计算机研究与发展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202330726
61(1): 2−19, 2024

大规模复杂终端网络的云原生强化设计

李振华　王泓懿　李　洋　林　灏　杨昕磊
（清华大学软件学院　北京　100084）
（lizhenhua1983@tsinghua.edu.cn）

Cloud Native Reinforced Design for Large-Scale Complex Terminal Networks

Li Zhenhua, Wang Hongyi, Li Yang, Lin Hao, and Yang Xinlei
（*School of Software*, *Tsinghua University*, *Beijing* 100084）

Abstract　As the “last mile” of Internet content delivery, terminal networks seem rather simple but in fact constitute 99% of the performance bottlenecks. Classic design is usually oriented to typical devices and regular environments, thus making it difficult to accommodate and reproduce diversified scenarios and resulting in severe performance degradation. By comprehensively gathering and deeply diagnosing the anomalies of large-scale complex terminal networks at the cloud, we have revealed several important defects of the classic design for terminal networks in three key dimensions—availability, reliability and credibility. In order to fix these defects effectively and efficiently, the cross-layer and cross-technology collaboratively reinforced design methodology is adopted (e.g., the time-inhomogeneous 4G/5G dual connectivity management method is utilized to minimize the probability of network disconnection), so as to fulfill self-regulation mechanism design for ubiquitous scenarios. The research achievements have been applied to the high-speed network of the Ministry of Public Security, 17 million UUSpeedTest App users, 70 million Xiaomi mobile phones, 100 million Baidu PhoneGuard users, and 900 million WiFi devices. In recent years, we have also conducted forward-looking network design based on cloud-hosted emulators to discover and fix potential defects without real-world user engagement, thus making the design of terminal networks “born in the cloud and grow in the cloud”. The research achievements have been applied to Huawei DevEco Studio IDE (Integrated Development Environment), Tencent App Market, Google Android Emulator, and multiple popular Apps (like Douyin and Toutiao) of ByteDance.

Key words　terminal network; network measurement; network design; cloud native; network emulation

摘　要　作为互联网数据传输的“最后一公里”，终端网络看似简单却构成99%的性能瓶颈. 经典设计面向典型设备常规环境，难以兼顾多样化场景，导致严重性能落差. 通过云端汇聚并深度诊断大规模终端网络异常，在可用、可靠、可信3个关键维度揭示经典设计多处重要缺陷，采用跨层跨代的协同强化方法针对性修复（如时变非齐次4G/5G双连接管理方法最小化断网概率），实现无场景预设的自调控机制设计. 应用于公安部高速网络、1 700万“测网速”app用户、七千万小米手机、一亿百度手机卫士用户以及九亿WiFi设备. 近年来进一步开展基于云端模拟器的前瞻网络设计，无需真实用户设备参与即可发现并修复潜在缺陷，让终端网络设计“生于云、长于云”. 研究成果应用于华为DevEco Studio集成开发环境、腾讯应

收稿日期：2023-09-11；**修回日期**：2023-10-07
基金项目：国家重点研发计划项目（2022YFB4500703）；国家自然科学基金项目（61902211, 62202266）；微软亚洲研究院合作研究项目（100336949）
This work was supported by the National Key Research and Development Program of China (2022YFB4500703), the National Natural Science Foundation of China (61902211, 62202266), and the Microsoft Research Asia Collaborative Research Project (100336949).
通信作者：李振华（lizhenhua1983@tsinghua.edu.cn）

用市场、谷歌安卓模拟器及字节跳动多款流行应用（如抖音和今日头条）.

关键词 终端网络；网络测量；网络设计；云原生；网络模拟

中图法分类号 TP391

自 1969 年诞生以来，互联网数据传输的基本结构一直由风格迥异的两部分组成：骨干网络和终端网络. 前者包含数千万路由中继节点，由专业人员实时维护，通常能保持高速稳定；后者作为数据传输的“最后一公里”，服务于多样化动态接入场景，如 ADSL 有线电缆、WiFi 4/5/6 热点和蜂窝 2G/3G/4G/5G，对用户体验的影响最为直接，但由于规模巨大（近千亿节点）并且缺乏专业维护，很容易出现各种问题，看似简单却构成绝大多数（99% 以上）数据传输过程的性能瓶颈[1-3].

长期以来，终端网络设计一直遵循“实验研究—原型测试—规模部署—用户报修”的经典模式. 随着现代互联网愈发庞大繁杂、多源异构，经典模式拖沓沉重、弊端重重——实验研究与原型测试往往面向典型设备和常规环境，难以复现与兼顾终端复杂性，造成严重性能落差[4]，带给用户 4 个方面的困惑：

1）网速困惑. 5G 和 WiFi 6 接入网络都宣称高达 10 Gbps 的带宽和低至数毫秒的时延，然而用户实际体验到的却相去甚远[5]，甚至连我们日常拨打微信语音电话至今都经常卡顿.

2）断连困惑. 作为国家重点投资、市场热点宣传的 5G 接入网络，虽然峰值带宽很高，但信号覆盖半径小、穿透能力弱、运营成本高（不少 5G 基站夜间会自动关闭），连接中断的概率显著增加.

3）安全困惑. 用户身边通常遍布蜂窝基站和看似免费的 WiFi 热点，但很多都存在风险，比如央视 3·15 晚会就多次报道犯罪分子利用蜂窝伪基站和 WiFi 伪热点实施电信诈骗的猖獗现象.

4）代际困惑.5G 手机和流量套餐通常都比 4G 要贵，但与之匹配的应用，如自动驾驶和虚拟现实并不成熟，并且 5G 基站的总体覆盖范围小于 4G，导致很多用户对是否要升级 5G 存在疑惑.

更糟糕的是，这些困惑往往长期存在、得不到及时解决，用户怨声载道却又不明就里. 形象地说，终端网络设备一旦离开生产厂家，就像一个离家流浪闯荡的孩子，无时无刻不在面对各种未知、异常和风险.

为此，本文作者（以下简称“我们”）多年来一直致力于终端网络基础设计模式的反思与革新. 我们同终端设备制造商、网络运营商、操作系统和应用软件开发商广泛交流合作，发现想要解决大规模复杂终端网络的性能落差问题“知易而行难”. 举一个典型案例，如图 1 所示是安卓 11/12/13 操作系统（分别发布于 2020/2021/2022 年）共用的一段源代码，虽只有 10 行但意义重大，它控制着安卓设备对蜂窝网络的优先连接模式. 不难看出，当一台安卓 11/12/13 设备附近既有 5G 基站又有 4G 基站时，这段代码的处理方式非常简单：能连 5G 就连 5G，不能才降到 4G，而不考虑具体环境因素，比如信号强度或资源竞争.

```
private int getPreferredNetworkMode() {
        int networkMode = MobileNetworkUtils.getNetworkTypeFromRaf(
                (int) mTelephonyManager.getAllowedNetworkTypesForReason(
                        TelephonyManager.ALLOWED_NETWORK_TYPES_REASON_USER));
        if (!showNrList()) {
                Log.d(LOG_TAG, "Network mode :" + networkMode + " reduce NR");
                networkMode = reduceNrToLteNetworkType(networkMode);
        }
        Log.d(LOG_TAG, "getPreferredNetworkMode: " + networkMode);
        return networkMode;
}
```

缩写 Raf = Radio access family, Nr = New radio = 5G（目前阶段）

Fig. 1 Controller code for cellular network priority connected mode in Android 11/12/13 operating systems

图 1 安卓 11/12/13 操作系统对蜂窝网络优先连接模式的控制代码

如果一部安卓 5G 手机当前被很弱的 5G 信号和很强的 4G 信号同时覆盖，常识告诉我们：此时应该选择 4G 连接更为合理，不幸的是，基于图 1 中的代码，安卓会直接选择信号很弱的 5G 连接. 更糟糕的是，由于 5G 基站附近的安卓手机通常都会优先连接到它，将导致该 5G 基站网络资源被多部手机激烈竞

争，而附近广泛存在的4G基站却由于无人问津而长期空闲、浪费充裕的网络资源，我们称这一现象为“饱和饥饿”[6]：明明服务方的总体资源是充足的，但客户方却得不到或者所得资源太少.

更进一步，我们反思安卓操作系统蜂窝网络管理模块的设计者是否知道上述问题？基于常识推断，他(她)很可能是知道的. 如果确实知道，那他(她)为什么不改进呢？仔细思考这个看似矛盾的状况，我们发现：即使设计者知道，也很难修改，因为这涉及到动机、平台、资源与知识层面的四道鸿沟：

1) 动机鸿沟. 终端网络的软硬件设计者很多时候并非用户咨询和投诉压力的直接承担者，没有足够强烈的动机来解决问题. 典型案例是安卓操作系统中多个网络模块的设计所导致的问题几乎都被投诉到安卓手机制造商.

2) 平台鸿沟. 即使设计者有动机解决性能落差问题，也经常因为缺乏平台支持，无法了解丰富场景下大量终端设备的实际使用情况. 没有手机制造商的许可和帮助，即便安卓网络模块的设计者也很难洞悉用户在非预设场景下的真实体验.

3) 资源鸿沟. 假设动机和平台兼备，大规模复杂终端网络的设计者还需要承担十分可观且不确定的后端资源开销来进行长期细致的数据采集和分析. 值得注意的是，即便在利益相关的大公司内部，所需后端资源也不容易(充分)审批.

4) 知识鸿沟. 即使动机、平台、资源方面都得到支持，想要妥善解决大规模复杂终端网络的性能落差问题，设计者还需要具备硬件、信号、频谱、网络、用户心理等多维度全栈知识，这通常是最深也最隐蔽的科研鸿沟.

针对用户的四重困惑和研发的四道鸿沟，我们多年来一直和终端网络用户投诉压力的直接承担者(如小米手机研发团队、“测网速”App研发团队以及WiFi万能钥匙研发团队)紧密合作，以克服动机鸿沟. 同时，这给予我们贴近数亿终端网络设备所处丰富场景的机会，具备细致识别用户真正痛点问题的客观条件，从而克服平台鸿沟. 然而，对于剩下的资源鸿沟和知识鸿沟，工业界合作方通常并不能够提供充分的支持. 因此，我们的科研工作聚焦于大规模复杂终端网络的低开销测量分析与自调控机制设计.

一方面，针对终端网络设备规模性带来的调研成本挑战——用户上报网络异常的常规和峰值负载可能相差上万倍，利用服务器无感知的云原生基础设施，毫秒级精细取用云端资源，最小化测量分析开销，在可用、可靠、可信3个关键维度揭示终端网络经典设计的多处重要缺陷. 具体来说，面对数量可观而时空不确定的终端网络异常数据，利用云原生的微服务核心技术以及轻量级容器/沙盒载体，能够细粒度按需灵活部署及快速回收后端资源，避免传统云计算平台中频繁发生的、由于虚拟机服务器绝大部分时间“空转”所导致的资源浪费. 在云端为终端网络营造一个牢固而集约的家，让终端设备一个都不掉队.

另一方面，针对终端网络场景复杂性带来的定制优化挑战——非典型场景和非主流用户普适存在，采用跨层跨代的协同强化方法，整合多层次各方面知识，实现无场景预设的自调控机制设计. 尤其要避免经典设计和理论文献中看似合理的“震荡型”片面设计(牺牲较为隐蔽的“长尾”用户性能，成全更为显著的典型用户性能)，网络整体性能提升的同时，不损害非主流长尾用户的局部性能. 研究成果应用于公安部高速网络、三百万小米移动(虚拟运营商)用户、1 700万友声科技“测网速”App用户、七千万小米手机、一亿百度手机卫士用户以及九亿WiFi终端设备.

1 主要研究内容和创新点

如图2所示，我们的研究立足终端网络的3个关

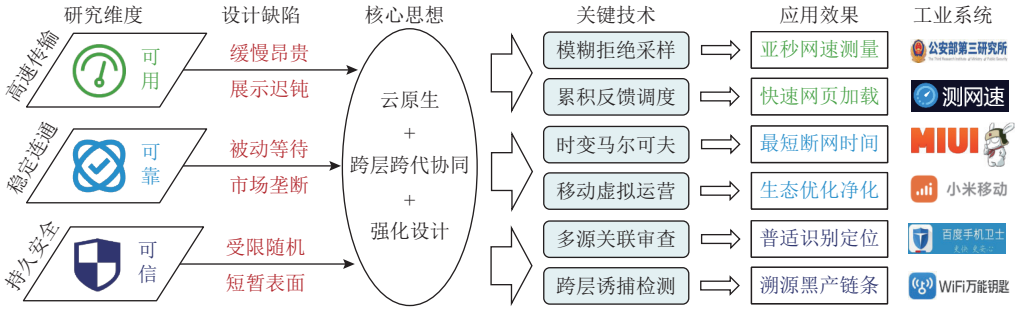


Fig. 2 Cloud native reinforced design for large-scale complex terminal networks: main research contents and innovations

图2 大规模复杂终端网络的云原生强化设计：主要研究内容和创新点

键维度：可用（高速传输）、可靠（稳定连通）和可信（持久安全），揭示经典设计的多处重要设计缺陷：1）可用性方面，终端网络测速过程缓慢昂贵，网页展示过程迟钝拖沓；2）可靠性方面，断网修复机制被动等待、无法及时连通，网络运营市场长期垄断、阻碍技术创新；3）可信性方面，危险网络接入点的检测方式受限随机、检测结果很不理想，并且即使检测成功、防御或反击策略的效果也往往短暂而表面.

针对这 3 个缺陷，我们面向大规模复杂终端网络，运用云原生强化设计的核心思想，结合（全协议栈）跨层跨代协同强化的关键理念，从广大用户体验痛点出发，以点带面渐进修复设计缺陷，最终突破网络整体性能边界，并在多个代表性工业系统中部署应用. 研究创新性主要体现在 3 个方面：

1）软硬件及人类行为因素的综合考量. 不同于一般的强化学习过程，终端网络的设计不是一个简单抽象的数学模型（往往很难以公式来形式化），涉及复杂的软硬件及人类行为因素，很多时候还要考虑工业界接受程度和对互联网整体生态系统的影响，因此需要大规模用户配合众包测量、大数据关联分析以及网络跨层协同设计.

2）云原生微服务技术的双重有效支撑. 大规模复杂终端网络的测量分析往往涉及数量可观而时空不确定的后端存储和计算资源开销，作为云原生的核心技术，微服务构建于轻量级容器或沙盒之上，如果设计合理、部署得当能够精细粒度按需灵活部署及快速回收，从而大幅度降低调研成本；同时，通过服务器无感知的开发模式，有效降低网络研究者利用云计算技术的知识门槛[7].

3）面向长尾用户体验的普适性能改进. 终端网络的经典设计面向主体用户、主流环境，其优化升级通常能够提升网络主体性能，但同时不可避免地损害部分用户局部性能，这种“牺牲小我、成全大我”的方式对现代互联网产业而言并不可取. 与此相反，我们的研究注重细致周全的终端网络设计，从长尾用户的体验痛点出发，以点带面，最终促成网络整体的完善和进化.

基于上述云原生强化设计新模式，我们对多个大规模复杂终端网络工业系统开展全面测量、深度分析和谨慎改造. 在可用、可靠、可信 3 个关键维度所研发的 6 项代表性关键技术及其应用效果简介如下：

1）提出模糊拒绝采样理论，首次实现亚秒级网速测量. 网络带宽（即网速）的准确测量是保障诸多网络应用服务质量的重要前提，而影响准确性的关键因素是信道噪音. 主流工业系统（如 Ookla Speedtest 和 Google FAST）所采用的经典测速技术致力于从时间或空间角度减少噪音，导致测量过程缓慢拖沓、流量开销高昂. 我们采取迥异于前人的解决思路：不再从时空角度减少噪音，而是从统计分析角度利用噪音. 基于计算几何学和离散控制论研发“模糊拒绝采样”双向强化测速技术，同时达成快速、轻量、准确这 3 个传统意义上看似矛盾的目标[8-9]. 此外，自适应部署和回收跨运营商的（瞬时）容器和（长时）虚拟机作为测试服务器，将平均测速时间降到 1 s 以内，即“亚秒级”[5]，同时将平均测速开销（主要是网络流量）降低为原来的 1/10. 该技术成果应用于公安部第三研究所高速网络以及拥有 1 700 万移动终端用户的友声科技“测网速”App.

2）设计累积反馈调度算法，提升网页可视内容加载速度. 绝大部分互联网流量以 Web 网页形式传输和展现，但经典网页加载方式并不契合实际用户体验，往往加载很多资源而页面并无（关键）可视内容，浪费网络流量和用户时间. 我们将可视化速度指标（speed index）融合进网页加载过程，创新设计“累积反馈调度”算法[10]，通过云端（微）服务器和客户端浏览器的协同交互，以渐进反馈方式应对网络状态、浏览器执行以及视窗大小在加载过程中的不确定性，不断累积优化网络资源的调度下载顺序，使得可视内容的展现速度提高 40% 以上，而给服务器端带来的计算开销少于 5%. 该技术成果部分应用于“测网速”App 的网页加载过程.

3）构建时变非齐次连接管理模型，最优化蜂窝网络故障的诊断和修复时机. 从 2G 到 5G，蜂窝网速越来越高、服务密度越来越高、时延越来越低，然而手机异常断网的发生概率却越来越大. 我们同小米手机操作系统（MIUI）团队合作定制安卓操作系统，连续 8 个月跟踪测量 34 个型号共计 7 000 万部小米手机的异常断网故障，总共收集到 20 亿条故障事件日志. 通过服务器无感知的日志存储和分析，发现安卓蜂窝通信模块多处不为人知的重要设计缺陷，是导致手机异常断网的主要根源. 从而构建基于“时变非齐次马尔可夫过程”的连接管理模型，在统计意义上最优化蜂窝网络故障的诊断和修复时机，将所有手机断网时长缩短 36%、5G 手机断网次数减少 40%. 对应论文[11]获得 ACM SIGCOMM 2021 会议唯一最佳学生论文奖，这也是亚洲科研院校历史上首次获颁该奖项.

4) 运用人工智能结合统计建模方法，优化与净化移动虚拟运营生态系统. 受电信资源充分利用和激发良性市场竞争的双重驱动，移动虚拟运营商(简称“移动虚商”)近年来迅速流行，为用户提供灵活优惠的入网服务. 另一方面，移动虚商的发展也面临多重挑战，比如网络性能歧视、套餐转售利润微薄、计费不准确以及诈骗电话多等问题. 我们同拥有 300 万用户的移动虚商“小米移动”合作，基于较低的预算为移动用户建立云原生月度时序数据库，运用微观人工智能结合宏观统计建模的方法，解决移动虚商所面临的一系列挑战问题[12-13]. 尤其是融合多种数据清洗和机器学习算法，准确预测用户网络流量、离网概率和风险系数[14]，帮助小米移动提高运营利润 60%、降低用户流失 61%、减少诈骗电话 94%.

5) 通过多源数据关联审查，普适识别和准确定位伪基站. 移动运营商部署合法基站的同时，不法分子也部署了大量非法伪基站，使用极高信号强度诱使移动设备错误连接，趁机发送垃圾或诈骗短信. 我们同百度手机卫士团队合作研发“伪基站雷达”系统[15]，云端收集并关联分析来自上亿手机的非隐私多维数据，平均每天识别过滤数百万条伪基站短信，定位伪基站的中值误差低至 11 m，该精度已足够支持执法机构实时跟踪伪基站的装载车辆. 伪基站雷达系统一直工作、维护和优化至今，其识别和定位结果为公安部持续提供关键信息，帮助公安干警每月抓捕伪基站犯罪人员数十名、收缴伪基站设备数百台.

6) 设计主被动结合的跨层诱捕方法，准确识别并自动溯源 WiFi 网络攻击. WiFi 热点(即 WiFi 路由器)承载移动互联网大部分终端流量，不可避免地成为各种安全威胁的攻击目标. 为全面了解全国范围的 WiFi 安全威胁，我们与拥有 9 亿用户的“WiFi 万能钥匙”App 团队合作搭建 WiFi 安全检测系统[16]. 云端定制弹性 IP 地址及合成网页内容，以个性化微服务方式对 1 900 万 WiFi 热点实施攻击行为诱捕与主被动跨层检测，从而宏观把握 WiFi 安全威胁的普遍性(至少 4% 的 WiFi 热点存在安全威胁)、风险性和技术特征. 最重要地，我们首次揭示 WiFi 攻击背后的地下黑色广告产业，并发现第三方 Web 分析平台(通常合法运营但并不知情)是其盈利链条的关键环节. 据此向国内主要 Web 分析平台检举报告，使得基于 WiFi 的广告攻击减少接近一半.

2 终端网络可用性测量与改进

用户对终端网络的第一诉求是可用(usability)，能够以应用所需要的速度接收来自互联网的数据包，并且能够看到这些数据包的直观效果. 就接收速度而言，可用性要求数据的高速传输，通常称为“高网速”或者“高带宽”. 同等带宽前提下，不同的数据传输方式，如先传谁、后传谁、怎么传则会产生不同的直观效果，这突出地表现在 Web 网页的展示上，而 Web 网页是今天绝大多数移动应用的呈现形态.

2.1 网络带宽的快速测量理论、技术及应用

网络带宽的准确测量是保障诸多网络应用服务，如微信语音、视频直播和在线会议质量的重要前提[17-18]. 网络带宽观测数据经常被政府报告、商业新闻和运营商广告所引用，直接或间接地影响网络用户的选择和决策[19]. 实际上，在新冠疫情期间，网络带宽测量(简称“测网速”)一度成为家庭网络最热门的服务，因为居家办公的网民非常关心其带宽情况[20]. 因此，移动虚拟运营商用户对带宽也十分重视，他们很担心被实体运营商区别对待而导致服务质量降级. 此外，移动网络的兴盛使得带宽随环境而剧烈变化，因而测网速服务的需求频率大幅度提升[21-22]. 安卓 11 操作系统甚至专门提供 API 帮助 5G 应用粗略评估网络带宽.

影响带宽测量准确性的关键因素是信道噪音，它来源于拥塞控制和信道共享等诸多因素. 长期以来，主流工业系统(如 Ookla Speedtest、Google FAST 和友声科技“测网速”App)所采用的带宽测量技术均致力于从时间或空间的角度来降低信道噪音的影响，要么像 Google FAST 那样延长测量时间直到噪音变得很弱，要么像 Ookla Speedtest 那样使用空间邻近的测量服务器(全球范围已经分布式部署 16 000 多台服务器，几乎能给任何一个测速终端分配一台)以直接避免绝大多数噪音. 前者使测量过程缓慢拖沓，后者使测量系统成本高昂，并且这些缺点伴随高速无线网(如 5G 和 WiFi 6E)的普及而急剧恶化. 如果一台 5G 手机当前下行带宽约 1 Gbps，那么使用 Speedtest 测速一次就需要持续 15 s、消耗 1.4 GB 左右的蜂窝流量——前者尚可忍耐，后者极难接受，因为 1.4 GB 蜂窝流量一般需要花费好几块钱！

针对这一领域瓶颈问题，我们采取迥异于前人的解决思路：不再从时间或空间的角度降低或避免

噪音，而是从统计分析的角度容纳和利用噪音. 该创新思路来自一个关键发现：虽然噪音经常导致带宽瞬时采样的显著波动，却几乎从不影响带宽关键区间的存在和位置. 如图 3 所示，在测速过程的任一时刻，带宽的关键区间都包含数量众多的采样点，并且这些采样点密集地聚拢在一起（V_x 和 V_y 之间）. 随着时间的推移，新出现的采样点可能上下跳跃，甚至成为离群值（outlier），但新计算的关键区间则只会微调，即便是离群值的出现也只会反向强化关键区间的稳定.

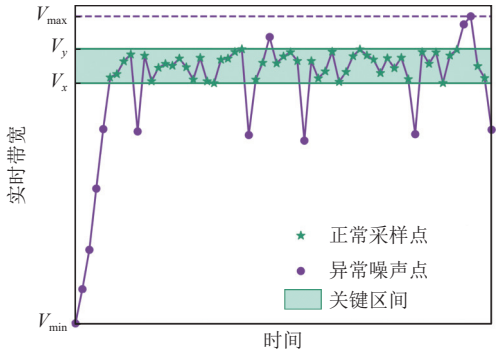


Fig. 3 Existing sampling points of the bandwidth at a certain time as well as their crucial interval

图 3 某一时刻带宽的现存采样点及其关键区间

在上述创新思路和关键发现的指导下，我们基于计算几何学和离散控制论研发“模糊拒绝采样”[8-9]双向强化带宽测量技术：正常采样点被接受，正向强化关键区间；异常噪音点被拒绝，反向强化关键区间；关键区间往往在测速刚开始的几秒就已经十分明显，并不需要拖沓冗长地再测十几秒.

由于我们的带宽测量技术对噪音不敏感，并不需要距离用户终端很近的测速服务器，因此可以使用廉价弹性的公有云虚拟机和容器构建测速（微）服务器池，在几乎不影响测速性能的前提下，能够将经典设计的基础设施开销降低数十倍. 具体来说，如果测网速系统服务用户较少，则完全使用容器微服务、按需瞬时部署测速服务器，测试完毕马上回收；反之，如果测网速系统服务用户较多且地理位置随机，则更适合使用常驻虚拟机作为测速服务器，并且尽量分散部署在核心互联网流量交换点①（Internet eXchange points, IXP）附近，从而兼顾成本和性能.

通过模糊拒绝采样结合云原生测试服务器部署，能够同时达成快速、轻量、准确这 3 个在传统意义上看似矛盾的目标. 基于 Web 的技术实现，其架构如图 4 所示，能将网络带宽的测量时间从主流工业系统的 10~30 s 降低到 3~5 s，相关代码全部开源②. 还部署了在线原型系统③一键测速，对应技术应用到公安部第三研究所的高速网络带宽检测工作中，取得预期效果：平均测量时长为 2.4 s，流量消耗为 1.2 GB；相比 Ookla SpeedTest，平均测量时长缩短 6.2 倍，流量消耗减少 9.1 倍.

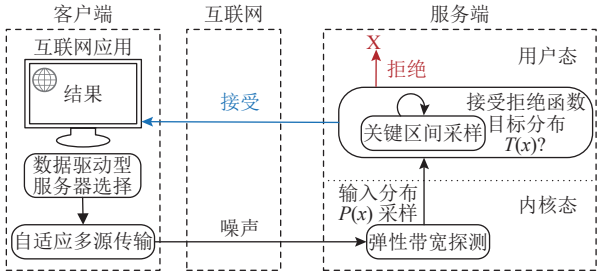


Fig. 4 Architecture of Web-based Fast BTS system based on fuzzy rejection sampling

图 4 基于模糊拒绝采样的 Web 快速轻量带宽测量系统架构

更进一步，基于移动 App 的技术实现应用到国内最大规模的“测网速”App④，绝大部分情况下能在亚秒级（即少于 1 s）时间内准确测量移动终端的网络带宽[5]，相关代码和数据已开源⑤.

在“测网速”App 研发团队的大力支持下，我们顺利邀请到 354 万移动手机用户开展为期 4 个月（2021 年 8 月至 11 月）的“深度”带宽测量，总共收集到 2 360 万条深度测量数据，每一条对应一次带宽测量所涉及到的物理层、链路层、网络层、传输层和应用层的丰富信息，但不包含任何用户隐私数据. 从这些数据中我们发现一个令人惊讶和沮丧的事实：从 2020 年和 2021 年，尽管 WiFi 6 路由器和 5G 基站被不断部署，WiFi 数据传输的平均带宽基本保持不变，2020 年为 132 Mbps，2021 年为 137 Mbps，4G/5G 数据传输的平均带宽甚至还下降了，2020 年分别为 68 Mbps 和 343 Mbps，2021 年分别为 53 Mbps 和 305 Mbps.

仔细分析各项深度信息，我们发现 WiFi 带宽主

① 以中国大陆为例，存在 8 个核心 IXPs，分别位于北京、上海、广州、南京、沈阳、武汉、成都和西安.

② https://FastBTS.github.io

③ http://FastBTS.thucloud.com

④ http://uuspeed.uutest.cn

⑤ https://MobileBandwidth.github.io

要受制于其上游固定宽带的接入带宽,而接入带宽最近几年停滞不前.4G带宽下降主要因为它原先拥有的3个高带宽频段被“重耕”给5G使用,但这些频段对5G来说又过于狭窄、不够高速.此外,还发现建筑物林立的城市中心区域虽然密集部署了大量5G基站(因此手机端接收信号很强),但由于建筑物多径干扰、用户之间负载均衡以及密集基站之间糟糕的切换问题,强信号却往往对应较低的接入带宽[23].上述根因分析基本上解释了本文开头提到的“网速困惑”.

2.2 面向可视内容的快速网页加载技术研究

绝大部分互联网流量以Web网页形式传输和展现,但经典的网页加载方式并不契合实际用户体验,往往加载很多资源而页面并无关键可视内容,浪费网络流量和用户时间,如图5(a)所示.为此,我们将近年来广受关注的可视化速度指标引入网页加载过程,以量化页面被可视内容填充的速度(当然越快越好),如图5(b)所示.虽然速度指标的直观效果简单清晰,但其积分计算方式却十分复杂,所以此前一直被学术界和工业界用来回顾式地评估已发生的网页加载过程[24-29],并不适合用做显式启发来指导即将发生的网页加载过程.



(a) 原始顺序加载元素



(b) 优先加载视觉关键元素

Fig. 5 Different loading processes of a typical web page (CCTV homepage) under the same network environment

图5 同等网络环境下一个典型网页(央视网主页)的不同加载过程

深度分析使用速度指标启发指导网页加载调度的实际困难,我们发现其主要来自3方面不确定性:1)网络环境;2)浏览器执行环境;3)用户端视口(viewport).首先,使用同一设备加载相同的静态页面,在不同的网络环境下会产生显著的速度指标差异.事实上,即使此操作是在非常类似的受控网络环境下重复执行,仍会出现相当大的速度指标差异,原因在于为加载页面而分配的计算资源可能随时间而变化.此外,考虑到不同的用户端视口大小,即便同一页面的关键元素也是不确定的.即便客户端知道视口大小,很多时候也无法执行速度指标最优化的网页加载调度,因为在网页加载完成之前它并不知道哪些元素是“重要”的,特别是许多现代网站采用“流动式”布局来自动调整网页元素的大小和位置.这些不确定性使得我们不可能提前或一次性地获得速度指标最优化的网页元素加载调度方案.

为克服上述困难,我们从经典分布式调度理论出发,结合速度指标的积分特性,创新设计“累积反馈调度”算法[10],以微服务渐进反馈方式应对网络环境、浏览器执行环境以及视窗大小在网页加载过程中的不确定性,不断累积优化网络资源的调度下载顺序,从而接近最大速度指标.首先为网页在不同窗口下创建预加载微服务,离线提取元素位置和资源依赖并保存;客户端访问网页时,向服务端查询可见元素.随后,为每个客户端创建专有的传输微服务负责双端通信,客户端据此利用依赖信息生成基本的资源加载顺序.传输微服务和客户端协同监听网络和JavaScript引擎的不确定事件,实时调整以接近最优加载顺序.相比传统整体式架构,基于微服务的调度架构可以利用双端信息更准确地调控加载过程,并且独立于网页原有服务,不会对Web系统产生整体性影响.

多主流网站、多网络场景下的实验结果表明:可视内容的展现速度能提高40%以上,介于30%到50%之间,平均41%.创新调度算法运行1次平均只需要12 ms,仅为网页加载时间的0.3%,具体时间复杂度为$O(n\cdot\log n)$、其中n表示网页中的元素数目.给服务器端带来的额外计算开销通常少于5%,具体内存开销通常低于400 MB,而具体网络流量开销一般在12 KB左右.值得一提的是,我们所开发的“累积反馈调度”协作模块适用于当前所有主流Web(微)服务器,并可通过简单配置动态加载,相关的所有代码和数据均已开源①.

值得一提的是,上述快速网页加载技术已经部分应用于“测网速”App的网页加载过程,视觉效果上进一步加速了2.1节所述网络带宽的快速测量过程.

① https://SipLoader.github.io

3 终端网络可靠性测量与改进

解决了终端网络的可用性问题，用户基本上能够体验到期待的数据传输性能，此时网络设计者的研究通常会朝着纵向或横向继续努力，纵向努力进一步提高传输带宽或降低传输时延[30]，横向努力则致力于大规模扩展或大面积覆盖[31]. 然而，在终端网络领域真实的工业实践中，当基本可用性得到保证之后，用户的诉求往往并不在于纵向或横向的无止境提高，而是更关注一个隐藏维度——可靠性（reliability），即可用性的长时间大面积保障. 一种带宽时延极优但时不时断网、无故扣费甚至被恶意攻击的网络服务，对比另一种带宽时延能接受、同时稳定规范安全的网络服务，对于普通终端用户来说，几乎都会选择后者——并不夸张地说：“网络不可靠，则性能无意义！”

3.1 蜂窝网络可靠性的全国性测量、分析及增强

人们每天都使用手机上网，从 2G、3G、4G 到 5G，网速越来越高、时延越来越低、单位覆盖面积的服务密度越来越大[32]. 然而，根据小米手机团队近年来的运维经历，手机异常断网的发生概率却越来越高. 这里“异常断网”指的是手机有信号且基站未过载、但依然无法接入互联网，导致蜂窝连接十分不可靠，小米手机用户投诉和吐槽激烈. 具体来说，安卓操作系统将蜂窝异常断网故障分为 3 类.

1）数据连接错误（Data_Setup_Error）. 用户设备可以接收到附近基站的信号，却无法和它建立数据连接，并且该基站当前并未过载、还有服务能力.

2）无服务（Out_of_Service）. 用户设备和基站成功建立数据连接，但收不到蜂窝数据.

3）数据阻塞（Data_Stall）. 用户设备可以从基站接收到数据，但突然发生长时间停滞：有发出的数据包、却没有接收的数据包；如果停滞时间超过 1 min，安卓就认为发生了数据阻塞故障.

为了深度理解并有效解决该痛点问题，我们同小米手机团队合作，基于定制化 MIUI 操作系统（其架构如图 6 所示）连续 8 个月跟踪测量 34 个型号共计 7 000 万部小米手机的异常断网事件. 和小米手机（而不是苹果手机、三星手机或者华为手机）研发团队开展合作并非偶然，而是受到本文开头所述“动机鸿沟”的强烈驱使：苹果手机如果异常断网，用户通常认为是移动运营商的问题，因为苹果手机昂贵“高端”；华为手机如果异常断网，用户一般认为是鸿蒙操作系统的问题，毕竟鸿蒙相比安卓还很不成熟；但小米手机如果断网，大多数用户都倾向认为是手机本身的问题，因为小米手机向来廉价“低端”，不得不成为用户投诉压力的直接承担者，这只“替罪羊”有足够强烈的动机来分析和解决蜂窝网络的可靠性问题.

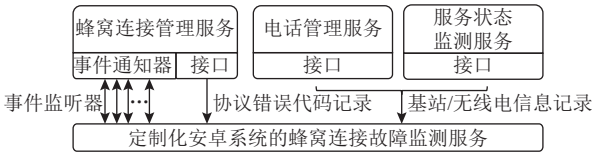


Fig. 6 Architecture of the monitoring infrastructure for cellular connection failures in the customized MIUI Android operating system

图 6 定制化 MIUI 安卓操作系统的蜂窝连接故障监测架构

7 000 万小米手机 8 个月内总共向 MIUI 系统后台上报了 20 亿条异常断网事件，这个数字（20 亿）看上去很大，但平摊到每部手机则很小：每部手机平均每天仅异常断网 0.12 次，对于 MIUI 系统后台属于相对较小的日志数据，所以后台并没有给我们分配专门的日志数据库和分析服务器，只提供服务器无感知的日志存储和分析功能. 这对我们来说既是挑战又是机遇：挑战在于需要仔细设计异常断网数据的分析逻辑，避免时间复杂度太高；机遇则在于无需维护任何软硬件基础设施，将所有精力聚焦于分析逻辑本身.

通过对异常断网事件大数据的综合分析，我们发现实际情况比用户吐槽上报的还要糟糕：虽然每部手机平均每天仅异常断网 0.12 次，但一旦发生异常断网，其平均持续时间则长达 3.1 min：想象一下如果是义务急救或者火灾报警等极端场景，3 min 的网络断连意味着多大的风险和损失！更重要的是，将表观现象和系统源代码关联到一起，我们发现安卓操作系统的蜂窝网络管理模块中存在多项不为人知的重要软件设计缺陷，是损害蜂窝连接可靠性、导致手机异常断网的主要根源；反过来，小米手机本身的软硬件设计基本上不会加剧异常断网的发生. 一个代表性案例如本文开头图 1 所示，安卓 11/12/13 操作系统盲目优先 5G 连接、完全不考虑附近可用的 4G 连接以及 5G 资源的竞争程度，导致“饱和饥饿”现象[6,33].

另一个代表性案例如图 7 所示，每当安卓操作系统探测到终端设备发生数据阻塞故障时，将启动三阶段恢复机制来解决：首先清除当前建立的数据连

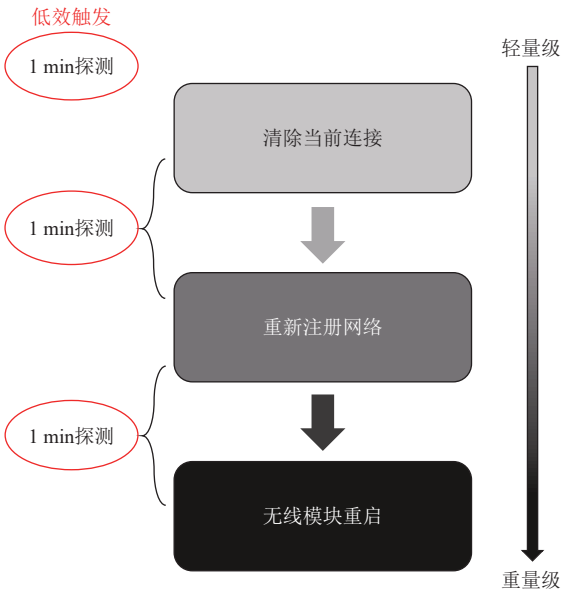


Fig. 7 The three-stage recovery mechanism for cellular data stall in the Android operating system

图 7 安卓操作系统对蜂窝数据阻塞的三阶段恢复机制

接, 然后等待 1 min; 如果故障未能解决, 则重新注册蜂窝网络, 可能会改变基站甚至运营商的选择, 再等待 1 min; 如果故障还是未能解决, 则重启整个无线模块. 三阶段所采取的解决措施从轻量级到重量级, 直观地模仿了普通用户处理数据阻塞故障时的三阶段行为: 首先重启网络应用, 然后重置蜂窝连接, 实在不行就重启手机(自然重启整个无线模块), 绝大多数情况下效果显著, 但耗时较长, 因为它没有考虑到普通用户在三阶段中的耐心递减——根据小米手机团队的用户调研, 几乎没有人能等待 3 min 之久, 用户希望能合理提前触发每个阶段, 不必等满 1 min.

针对上述问题, 我们对经典的马尔可夫过程实施动态化改造, 让状态转移概率随时间的流逝而逐渐增大, 从而提出基于“非齐次时变马尔可夫过程”的最优化方法, 将蜂窝连接修复的传统被动等待策略革新为自适应的主动触发策略, 从而最小化手机断网持续时间. 具体来说, 图 7 所示的三阶段恢复机制本质上是一个时间确定的状态转换过程, 其步骤合理但等待时间固定, 导致每一步骤均被低效触发. 为此, 我们不对 3 个阶段等同看待(非齐次), 基于收集到的数据阻塞故障恢复日志, 用马尔可夫过程重新建模整个恢复机制, 从而根据实际情况自适应主动提前触发每个阶段, 既不拖延又不仓促. 该方法在 2 800 万部小米手机上真实部署应用, 我们观察到所有手机异常断网的持续时间缩短了 36%.

针对“饱和饥饿”问题, 我们基于收集到的数据连接错误和无数据连接故障日志, 量化在各种不同信号强度下(0 格至 5 格)5G 和 4G 接入互相切换所导致的异常断网概率变化, 从统计意义上最优化 5G 终端设备的 5G/4G 接入选择. 考虑到统计意义最优不代表单次选择最优, 还引入稳定性兼容的 4G/5G (控制平面[34])双连接机制[35], 在不降低数据传输率的前提下, 实现蜂窝连接的高可靠平滑切换. 该方法同样在 2 800 万部小米手机上真实部署应用, 将 5G 手机异常断网的次数减少了 40%.

上述研究工作所对应的论文[11]的相关测量、分析和增强代码均已开源①. 值得一提的是, 该论文被录用后的指导评审人(Shepherd)是 ACM SIGMOBILE 主席、微软亚洲研究院副院长邱锂力教授, 她为本文撰写了近 600 字的公开评审意见②, 认为我们的工作“非常及时, 是迄今最大规模的蜂窝测量研究. 为蜂窝网络特别是 5G 的发展做出多方面重要贡献, 包括有用的蜂窝断网连续监控工具、多方面令人惊奇的科学发现以及行之有效的解决方案. 是 SIGCOMM 2021 会议最高分论文之一, 为优秀的网络测量研究树立了榜样.”

3.2 虚拟运营商的性能特征化、服务优化与生态净化

受电信资源充分利用和激发良性市场竞争的双重驱动, 移动虚拟运营商(简称“虚商”)近年来迅速流行[36], 依靠传统电信运营商(简称“实商”)提供的网络基础设施, 为用户提供更加灵活优惠的上层服务. 另一方面, 虚商发展也面临多重挑战, 比如网络性能歧视、套餐转售利润微薄、计费不准确[37]、缺乏实体店认证[38](因此诈骗号码很多)等. 如图 8 所示,

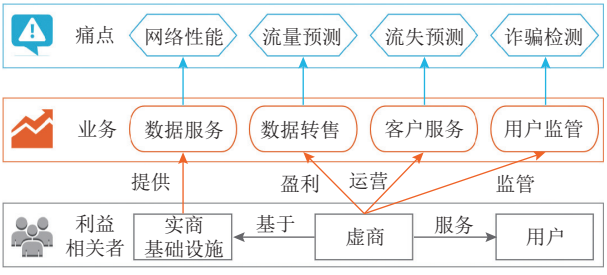


Fig. 8 Ecosystem architecture of a typical light mobile virtual network operator

图 8 典型的轻型移动虚拟网络运营商所处的生态系统架构

① https://CellularReliability.github.io

② https://dl.acm.org/doi/10.114 5/3 452 296.347 290 8

虚商处在一个复杂而庞大的生态系统中，需要考虑上下游客户和处理多种业务，尤其是关注用户的痛点投诉. 到目前为止，这一新兴市场的相关文献很少，切实可用的解决方案更少，但其潜力是令人期待的.

针对上述挑战，我们同拥有 300 万用户的商业虚商“小米移动”①深度合作研究，通过大数据分析深入理解商业虚商生态系统各方面的关键特征，并采用云端机器学习结合统计建模的方法解决虚商痛点问题. 具体来说，为合理降低预算，将采集到的用户话单计费数据预处理为月度时序数据；为保证强一致性、高可靠性与可扩展性，基于小米云平台建立存储与计算分离、读写分离的云原生月度时序数据库. 其各项服务相互独立的特点也提高了用户数据的安全性. 此外，云原生支持数据库管理和部署过程的灵活编排，便于简洁高效的查询、更新与分析.

针对网络性能痛点，通过大规模采样测量澄清了虚商客户所反馈的带宽时延歧视问题，发现小米移动用户接入互联网数据服务的上行/下行带宽、端到端时延与实商用户并无本质差别，所谓歧视主要来自新闻媒体的误导宣传以及用户因此形成的心理暗示[12].

针对流量预测痛点，考虑到虚商从实商批量购买网络流量时仅部分套餐有折扣(大部分套餐以原价购入)，虚商需要对用户每月使用的流量进行精准预测，才有可能在转售流量套餐时盈利. 通过分析小米移动用户的月度流量数据，发现绝大部分用户“两极分化”——要么流量套餐大量剩余，要么严重超出套餐额度. 此外，还发现不少流量异常(由系统计费 bugs 导致)和缺失现象(用户入网时间过短导致). 基于这些基本观察，我们调研并实验对比了多种数据异常检测算法和缺失插值算法，最终采用格鲁布斯检验法结合近邻平均插值法开展有效的数据清洗[13]. 对于清洗后的高质量数据，我们使用 SVR-RBF 机器学习算法结合统计不确定建模方法(利用多用户宏观模型修复个别用户的机器学习预测误差)，将预测准确率提升到 93.3%，最终使得小米移动的盈利提高了 60%.

针对流失预测痛点，对小米移动来说，维护一位老客户的费用相比获取一位新客户的成本要低很多，因此需要准确预测本月最可能流失(也称“离网”)的用户来提前挽留. 我们首先采用随机森林方法达到 94.45% 的召回率和 95.52% 的精度(召回率比精度更重要)，也就是说仍然存在 5.55% 的离网用户遗漏. 通过对遗漏用户数据的仔细分析，发现错误预测主要是由数据集的类别不平衡导致，于是调研并实验对比了多种基于重采样的不平衡数据集处理方法[13]，发现采用“单边选择”方法[39] 能够将离网预警模型的召回率提高到 97.8%、精度提高到 96.3%，最终使得小米移动的用户流失率降低了 61%.

针对诈骗检测痛点，很多不法分子利用在线身份认证的漏洞购买虚商 SIM 卡拨打诈骗电话，导致虚商用户中诈骗号的比例高达 0.37%，远大于实商的 0.1%. 一开始，我们直接求助实商成熟的监管服务来获取诈骗号码列表，但由于虚商缺少两类关键数据：实体店现场身份认证数据和用户信令建立数据，导致实商的监管服务无能为力. 实际上，虚商基本上只能获得通信会话层信息，无论用什么机器学习方法，识别效果都不尽如人意. 另一方面，通过仔细分析被标注诈骗用户的表层行为和深层动机，发现两者存在强烈关联——诈骗行为几乎总是受(潜在)暴利的驱动才可能发生，是有组织、按计划、成规模的. 从而以攻击经济学为基础提出利用诈骗电话时空特征的新型检测方法[14]，取得意想不到的效果——将虚商的诈骗用户比例大幅度降低 16 倍(低至 0.023%)，远低于实商在充分信息前提下所达到的 0.1%.

在避免上述方案被破解的前提下，相关代码和数据已部分开源②. 值得一提的是，在 3.1 节我们对小米手机蜂窝连接可靠性的深度研究和事实澄清，也有效帮助到小米移动团队解决虚商客户所反馈的带宽时延歧视问题.

4 终端网络可信性测量与改进

第 3 节我们说过：“网络不可靠，则性能无意义！”，本节我们继续套用这一句式：“网络不可信，则性能副作用！”一个网络一旦被黑客入侵甚至控制，其网络性能越高(高可用)、连接越稳定(高可靠)，对用户产生的危害越大. 今天的终端设备接入互联网主要通过蜂窝基站和 WiFi 热点，所以下文我们聚焦于恶意的蜂窝基站和 WiFi 热点两方面研究工作.

4.1 蜂窝伪基站的大规模检测、定位与公安协助

近年来全球移动电信产业蓬勃发展，运营商不

① https://10046.mi.com

② https://mvno-optimization.github.io

断部署 4G/5G 蜂窝基站提升服务性能. 同时，也出现了大量的非法伪基站设备，通过使用极高的信号强度，诱使周围的移动设备断开与合法基站的连接，从而连接到伪基站，趁机向用户发送垃圾或诈骗短信，并且往往使用权威号码(如 10086 和 95588)使用户放松警惕、容易上当受骗[40]. 据报道，我国手机用户接收到至少数百亿条伪基站短信，造成经济损失数百亿元.

伪基站之所以成为可能，是利用了 GSM(2G)蜂窝网络协议简单而著名的“单向认证”漏洞——只要求基站认证用户、却不要求用户认证基站. 虽然之后的 3G/4G/5G 蜂窝网络协议都改成了双向认证，但由于 2G 基站历史悠久、数量庞大且覆盖范围最广，终端蜂窝设备通常都保留 2G 通信模块以备不时之需. 实际上，即使终端蜂窝设备附近存在可以连接的 3G/4G/5G 基站，“高端的”伪基站依然能够通过发送干扰信号强迫这些设备以 2G 模式连接到自身.

为及时有效地检测伪基站、同时保护手机用户的安全，我们同拥有上亿手机用户的“百度手机卫士”① 团队合作，研发大规模伪基站识别与定位系统“伪基站雷达”[15]. 如图 9 所示，通过云端收集并关联分析来自上亿手机的非隐私多维数据建立(短信分类)大语言模型，能够利用 SVM 机器学习集群和可信电话号码列表快速准确识别伪基站短信. 此外，基于用户报告的消息日志、蜂窝基站位置数据库和 WiFi 热点位置数据库开展内容无关分析，特别是信号强度检查、基站 ID 合法性检查、基站-WiFi 热点地址一致性分析以及基站交接速度评估，也能够识别绝大多数(98%)伪基站短信，同时最大程度地保护用户隐私.

伪基站雷达系统平均每天为用户识别过滤数百万条伪基站短信(用户上报的可疑短信中至少 4.7%来自伪基站)，识别准确率高达 99.95%，定位伪基站的中值误差仅为 11 m，该精度已足够支持执法机构实时跟踪伪基站的装载车辆. 该系统一直工作、维护和优化至今，其识别和定位结果为公安部持续提供关键信息支持，帮助公安干警每月抓捕伪基站犯罪人员数十名、收缴伪基站设备数十至数百台.

4.2 WiFi 安全的全国性测量、黑产溯源及平台举报

WiFi 热点(也称 WiFi 路由器)承载移动互联网 75% 以上的“最后一公里”流量，同时不可避免地成为各种安全威胁的攻击目标. 犯罪分子利用 WiFi 热点的安全漏洞来窃听无线流量，通过 DNS 劫持发起钓鱼攻击，甚至通过挖矿劫持直接获利. 实际上，攻击者不仅会操控他人的 WiFi 热点，还会自行部署完全以攻击为目的的 WiFi 热点. 目前，基于 WiFi 的攻击已经成为全国性的安全威胁(央视 3·15 晚会多次报道)，损害数亿终端用户的上网体验和安全隐私.

为全面了解全国范围的 WiFi 安全威胁，我们与拥有 9 亿用户的 WiFi 热点发现和管理应用“WiFi 万能钥匙”②合作，搭建 WiFi 安全检测系统[16]. 如图 10 所示，以个性化微服务方式实施攻击行为诱捕与主被动跨层探测——云端定制弹性 IP 地址(其中大部分被故意设置为不可达地址，只有恶意 WiFi 热点才会有所反馈)及合成网页内容(其中隐藏防篡改签名，从而准确检测恶意 WiFi 热点对网页内容的更改)，每台 WiFi 热点设备被检查的方式及相关参数不尽相同，取决于具体设备配置和网络环境.

WiFi 安全检测系统 6 个月内从 1 400 万用户设

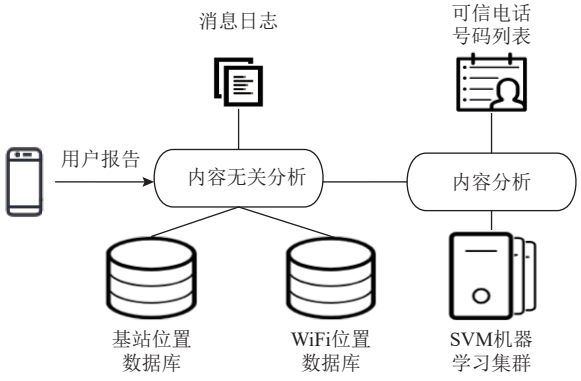


Fig. 9 Architecture of FBS-Radar system integrated by Baidu PhoneGuard

图 9 百度手机卫士集成的“伪基站雷达”系统架构

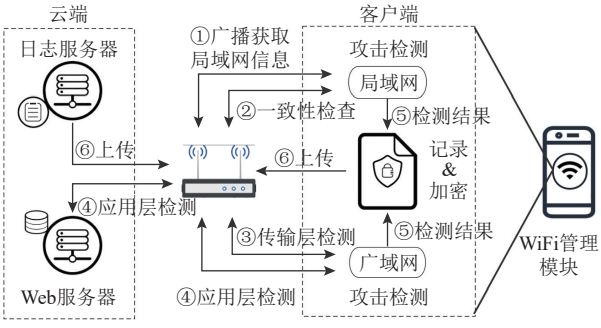


Fig. 10 Architecture of WiFi security detection system integrated by WiFi Master

图 10 “WiFi 万能钥匙”集成的 WiFi 安全检测系统架构

① https://shoujiweishi.baidu.com

② https://www.wifi.com

备上收集到 1 900 万个 WiFi 热点的安全检查数据，使得我们能宏观把握 WiFi 安全威胁的普遍性（至少 4% 的 WiFi 热点存在安全威胁）、风险性（90% 以上 HTTPS 网页依然能被劫持）和隐蔽性（插入网页广告的概率仅为 17%）. 更重要的是：首次揭示大多数 WiFi 安全攻击由地下黑色广告产业驱动，而第三方 Web 分析平台是其盈利链条瓶颈. 因此，我们向国内主要 Web 分析平台举报地下黑色产业，使得基于 WiFi 安全攻击发出的广告减少接近一半.

虽然终端网络的存在形态多种多样，但对普通用户日常生活影响最大的只是蜂窝网络和 WiFi 网络，蜂窝网络侧重服务室外环境，WiFi 网络侧重服务室内环境，两者最容易被安全攻击的环节都是互联网接入点. 4.1 和 4.2 节分别面向蜂窝基站和 WiFi 热点的安全保护，将两者结合到一起，大幅度提升互联网接入点的可信性.

5 基于云端模拟器的终端网络前瞻设计

前面 4 节所讲述的研究工作，虽然都在大规模工业系统中取得显著效果，但是都需要真实用户设备的参与，导致研究门槛较高、不利于其他研究者重现或复用. 更大的限制在于“后动性”：总是等到终端网络实际部署应用之后才能发现其中的设计缺陷，此时已有相当一部分用户设备已经承受了这些缺陷所带来的不利影响，并且某些缺陷即使发现也很难修复.

针对上述两方面限制，我们近年来一直构思“前动性”的研究方案，依然运用云原生思想进行终端网络的强化设计，但研究客体从真实用户终端设备转变为虚拟化终端模拟器. 相比真实终端设备，终端模拟器成本低、易维护、扩展性强，云计算数据中心部署到地球的哪个角落，终端模拟器就能跟随部署到哪里，而且也能接入到特定终端网络（已有不少云平台提供网络接入的定制服务），从而快速便捷地模拟出地理分散的大规模多样化终端网络场景，让终端网络的设计“生于云、长于云”.

主要研究内容和创新点如图 11 所示，从云端模拟器的 3 个核心属性（效率、兼容、安全）出发，首先研究能让每个属性达标的经典技术当前所具备的优缺点. 初步考察发现：每种经典技术目前都存在较大设计缺陷，难以匹配终端网络前瞻性研究的应用场景，尤其是极端终端环境（比如低端个人电脑）和应用类型（比如重型移动应用）的需求. 具体地，针对直接安卓模拟技术强人工、低健壮的缺陷，研发间接图形映射的创新解法，使得重型三维安卓应用也能在低端个人电脑上流畅运行. 针对动态二进制翻译技术低保真、弱封闭的缺陷，探索云原生软件蜜罐的移动场景适配，从而支持物联网终端的前瞻性设计. 针对常规虚拟化执行易暴露、常循环的缺陷，重点关注智能事件生成机制，使其能在移动应用市场规模化部署和维护. 下文简要介绍每个方面的前期工作基础和近期工作重点.

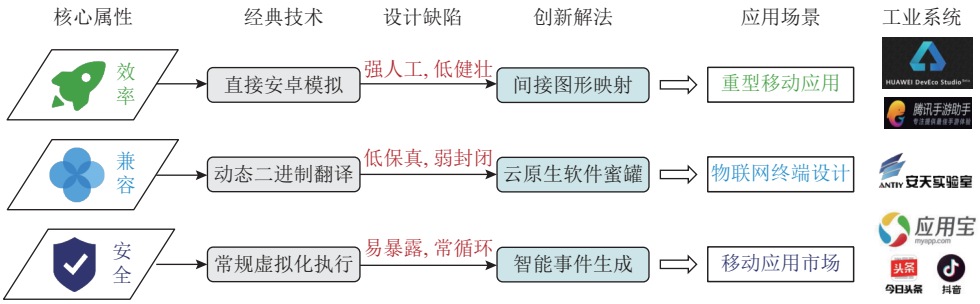


Fig. 11 Proactive design for terminal networks based on mobile emulators: main research contents and innovations

图 11 基于移动模拟器的终端网络前瞻设计：主要研究内容和创新点

5.1 高性能移动模拟器的创新设计与工业应用

移动模拟器让移动操作系统和应用无缝运行在个人电脑和服务器之上，对整个移动生态起基础性支撑作用. 过去二十多年中，为适应强交互、重渲染的应用场景，各种工业移动模拟器要么选择叠加高硬件兼容、强安全的图形渲染框架和虚拟层（如 VMware 和谷歌安卓模拟器），要么抛弃抽象、直接模拟（如腾讯手游助手），因此始终是在模拟效率、安全和兼容 3 个关键维度折中妥协.

3 个维度中最难提升的是模拟效率，因为移动设备和桌面设备在 CPU 指令集、操作系统和互动控制 3 个方面都存在巨大差异. 经典的动态二进制翻译[41-42]、类虚拟化[43-44]、硬件辅助虚拟化[45-46] 技术很难填补这道鸿沟. 强人工定制化的库函数虚拟化技术（很多人

认为这属于非虚拟化技术)理论上有能力填平鸿沟，但实际上由于人为性太强、工程量太大，只能部分填补、无法面向普适场景，比如微软公司著名的第一代 Windows Subsystem on Linux(简称 WSL)[47-48]，一直停留在命令行黑窗口阶段，无法支持图形化移动应用，实际上即使是命令行支持也不全面.

移动模拟效率的突破首先发生在移动游戏场景. 随着移动设备的普及，移动游戏越来越流行，已经成为游戏市场的主体. 相比于移动设备，个人电脑具有较大的显示屏、较高的分辨率以及更精准的键盘/鼠标操控，因此近年来在个人电脑上使用模拟器运行移动游戏渐成潮流. 工业界传统的移动游戏模拟器，如“蓝叠”“雷电”和“逍遥”采用的都是常规 AOVB 软件加速虚拟化技术，导致重型三维游戏，如“和平精英”运行流畅度很差，对用户来说很难甚至无法忍受. 为突破模拟效率瓶颈，我们研发面向重型移动应用的“直接安卓模拟”(direct Android emulation)技术[49]，实现 x86 架构 Windows 操作系统上直接运行安卓应用的可执行文件，从而消除了常规虚拟化技术带来的额外计算开销，并且不依赖使用门槛较高的硬件虚拟化技术，如 Intel VT. 该技术被“腾讯手游助手”① 产品实际采用，活跃用户量超过 2 000 万，运行重型三维移动应用超过 8 000 个，模拟运行流畅度持平或接近高端手机直接运行. 对应系统还获得国际移动计算领域重要会议 International Conference on Mobile Computing and Networking (MobiCom) 2019 的最佳系统展示奖.

虽然直接安卓模拟技术效率很高，但其本质上依然属于库函数虚拟化，而且面向游戏市场，所以兼容性和安全性依然不足，不少安卓应用无法支持，“腾讯手游助手”每次运行前需要首先启动腾讯杀毒软件来“清理”环境. 那么，有没有可能实现效率、安全和兼容同时具备的理想化“三位一体”移动模拟器呢？如图 12 所示，我们提出“间接图形映射”[50] 虚拟化渲染技术，将 99.93% 以上的虚拟机-宿主机间图形交互操作，特别是上下文图形调用和资源图形调用消融或缓存于虚拟机内部，同时设计出辅助配合的弹性控制流调度机制和自适应数据流传送机制，从而大幅度降低虚拟化上下文切换开销、并且高效复用宿主机显卡的计算能力，在国际上实现 3 个关键维度同时兼顾的“三位一体”移动模拟器. 平均模拟

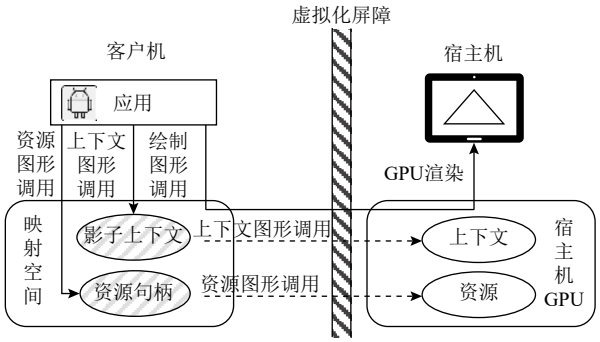


Fig. 12 Basic idea of indirect graphics projection is to construct a projection space in the virtual machine

图 12 间接图形映射的基本思想：在虚拟机中构造映射空间

效率 1.4 倍于腾讯手游助手、2.8 倍于谷歌安卓模拟器，而兼容性和安全性与谷歌安卓模拟器持平.

“三位一体”模拟器已被华为公司采用，替换原先的谷歌安卓模拟器，作为其 DevEco Studio 集成开发环境②的重要组成部分，解决我国基础软件领域的卡脖子问题. 相关代码和数据已开源③. 虽然已经突破关键技术瓶颈，但该模拟器目前还存在不少功能性瑕疵和稳定性漏洞，我们一直在和华为公司合作探索系统性的检测和修复方案.

5.2 基于云原生软件蜜罐的物联网终端设计

分布于人们日常生活方方面面的物联网终端设备数量庞大、功能繁杂，同时也面临严峻安全威胁，因为物联网设备通常廉价，异构与简化的设计导致适配服务器和个人电脑的传统安全措施收效甚微——物联网设备几乎都不能安装和运行常见的杀毒软件或防火墙. 蜜罐(honeypot)技术是互联网上发展成熟、广泛应用的研究已知或未知安全攻击的重要工具，但由于物联网设备架构多样、外设连接方式复杂，经典的互联网蜜罐难以适用. 国际上对物联网蜜罐的研发尚处于起步探索阶段，在保真性、完整性、封闭性等方面差强人意，规模化部署管理的开销也比较高昂，所以始终停留在小范围实验室阶段.

致力于以较低基础设施和人力成本开展全面系统的物联网安全研究，基于兼容性最强的动态二进制翻译技术(更具体地是 QEMU 模拟器)研发“云原生蜜罐”[51]，具备 3 个方面的特色：1) 硬件测量驱动软件设计，以小规模物联网硬件蜜罐实测结果指导

① https://syzs.qq.com

② https://DevEcoStudio.huawei.com

③ https://TrinityEmulator.github.io

模拟器软件蜜罐的科学设计，兼具保真性、完整性和封闭性；2）在跨越全球的多个公有云上自动化部署软件蜜罐，大幅度降低规模化管理维护的成本，同时将每个蜜罐收集到的攻击信息实时上传到后端云存储；3）关注强隐藏性的“无痕攻击”（也称“无文件攻击”），在文件系统无修改、Shell 命令未执行的不利条件下，依然能够实现较为准确的攻击画像与追根溯源.

如图 13 所示，我们研发的云原生软件蜜罐原型系统已在全球 8 个公有云（亚马逊 EC2、微软 Azure、谷歌云、DigitalOcean、Linode、Vultr、阿里云和腾讯云）上开展了长周期（12 个月）、中规模（108 节点）、低成本（6 美元/节点/月）的实战检验，收集到上亿次安全攻击的详细数据，发现近 600 种有痕攻击和 8 种无痕攻击，相关代码和收集到的各种数据已开源①. 同时我们也发现：8 个公有云对软件蜜罐的支持性和友好度差异很大，比如亚马逊云的 IP 地址范围全网公开、阿里云会拦截外部流量，需要灵活调整比例、弹性部署，并进一步降低维护成本，这些都是亟待解决的系统性问题.

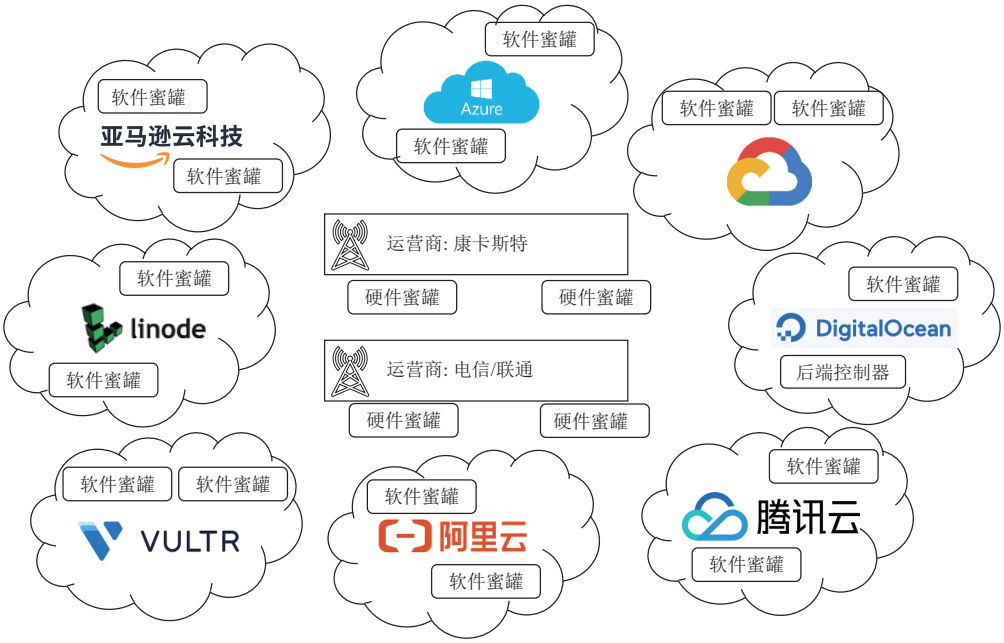


Fig. 13 Prototype system deployment for cloud native software honeypots

图 13 云原生软件蜜罐的原型系统部署

5.3 基于常规虚拟化执行的移动应用市场安全检测

移动应用市场是当前移动应用分发的主要渠道，同时也成为恶意应用快速散播的便捷途径. 过去十几年里，人工智能技术被广泛用于恶意应用检测，但是至今依然缺少面向移动应用市场的恶意应用智能检测系统. Google Play Store 据说采用了相似的技术方案，但从未对外公布任何技术细节和性能数据，研究者只能通过黑盒基准测试窥其一斑. 为了搞清楚构建此类系统的关键挑战，与腾讯应用市场②合作，研发基于行为特征（主要包括应用编程接口、申请权限和消息传递对象）和常规 AOVB 虚拟化技术的安卓应用智能安全检测系统[52-53]，其工作流程如图 14 所示.

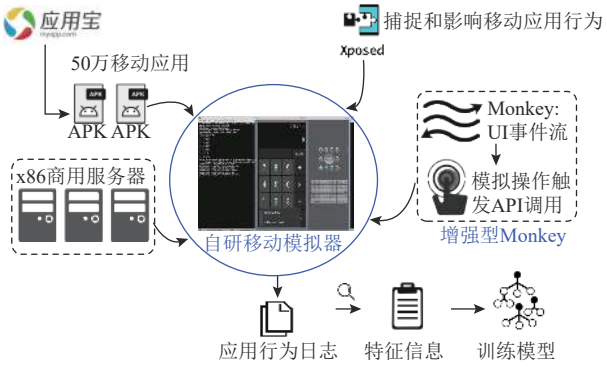


Fig. 14 General workflow of intelligent security detection system for Android Apps

图 14 安卓应用智能安全检测系统通用工作流程

① https://HoneyCloud.github.io

② https://sj.qq.com

研发经验揭示成功开发此类系统的5个关键点: 1)特征选择; 2)特征工程; 3)应用分析速度; 4)开发者参与; 5)模型进化. 其中任何一点失败将会导致整个系统因“木桶效应”而失败. 更重要地, 影响多数关键点的一项核心技术称为“智能事件生成”, 在经典的谷歌Monkey终端用户事件生成工具的基础上增加了针对执行死循环、事件冗余、频率单调、易被识别等原始设计缺陷的多项重要优化机制[54-55]. 最终实现的系统称为APIChecker, 因为安卓软件开发工具包(SDK)所提供的大量APIs是我们分析安卓应用行为的最基本特征, 实际部署并长期运行于腾讯应用市场, 仅使用单台商品化服务器(价格不超过10万元的普通配置服务器)就可以完成每天1万多个互联网新上传安卓应用的安全检测任务, 准确率和召回率都超过98%, 单个应用平均分析时间仅为1.3 min, 远低于Google Play Store所需的5 min. 我们的实践工作为人工智能和云计算技术系统性落地移动应用安全检测场景提供了第一手经验, 相关代码和数据已开源①.

基于上述经验, 进一步和字节跳动公司开展延伸科研合作, 解决通用模拟器和多版本安卓SDK、异构互联网流媒体编码协议以及图形渲染接口(比如OpenGL和OpenGL ES)之间的自适应兼容性问题, 帮助字节跳动公司多款移动应用(比如抖音、今日头条和西瓜视频)发现未知的设计缺陷和安全漏洞, 有效增强它们的运行时可靠性[56-57]. 在此过程中, 我们还发现谷歌安卓模拟器在解码特定格式视频时存在的重要缺陷, 分析缺陷原因并提出合理解法, 上报谷歌官方论坛后, 得到其研发团队的肯定与接受②.

未来我们计划将这一系列技术成果推广到更为普适以及前沿的场景, 特别是增强现实、虚拟现实、混合现实、元宇宙类资源消耗极高且其设计缺陷对用户体验影响极大的移动应用[58-59].

6 结论与展望

正如光是宇宙间最快速普适的信息载体, 互联网是人类在这个星球上最快速普适的信息传输工具; 光具有波粒二象性, 互联网则由差异极大的骨干网络和终端网络两部分构成. 相对于直观、简洁、稳定的光来说, 互联网深邃、复杂、多变; 对普通用户来说, 终端网络是互联网唯一直观可见的部分, 却又是最捉摸不定、瓶颈故障最多的部分. 本文以云原生强化设计的方法研究大规模复杂终端网络, 力图通过“生于云、长于云”(简洁)和“泛在场景、普适改进”(稳定)将其设计得更加可用、可靠、可信, 在多个工业系统中取得明显效果, 造福十亿网民.

然而, 我们的研究成果并不彻底、也不完整, 因为封闭神秘的骨干网对我们来说一直是个“黑洞”. 虽然它只导致1%的性能瓶颈, 却是解决互联网诸多问题的必然途径和最终归宿, 很多看似终端网络的问题只有同时诉诸骨干网络才能找到完备答案. 因此, 我们一直在寻求“双网合璧”的宝贵研究机会, 期望能从更为深广的层次洞悉互联网问题的本质.

此外, 第5节所介绍的基于移动模拟器的前瞻网络设计, 目前在严密性和可操作性2个方面还存在较大提升空间. 首先, 移动模拟器虽然在软件功能上能够仿真移动终端网络节点, 但在硬件(信号和功耗)和性能上却很难实现高保真性, 前瞻设计可能遗漏或者衍生的问题还需要从理论和实验上严密分析论证. 此外, 移动模拟器的配置和使用对普通网络研究者来说难度较大, 需要科学系统的教材、深入浅出的教程和清晰易懂的界面来帮助降低操作门槛.

作者贡献声明: 李振华总结科研成果、构思并撰写全文; 王泓懿负责部分绘图并完善参考文献; 李洋、林灏、杨昕磊负责部分技术内容补充和参考文献整理.

参 考 文 献

[1] Sundaresan S, De Donato W, Feamster N, et al. Broadband Internet performance: A view from the gateway[C]//Proc of 2011 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2011: 134−145

[2] Baranasuriya N, Navda V, Padmanabhan V N, et al. QProbe: Locating the bottleneck in cellular communication[C]//Proc of 2015 ACM Int Conf on Emerging Networking Experiments and Technologies (CoNEXT). New York: ACM, 2015: 1−7

[3] Tahir A, Mittal R. Enabling users to control their Internet[C]//Proc of 2023 USENIX Symp on Network System Design and Implementation (NSDI). Berkeley, CA: USENIX Association, 2023: 555−573

[4] Li Zhenhua, Dai Yafei, Chen Guihai, et al. Background and overview[M]// Content Distribution for Mobile Internet: A Cloud-

① https://APIChecker.github.io

② https://issuetracker.google.com/issues/262 255 458

based Approach, Second Edition. Berlin: Springer, 2023: 3−15

[5] Yang Xinlei, Lin Hao, Li Zhenhua, et al. Mobile access bandwidth in practice: Measurement, analysis, and implications[C]//Proc of 2022 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2022: 114−128

[6] Liu Yunhao, Li Zhenhua, Li Yang, et al. Recency effect and self-regulating design of mobile cellular systems in the context of interlaced generations: Network bandwidth, power efficiency, and connection reliability[J]. SCIENTIA SINICA Informationis, 2022, 52(12): 2290−2305 (in Chinese)
(刘云浩, 李振华, 李洋, 等. 代际交错背景下移动蜂窝系统的近因现象与自调控设计: 速度、能耗与可靠性 [J]. 中国科学: 信息科学, 2022, 52(12): 2290−2305)

[7] Liu Yunhao, Yang Qifan, Li Zhenhua. Cloud applications' development environment: From code logic to data flow diagram[J]. SCIENTIA SINICA Informationis, 2019, 49(9): 1119−1137 (in Chinese)
(刘云浩, 杨启凡, 李振华. 云计算应用服务开发环境: 从代码逻辑到数据流图 [J]. 中国科学: 信息科学, 2019, 49(9): 1119−1137)

[8] Yang Xinlei, Wang Xianlong, Li Zhenhua, et al. Fast and light bandwidth testing for Internet users[C]// Proc of 2021 USENIX Symp on Network System Design and Implementation (NSDI). Berkeley, CA: USENIX Association, 2021: 1011−1026

[9] Li Zhenhua, Li Xingyao, Yang Xinlei, et al. Fast uplink bandwidth testing for Internet users[J]. IEEE/ACM Transactions on Networking, 2023, 31(4): 1886−1901

[10] Liu Wei, Yang Xinlei, Lin Hao, et al. Fusing speed index during web page loading[C]//Proc of 2022 Int Conf on Measurement and Modeling of Computer Systems (SIGMETRICS). New York: ACM, 2022, 6(1): 1−23

[11] Li Yang, Lin Hao, Li Zhenhua, et al. A nationwide study on cellular reliability: Measurement, analysis, and enhancements[C]//Proc of 2021 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2021: 597−609

[12] Xiao Ao, Liu Yunhao, Li Yang, et al. An in-depth study of commercial MVNO: Measurement and optimization[C]//Proc of 2019 ACM Int Conf on Mobile Systems, Applications, and Services (MobiSys). New York: ACM, 2019: 457−468

[13] Li Yang, Zheng Jianwei, Li Zhenhua, et al. Understanding the ecosystem and addressing the fundamental concerns of commercial MVNO[J]. IEEE/ACM Transactions on Networking, 2020, 28(3): 1364−1377

[14] Li Yang, Li Zhenhua, Xin Xianlong. Attack economics based fraud detection for MVNO[J]. Computer Science, 2023, 50(8): 260−270 (in Chinese)
(李洋, 李振华, 辛显龙. 基于攻击经济学的移动虚拟运营商诈骗检测 [J]. 计算机科学, 2023, 50(8): 260−270)

[15] Li Zhenhua, Wang Weiwei, Wilson C, et al. FBS-Radar: Uncovering fake base stations at scale in the wild[C]//Proc of 2017 ISOC Network and Distributed System Security Symp (NDSS). Rosten, VA: ISOC, 2017

[16] Gao Di, Lin Hao, Li Zhenhua, et al. A nationwide census on WiFi security threats: Prevalence, riskiness, and the economics[C]//Proc of 2021 ACM Int Conf on Mobile Computing and Networking (MobiCom). New York: ACM, 2021: 242−255

[17] Udit P, Liu Jiamo, Gu Mengyang, et al. The importance of contextualization of crowdsourced active speed test measurements[C]//Proc of 2022 ACM Internet Measurement Conf (IMC). New York: ACM, 2022: 274−289

[18] Hu Ningning, Li Li, Mao Zhuoqing, et al. Locating Internet bottlenecks: Algorithms, measurements, and implications[C]//Proc of 2004 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2004: 41−54

[19] Feamster N, Livingood J. Measuring Internet speed: Current challenges and future recommendations[J]. Communications of the ACM, 2020, 63(12): 72−80

[20] Marcel D, Haeberlen A, Gummadi K P, et al. Characterizing residential broadband networks[C]//Proc of 2007 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2007: 43−56

[21] Deng Haotian, Peng Chunyi, Fida A, et al. Mobility support in cellular networks: A measurement study on its configurations and implications[C]//Proc of 2018 ACM Internet Measurement Conf (IMC). New York: ACM, 2018: 147−160

[22] Sommers J, Barford P. Cell vs. WiFi: On the performance of metro area mobile connections[C]//Proc of 2012 ACM Internet Measurement Conf (IMC). New York: ACM, 2012: 301−314

[23] Liu Jiajia, Kawamoto Y, Nishiyama H, et al. Device-to-device communications achieve efficient load balancing in LTE-advanced networks[J]. IEEE Wireless Communications, 2014, 21(2): 57−65

[24] Mardani S, Singh M, Netravali R. Fawkes: Faster mobile page loads via app-inspired static templating[C]//Proc of 2020 USENIX Symp on Network System Design and Implementation (NSDI). Berkeley, CA: USENIX Association, 2020: 879−894

[25] Netravali R, Mickens J. Prophecy: Accelerating mobile page loads using final-state write logs[C]//Proc of 2018 USENIX Symp on Network System Design and Implementation (NSDI). Berkeley, CA: USENIX Association, 2018: 249−266

[26] Ruamviboonsuk V, Netravali R, Uluyol M, et al. Vroom: Accelerating the mobile web with server-aided dependency resolution[C]//Proc of 2017 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2017: 390−403

[27] Mardani S, Goel A, Ko R, et al. Horcrux: Automatic JavaScript parallelism for resource-efficient web computation[C]//Proc of 2021 USENIX Symp on Operating Systems Design and Implementations (OSDI). Berkeley, CA: USENIX Association, 2021: 461−477

[28] Netravali R, Goyal A, Mickens J, et al. Polaris: Faster page loads using fine-grained dependency tracking[C]//Proc of 2016 USENIX Symp on Network System Design and Implementation (NSDI).

Berkeley, CA: USENIX Association, 2016: 123−136

[29] Sivakumar A, Jiang Chuan, Nam Y S, et al. NutShell: Scalable whittled proxy execution for low-latency web over cellular networks[C]//Proc of 2017 ACM Int Conf on Mobile Computing and Networking (MobiCom). New York: ACM, 2017: 448−461

[30] Meng Zili, Guo Yaning, Sun Chen, et al. Achieving consistent low latency for wireless real-time communications with the shortest control loop[C]//Proc 2022 of ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2022: 193−206

[31] Luo Zhihong, Fu S, Theis M, et al. Democratizing cellular access with CellBricks[C]//Proc of 2021 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2021: 626−640

[32] Xu Dongzhu, Zhou Anfu, Zhang Xinyu, et al. Understanding operational 5G: A first measurement study on its coverage, performance and energy consumption[C]//Proc of 2020 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2020: 479−494

[33] Narayanan A, Zhang Xumiao, Zhu Ruiyang, et al. A variegated look at 5G in the wild: Performance, power, and QoE implications[C]//Proc of 2021 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2021: 610−625

[34] Ahmad M, Jafri S U, Ikram A, et al. A low latency and consistent cellular control plane[C]//Proc of 2020 ACM Int Conf on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM). New York: ACM, 2020: 648−661

[35] Polese M, Giordani M, Mezzavilla M, et al. Improved handover through dual connectivity in 5G mm wave mobile networks[J]. IEEE Journal on Selected Areas in Communications, 2017, 35(9): 2069−2084

[36] Zarinni F, Chakraborty A, Sekar V, et al. A first look at performance in mobile virtual network operators[C]//Proc of 2014 ACM Internet Measurement Conf (IMC). New York: ACM, 2014: 165−172

[37] Peng Chunyi, Tu Guanhua, Li Chiyu, et al. Can we pay for what we get in 3G data access?[C]//Proc of 2012 ACM Int Conf on Mobile Computing and Networking (MobiCom). New York: ACM, 2012: 113−124

[38] Zhao Jinghao, Ding Boyan, Guo Yunqi, et al. SecureSIM: Rethinking authentication and access control for SIM/eSIM[C]//Proc of 2021 ACM Int Conf on Mobile Computing and Networking (MobiCom). New York: ACM, 2021: 451−464

[39] Kubat M, Matwin S. Addressing the curse of imbalanced training sets: One-sided selection[C]//Proc of 1997 Int Conf on Machine Learning (ICML). New York: ACM, 1997, 97(1): 179

[40] Zhuang Zhou, Ji Xiaoyu, Zhang Taimin, et al. FBSleuth: Fake base station forensics via radio frequency fingerprinting[C]//Proc of 2018 Asia Conf on Computer and Communications Security (AsiaCCS). New York: ACM, 2018: 261−272

[41] Bellard F. QEMU, a fast and portable dynamic translator[C]//Proc of 2005 USENIX Annul Technical Conf (ATC). Berkeley, CA: USENIX Association, 2005: 46−52

[42] Kedia P, Bansal S. Fast dynamic binary translation for the kernel[C]//Proc of 2013 ACM Symp on Operating Systems Principles (SOSP). New York: ACM, 2013: 101−115

[43] Barham P, Dragovic B, Fraser K, et al. Xen and the art of virtualization[C]//Proc of 2003 ACM Symp on Operating Systems Principles (SOSP). New York: ACM, 2003: 164−177

[44] Russell R. virtio: Towards a de-facto standard for virtual I/O devices[J]. ACM SIGOPS Operating Systems Review, 2008, 42(5): 95−103

[45] Kivity A, Kamay Y, Laor D, et al. KVM: The Linux virtual machine monitor[C/OL]//Proc of the 2007 Linux Symp, 2007: 225−230. [2023-12-13]. https://www.kernel.org/doc/ols/2007/ols2007v1-pages-225-230.pdf

[46] Christoffer D, Nieh J. KVM/ARM: The design and implementation of the Linux ARM hypervisor[C]//Proc of 2014 Int Conf on Architectural Support for Programming Languages and Operating Systems (ASPLOS). New York: ACM, 2014: 333−348

[47] Porter D E, Boyd-Wickizer S, Howell J, et al. Rethinking the library OS from the top down[C]//Proc of 2011 Int Conf on Architectural Support for Programming Languages and Operating Systems (ASPLOS). New York: ACM, 2011: 291−304

[48] Howell J, Parno B, Douceur J R. How to run POSIX apps in a minimal picoprocess[C]//Proc of 2013 USENIX Annul Technical Conf (ATC). Berkeley, CA: USENIX Association, 2013: 321−332

[49] Yang Qifan, Li Zhenhua, Liu Yunhao, et al. Mobile gaming on personal computers with direct Android emulation[C]//Proc of 2019 ACM Int Conf on Mobile Computing and Networking (MobiCom). New York: ACM, 2019: 1−15

[50] Gao Di, Lin Hao, Li Zhenhua, et al. Trinity: High-performance mobile emulation through graphics projection[C]//Proc of 2022 USENIX Symp on Operating Systems Design and Implementations (OSDI). Berkeley, CA: USENIX Association, 2022: 285−301

[51] Dang Fan, Li Zhenhua, Liu Yunhao, et al. Understanding fileless attacks on Linux-based IoT devices with HoneyCloud[C]//Proc of 2019 ACM Int Conf on Mobile Systems, Applications, and Services (MobiSys). New York: ACM, 2019: 482−493

[52] Yan Yuxuan, Li Zhenhua, Chen Qi, et al. Understanding and detecting overlay-based Android malware at market scales[C]//Proc of 2019 ACM Int Conf on Mobile Systems, Applications, and Services (MobiSys). New York: ACM, 2019: 168−179

[53] Gong Liangyi, Li Zhenhua, Qian Feng, et al. Experiences of landing machine learning onto market-scale mobile malware detection [C]//Proc of 2020 European Conf on Computer Systems (EuroSys). New York: ACM, 2020: 1−14

[54] Gong Liangyi, Lin Hao, Li Zhenhua, et al. Systematically landing machine learning onto market-scale mobile malware detection[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(7): 1615−1628

[55] Gong Liangyi, Li Zhenhua, Wang Hongyi, et al. Overlay-based Android malware detection at market scales: Systematically adapting to the new technological landscape[J]. IEEE Transactions on Mobile Computing, 2022, 21(12): 4488−4501

[56] Lin Hao, Qiu Jiaxing, Wang Hongyi, et al. Virtual device farms for mobile app testing at scale: A pursuit for fidelity, efficiency, and accessibility[C]//Proc of 2023 ACM Int Conf on Mobile Computing and Networking (MobiCom). New York: ACM, 2023

[57] Zhang Bing, Wen Zheng, Wei Xiaoyu, et al. InterDroid: An interpretable Android malware detection method for conceptual drift[J]. Journal of Computer Research and Development, 2021, 58(11): 2456−2474 (in Chinese)
(张炳, 文峥, 魏筱瑜, 等. InterDroid: 面向概念漂移的可解释性Android 恶意软件检测方法[J]. 计算机研究与发展, 2021, 58(11): 2456−2474)

[58] Zhou Chao, Li Zhenhua, Liu Yao, et al. A measurement study of Oculus 360 degree video streaming[C]//Proc of 2017 ACM Multimedia Systems Conf (MMSys). New York: ACM, 2017: 27−37

[59] Xiao Mengbai, Wang Shuoqian, Zhou Chao, et al. Miniview layout for bandwidth-efficient 360-degree video[C]//Proc of 2018 ACM Int Conf on Multimedia (MM). New York: ACM, 2018: 914−922



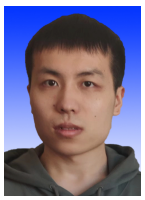
Li Zhenhua, born in 1983. PhD, tenured associate professor, PhD supervisor. Senior member of CCF. His main research interests include computer networking, operating systems, and cloud computing.

李振华, 1983 年生, 博士, 长聘副教授, 博士生导师. CCF 高级会员. 主要研究方向为计算机网络、操作系统、云计算.



Wang Hongyi, born in 2000. PhD candidate. Her main research interests include network measurement and machine learning.

王泓懿, 2000 年生. 博士研究生. 主要研究方向为网络测量、机器学习.



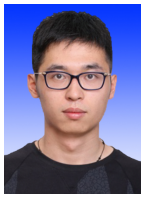
Li Yang, born in 1996. PhD candidate. His main research interests include network measurement and data mining.

李　洋, 1996 年生. 博士研究生. 主要研究方向为网络测量、数据挖掘.



Lin Hao, born in 1998. PhD candidate. His main research interests include operating systems and virtualization techniques.

林　灏, 1998 年生. 博士研究生. 主要研究方向为操作系统、虚拟化技术.



Yang Xinlei, born in 1997. PhD candidate. His main research interests include network measurement and Web techniques.

杨昕磊, 1997 年生. 博士研究生. 主要研究方向为网络测量、Web 技术.