24-2009.8.25 (会议论文)

(54) 李旭; 卢凯; 李根, Jikes RVM动态编译技术分析与性能评测, 计算机科学, 2009.4.15, (04): 129~132 (期刊论文)

(55) 高颖慧; 卢凯; 沈振康, 参数化通用一位量子门模型, 信号处理, 2009.4.25, (04): 543~547 (期刊论文)

(56) 高颖慧; 卢凯; 沈振康, 逐级目标淘汰量子遗传算法, 信号处理, 2009.2.25, (02): 238~242 (期刊论文)

(57) 冯华; 卢凯; 刘勇鹏, 高性能计算系统中的服务质量研究, 第八届全国信息隐藏与多媒体安全学术大会湖南省计算机学会第十一届学术年会, 中国湖南长沙, 2009.3.28-2009.4.2 (会议论文)

(58) Kai Lu (#); Wanqing Chi; Yongpeng Liu; Hongwei Tang, HPVZ: A High Performance Virtual Computing Environment for Super Computers, International Workshop on Advanced Parallel Processing Technologies, 2009.8.24-2009.8.25 (会议论文)

(59) 高颖慧; 卢凯; 沈振康, 基于角度编码染色体量子遗传算法的模板匹配, 计算机应用研究, 2008.11.15, (11): 3509~3512 (期刊论文)

(60) 马俊; 李根; 卢凯, 基于模拟器的缓冲区溢出动态检测方法, 全国第19届计算机技术与应用（CACIS）学术会议, 中国安徽合肥, 2008.7.1-2008.7.3 (会议论文)

(61) 鲁珺; 卢凯; 谭郁松, Keta中的Socket复用技术, 计算机工程与科学, 2007.1.30, (01): 22~25 (期刊论文)

(62) 张巍; 卢凯, 基于FreeBSD内核的Chamber技术, 计算机工程, 2007.9.20, (18): 93~94+136 (期刊论文)

(63) 谭郁松; 戴华东; 卢凯; 刘进元; 陈暄; 鲁; 李姗姗, 基于软流水体系结构的内核Web服务器——KETA, 计算机工程与科学, 2006.10.30, (10): 138~142 (期刊论文)

(64) 许立; 罗军; 卢凯, 具有节点亲近能力的NUMA调度算法, 计算机工程, 2006.1.5, (01): 99~101+156 (期刊论文)

(65) 杨梦梦; 卢凯; 卢锡城, 内存管理系统对NUMA的支持及优化, 计算机工程, 2005.8.20, (16): 80~82+109 (期刊论文)

(66) 卢凯; 胡湘华, 高性能计算机的系统分区技术研究, 计算机科学, 2004.12.25, (12): 179~181+204 (期刊论文)

(67) 黄进; 卢凯; 廖湘科, Linux应用二进制兼容技术研究与实现, 计算机工程, 2004.8.20, (16): 96~98 (期刊论文)

(68) Dongsheng Li; Xinxin Fang; Yijie Wang; Xicheng Lu; Kai Lu; Nong Xiao, A Scalable Peer-to-Peer Network with Constant Degree, APPT, 2003-2003 (会议论文)

(69) Dongsheng Li; Nong Xiao; Xicheng Lu; Yijie Wang; Kai Lu, Dynamic

Self-Adaptive Replica Location Method in Data Grids, CLUSTER, 2003-2003 (会议论文)

四、 论著之外的研究成果和学术奖励

(1) 卢凯(1/1), 天河一号高效能计算机系统, 国务院 特等奖, 科技进步, 其他, 2014.12.12 (卢凯) (科研奖励)

(2) 卢凯(1/1), 银河-X计算机系统, 国务院, 科技进步, 国家一等奖, 2009 (卢凯) (科研奖励)

(3) 卢凯(1/1), 国防科学技术大学高性能计算创新团队, 国务院, 科技进步, 国家一等奖, 2012.12 (卢凯) (科研奖励)

(4) 卢凯(1/1), 银河-Y并行操作系统, 总装备部, 科技进步, 省部一等奖, 2013.11 (卢凯) (科研奖励)

(5) 卢凯(1/1), 银河-X计算机并行操作系统, 总装备部, 科技进步, 省部一等奖, 2008.12 (卢凯) (科研奖励)

(6) 卢凯(1/1), 银河-W并行计算机, 总装备部, 科技进步, 省部一等奖, 2006.9 (卢凯) (科研奖励)

(7) 卢凯(1/1), 银河麒麟服务器操作系统, 湖南省, 科技进步, 省部一等奖, 2008.10 (卢凯) (科研奖励)

(8) 卢凯(1/1), 大规模并行处理技术, 总装备部, 科技进步, 省部二等奖, 2001.10 (卢凯) (科研奖励)

(9) 卢凯(1/1), 银河-U并行程序开发环境, 总装备部, 科技进步, 省部二等奖, 2001.9 (卢凯) (科研奖励)

(10) 冯华; 唐遇星; 唐宏伟; 迟万庆; 卢凯; 蒋杰; 刘勇鹏; 王睿伯; 王小平; 高颖慧; 樊葆华; 李根, 基于场景的处理器系统级验证完备性度量方法, 2013.5.7, 中国, ZL201310164429.X (专利)

(11) 王小平; 樊葆华; 卢凯; 冯华; 蒋杰; 刘勇鹏; 唐宏伟; 王睿伯; 李根; 高颖慧; 迟万庆, 面向多虚拟域可定制的PICe外设设备树生成方法, 2013.5.7, 其他国家, CN201310164137.6 (专利)

(12) 李根; 王睿伯; 卢凯; 迟万庆; 冯华; 蒋杰; 刘勇鹏; 高颖慧; 唐宏伟; 樊葆华; 王小平, Key-Value数据库用户请求的高速并发处理方法, 2013.5.7, 其他国家, CN201310164022.7 (专利)

(13) 卢凯; 迟万庆; 冯华; 蒋杰; 唐宏伟; 樊葆华; 王睿伯; 李根; 王小平;

高颖慧; 刘勇鹏, 用于大规模计算阵列操作系统的网络路由配置方法, 2013.5.7, 其他国家, CN201310163986.X (专利)

(14) 卢凯; 廖湘科; 迟万庆; 蒋艳凰; 冯华; 刘勇鹏; 唐宏伟; 高颖慧; 蒋杰, 基于操作系统网络驱动的无盘计算机启动方法, 2010.3.5, 其他国家, CN201010118456.X (专利)

(15) 卢凯; 迟万庆; 冯华; 刘勇鹏, 基于跨操作系统共享存储的消息通信方法SMCI, 2012.9.19, 中国, ZL200710082460.3 (专利)

(16) 廖湘科; 卢凯; 迟万庆; 冯华; 刘勇鹏; 唐宏伟, 基于同步寄存器的全局优化锁方法, 2012.8.8, 中国, ZL200810075044.5 (专利)

(17) 卢凯; 迟万庆; 冯华; 刘勇鹏; 唐宏伟; 黎铁军; 唐遇星; 张英; 欧国东; 高颖慧, 基于硬件模拟器的抽屉式Ramdisk调测试技术DPRD, 2010.8.4, 中国, ZL200910131912.3 (专利)

(18) 卢凯; 迟万庆; 冯华; 刘勇鹏; 唐宏伟; 高颖慧, 大规模并行计算的作业级资源统计与控制方法JRSC, 2010.8.4, 中国, ZL200910121913.8 (专利)

(19) 卢凯; 迟万庆; 冯华, 面向通信的分组并行输入/输出服务方法, 2004.5.28, 其他国家, CN200410023254.1 (专利)

(20) 刘勇鹏; 庞征斌; 张峻; 蒋杰; 迟万庆; 卢凯; 唐宏伟; 樊葆华; 王睿伯; 李根; 王小平; 高颖慧; 冯华, 大规模I/O共享系统PCI桥非预取访存空间扩展方法, 2013.5.7, 其他国家, CN201310164110.7 (专利)

(21) 陈海涛; 卢宇彤; 卢凯; 张伟; 周恩强; 谢旻; 董勇; 蒋艳凰; 曹宏嘉; 所光, 一种云存储客户端的低延迟元数据访问方法, 2012.11.22, 其他国家, CN201210479062.6 (专利)

(22) 廖湘科; 李春江; 杜云飞; 晏小波; 隋兵才; 邓让钰; 王永文; 杨灿群; 窦强; 徐炜遐; 卢凯, 用于共享存储多核多线程处理器硬件锁的验证方法, 2012.5.16, 其他国家, CN201210151448.4) (专利)

(23) 谭郁松; 戴华东; 卢凯, 构建基于软流水结构的Web服务器的方法及其服务器, 2005.6.24, 其他国家, CN200510031746.X (专利)

(24) 戴华东; 卢凯; 秦莹, 基于操作系统反向页表的页迁移和复制方法, 2004.2.27, 其他国家, CN200410022933.7 (专利)

(25) 戴华东; 卢凯; 秦莹, 基于即态访问信息的动态页迁移方法, 2004.2.25, 其他国家, CN200410022921.4 (专利)

除非特殊说明，请勿删除或改动简历模板中蓝色字体的标题及相应说明文字

参与者 简历

王鹏飞，国防科技大学，计算机学院网络空间安全系，助理研究员

教育经历（从大学本科开始，按时间倒序排序；请列出攻读研究生学位阶段导师姓名）：

2014/02－2018/06，国防科技大学，计算机学院网络空间安全系，博士，导师：卢凯

2011/09－2013/12，国防科技大学，计算机学院网络工程系，硕士，导师：赵文涛

2007/09－2011/06，国防科技大学，计算机学院网络工程系，学士

科研与学术工作经历（按时间倒序排序；如为在站博士后研究人员或曾进入博士后流动站（或工作站）从事研究，请列出合作导师姓名）：

1. 2018/06-至今，国防科技大学，计算机学院网络空间安全系，助理研究员
2. 2015/10-2016/10，英国伦敦大学学院，计算机科学系，访问学者

曾使用其他证件信息（申请人应使用唯一身份证件申请项目，曾经使用其他身份证件作为申请人或主要参与者获得过项目资助的，应当在此列明）

无

主持或参加科研项目（课题）及人才计划项目情况（按时间倒序排序）：

1. 国家自然科学基金青年项目，61902412，计算机系统跨边界漏洞检测技术研究，2020/01-2022/12，30万元，在研，主持

代表性研究成果和学术奖励情况（每项均按时间倒序排序）

（请注意：①投稿阶段的论文不要列出；②对期刊论文：应按照论文发表时作者顺序列出全部作者姓名、论文题目、期刊名称、发表年代、卷（期）及起止页码（摘要论文请加以说明）；③对会议论文：应按照论文发表时作者顺序列出全部作者姓名、论文题目、会议名称(或会议论文集名称及起止页码)、会议地址、会议时间；④应在论文作者姓名后注明第一/通讯作者情况：所有共同第一作者均加注上标“#”字样，通讯作者及共同通讯作者均加注上标“*”字样，唯一第一作者且非通讯作者无需加注；⑤所有代表性研究成果和学术奖励中本人姓名加粗显示。）

按照以下顺序列出：①10篇以内代表性论著；②论著之外的代表性研究成果和学术奖励。

采用以下格式填写：

一、期刊论文

(1) **Pengfei Wang** (#)(*), Kai Lu, Gen Li, Xu Zhou, DFTracker: Detecting Double-fetch Bugs by Multi-taint Parallel Tracking, Frontier of Computer Science, 2019, 13(2): 247-263.

(2) **Pengfei Wang** (#)(*), Jens Krinke, Xu Zhou, Kai Lu, AVPredictor: Comprehensive Prediction and Detection of Atomicity Violations, Concurrency and Computation: Practice and Experience, 2019. 31(7):35160.

(3) Kai Lu(#), **Pengfei Wang(***), Gen Li, Xu Zhou，Untrusted Hardware Causes Double-fetch Problems in the I/O Memory, Journal of Computer Science and Technology, 2018, 33(3): 587~602 .

(4) **Pengfei Wang** (#)(*), Kai Lu, Gen Li, Xu Zhou, A Survey of the Double-Fetch Vulnerabilities, Concurrency and Computation: Practice and Experience, 2018, 30(8):e4345.

二、会议论文

(1) Bo Yu(#), **Pengfei Wang**(*), Tai Yue, Tong Tang, Poster: Fuzzing IoT Firmware via Multi-stage Message Generation, 26th ACM Conference on Computer and Communications Security (CCS' 19), London, UK, 2019.11.11-2019.11.15.

(2) Yingqi Luo(#)(*), **Pengfei Wang**, Xu Zhou, Kai Lu, DFTinker: Detecting and Fixing Double-fetch Bugs in an Automated Way, The 13th International Conference on Wireless Algorithms, Systems, and Applications (WASA2018), Tianjin, China 2018.6.20-2018.6.22

(3) **Pengfei Wang** (#)(*) , Jens Krinke, Kai Lu, Gen Li, Steve Dodier-Lazaro, How Double-Fetch Situations turn into Double-Fetch Vulnerabilities: A Study of Double Fetches in the Linux Kernel, The 26th USENIX Security Symposium, Vancouver, 2017.08.16-2018.08.18.

(4) Kai Lu (#), **Pengfei Wang**(*), Gen Li, Xu Zhou, What if Hardware is Untrusted? A Study of the Hardware Double Fetch Problem in Linux Kernel (Poster), 26th USENIX Symposium, Vancouver, 2017.8.16-2017.8.18.

附件信息

序号	附件名称	备注	附件类型

项目名称： 大规模分布式并行fuzzing关键技术研究

资助类型： 面上项目

申请代码： F0205.网络与系统安全

国家自然科学基金项目申请人和参与者科研诚信承诺书

本人**在此郑重承诺**：严格遵守中共中央办公厅、国务院办公厅《关于进一步加强科研诚信建设的若干意见》规定，所申报材料和相关内容真实有效，不存在违背科研诚信要求的行为；在国家自然科学基金项目申请、评审和执行全过程中，恪守职业规范和科学道德，遵守评审规则和工作纪律，杜绝以下行为：

（一）抄袭、剽窃他人科研成果或者伪造、篡改研究数据、研究结论；

（二）购买、代写、代投论文，虚构同行评议专家及评议意见；

（三）违反论文署名规范，擅自标注或虚假标注获得科技计划等资助；

（四）购买、代写申请书；弄虚作假，骗取科技计划项目、科研经费以及奖励、荣誉等；

（五）在项目申请书中以高指标通过评审，在项目计划书中故意篡改降低相应指标；

（六）以任何形式探听尚未公布的评审专家名单及其他评审过程中的保密信息；

（七）本人或委托他人通过各种方式及各种途径联系有关专家进行请托、游说，违规到评审会议驻地游说评审专家和工作人员、询问评审或尚未正式向社会公布的信息等干扰评审或可能影响评审公正性的活动；

（八）向评审工作人员、评审专家等提供任何形式的礼品、礼金、有价证券、支付凭证、商业预付卡、电子红包，或提供宴请、旅游、娱乐健身等任何可能影响评审公正性的活动；

（九）其他违反财经纪律和相关管理规定的行为。

如违背上述承诺，本人愿接受国家自然科学基金委员会和相关部门做出的各项处理决定，包括但不限于撤销科学基金资助项目，追回项目资助经费，向社会通报违规情况，取消一定期限国家自然科学基金项目申请资格，记入科研诚信严重失信行为数据库以及接受相应的党纪政纪处理等。

申请人签字：

编号	参与者姓名 / 工作单位名称（应与加盖公章一致）/ 证件号码	签字
1	**卢凯** / 中国人民解放军国防科技大学 / 4**********X	
2	**王鹏飞** / 中国人民解放军国防科技大学 / 3**********7	
3	**宋丛溪** / 中国人民解放军国防科技大学 / 1**********1	
4	**张根** / 中国人民解放军国防科技大学 / 5**********6	
5	**乐泰** / 中国人民解放军国防科技大学 / 4**********1	
6	**刘陈一帆** / 中国人民解放军国防科技大学 / 4**********2	
7	**尹启迪** / 中国人民解放军国防科技大学 / 3**********0	
8	**刘莹莹** / 中国人民解放军国防科技大学 / 4**********8	
9		

项目名称： 大规模分布式并行fuzzing关键技术研究
资助类型： 面上项目
申请代码： F0205.网络与系统安全

国家自然科学基金项目申请单位科研诚信承诺书

本单位依据国家自然科学基金项目指南的要求，严格履行法人负责制，**在此郑重承诺：**本单位已就所申请材料内容的真实性和完整性进行审核，不存在违背中共中央办公厅、国务院办公厅《关于进一步加强科研诚信建设的若干意见》规定和其他科研诚信要求的行为，申请材料符合《中华人民共和国保守国家秘密法》和《科学技术保密规定》等相关法律法规，在项目申请和评审活动全过程中，遵守有关评审规则和工作纪律，杜绝以下行为：

（一）采取贿赂或变相贿赂、造假、剽窃、故意重复申报等不正当手段获取国家自然科学基金项目申请资格；

（二）以任何形式探听未公开的项目评审信息、评审专家信息及其他评审过程中的保密信息，干扰评审专家的评审工作；

（三）组织或协助项目团队向评审工作人员、评审专家等提供任何形式的礼品、礼金、有价证券、支付凭证、商业预付卡、电子红包等；宴请评审组织者、评审专家，或向评审组织者、评审专家提供旅游、娱乐健身等任何可能影响科学基金评审公正性的活动；

（四）包庇、纵容项目团队虚假申报项目，甚至骗取国家自然科学基金项目；

（五）包庇、纵容项目团队，甚至帮助项目团队采取“打招呼”等方式，影响科学基金项目评审的公正性；

（六）在申请书中以高指标通过评审，在计划书中故意篡改降低相应指标；

（七）其他违反财经纪律和相关管理规定的行为。

如违背上述承诺，本单位愿接受国家自然科学基金委员会和相关部门做出的各项处理决定，包括但不限于停拨或核减经费，追回项目经费，取消一定期限国家自然科学基金项目申请资格，记入科研诚信严重失信行为数据库以及主要责任人接受相应党纪政纪处理等。

依托单位公章:
日期： 年 月 日

合作研究单位公章:
日期： 年 月 日

合作研究单位公章:
日期： 年 月 日