曾祥哲

🌐 xzos.net | github.com/xz-dev | ✉ xz@xzos.net | ☎ +86 15530859511

专业技能

- 后端开发：Azure Serverless(Function/Container/Logic APP)，.NET，Azure VM，Redis，RPC，Protocol Buffers，Python WEB 服务器，区块链底层开发，IPFS 区块链存储，JAVA Spring，测试驱动开发，行为驱动开发，CI/CD 持续集成，嵌入式开发
- 运维技术：Docker，K8s，熟练使用 GNU/Linux 操作系统，熟悉 Linux 内核，理解 ArchLinux，Gentoo，openSUSE，Fedora 的内部原理
- 前端技术：React，Material Design，Android Jetpack Compose，QT
- 编程语言：Kotlin，Java，Python3，C#，Rust，C，PowerShell，Bash Script，JavaScript，SQL，LISPs
- 开发工具：Emacs，(Neo)Vim
- 证书：红帽认证工程师（285/300 分），红帽认证系统管理员（满分）

开源贡献和技能亮点

- 创建并维护包括 UpgradeAll (1k+★) 等 10 余个开源项目
- 积极参与开源社区：
  1. Armbian：PR#210 fix: config DNS for hotspot
  2. InfiniTime（嵌入式）：PR#2143 cmake: fix python path with use multi-version python，Chinese support branch
  3. Ansible-runner：PR#1306 Disable –tty for subprocess when parent process is non-tty
  4. DistroBox（容器化）：PR#1622 init: fix runtime sync fallback
  5. CachyOS-kernels（CachyOS 内核的 Gentoo 仓库）：维护 Szowisz/CachyOS-kernels 并添加了自动化工具简化维护
- 熟悉单元测试与持续集成，擅长对项目进行自动化转型
- 熟练使用版本控制系统 Git，领导过团队协作，熟悉且参与过敏捷开发
- 适应跨时区合作的工作
- 曾入选 Linux 基金会人才激励计划

工作经历

微软（外包） - Office365 互操作性和合规项目 - 全栈工程师 2023.10 - 至今

负责 Office 文档系统开发与自动化运维测试，为 GenDox 和 InteropTools 维护安全模型和更新，同时为 M365 活动创建演示项目。重写部分服务到 Azure Serverless，并通过 Azure 云服务自动化实现 CI 与自动运维工具。参与微软软件的中国 GB 字符认证，开发自动化测试平台和文档处理工具，并利用 AI 技术 (GPT 模型和视觉模型) 优化测试流程和错误检测。

- 设计并实现了安全合规的系统架构
- 构建完整的 CI/CD 流水线，提高部署效率
- 开发自动化测试平台，减少 80% 手动测试工作
- 训练定制 AI 模型用于 GB 字符渲染错误检测

拓维信息 - 联通 A 股门户网站 - JAVA 后端开发 2022.2 - 2022.7

联通为宣传 A 股的门户网站，因年久失修且技术落后，难以部署与维护网站内容。负责项目后端的微服务设计和代码重写，并实现了 k8s 的流水线部署与自动化扩/缩容。

- 使用微服务架构进行项目重构
- 采用敏捷开发方法开发
- 实现了可动态拓展的 k8s 部署

Power by LaTeX and Emacs. Written by Xiangzhe and AI

项目经历

微软 - Office Word 自动化测试平台 - Python 开发 2024.3 - 至今

项目背景：Office Word 需要大量字符兼容性和渲染测试来确保产品质量，传统人工测试方法效率低下且占用测试团队大量资源。为解决这一问题，我负责开发自动化测试平台，实现每周数百个测试用例的自动执行。

项目贡献：

- 设计并实现 Python 自动化测试框架，专用于 Office Word 软件的兼容性验证
- 创新性采用”代码即配置”的测试系统架构，使测试人员能通过 AI 辅助和 IDE 环境高效定义测试规则
- 集成 Windows API 接口模拟用户操作，构建端到端的自动化测试流程
- 开发基于 OCR 技术的渲染结果识别系统，提高字符兼容性测试的准确性
- 实现虚拟机环境中的批量测试调度功能，支持大规模并行测试执行

项目成果：

- 实现每周自动执行 800+ 测试用例，测试覆盖率显著提升。将测试团队投入的人工测试时间减少约 50%
- 通过系统化测试提高产品质量，降低字符渲染相关问题的用户反馈率

微软 - GB 文档校验与修正工具 - Python 开发 2024.6 - 至今

项目背景：微软 Office 套件需通过 GB 认证，要求测试团队提供标准化的测试结果文档。但合并文档时，常面临术语不统一和格式错乱等问题。为解决这些问题，我开发自动化工具对文档进行校验与修正，确保提交的文档符合标准要求。

项目贡献：

- 深入研究 Word 文档 Open XML 内部结构，掌握文档格式控制机制
- 实现 python-docx 未支持的剪切、复制和粘贴功能
- 利用 LLM 实现专业术语智能替换功能，确保文档术语表达一致性
- 利用 python-docx 和 oxml 库进行 OXML 底层操作，解决了多文档合并后出现排版错误的问题

项目成果：

- 将文档校验与修正时间减少 96.67%，从人均 2.5 小时缩短至 5 分钟
- 确保所有 GB 认证文档的术语一致性和格式规范性，提高认证通过率
- 减少测试团队在文档格式调整上的工作量，使其专注于测试内容本身
- 编写技术博客”Cut and move Runs via python-docx”，为开源社区贡献解决方案

微软 - Gendox 文档管理系统 - C# 开发 2023.10 - 至今

项目背景：Gendox 是微软内部的文档管理工具，以 Word 插件形式自动转化为 wiki。为产品经理提供结构化文档编写环境，采用”先建菜单再填内容”的方法，支持文档片段跨文档共享与同步修改。该项目涵盖从编辑到发布的全流程，集成了版本控制、自动化构建和安全保障等核心功能。

项目贡献：

- 调研 GenDox 插件加载与运行效率，开发基于 Python 和图像识别的自动化测试工具
- 基于 Azure Pipeline 构建自动发布系统，实现新版本的持续交付。并重构 Release 工具
- 设计 Azure Function 自动归档方案，集成 PowerBI 自动化遥测数据采集生成实时看板与邮件预警系统
- 升级安全模型，将基于密码的认证迁移至 Azure Managed Identity，并编写标准化迁移文档
- 开发基于 Azure Serverless 的自动化工具，实现 VM 的 Patch Tuesday 更新自动应用

项目成果：

- 高效处理每周高达 30G、40 万文件的 Release 文件
- 简化团队协作流程，减少新版本测试时间，消除跨团队人工交接，节省每次发布约 3 人/天的工作量
- 实现日志自动检查，消除人工审查可能造成的遗漏风险，减少潜在延误和损失
- 自动化月度维护工作，节省每月 1 人/天的系统检查与更新时间
- 完成微软 Q3 季度安全要求，提升系统整体安全性

微软 - Interop 部门数据同步与培训管理系统改进 - C# 开发 2023.10 - 至今

项目背景：该系统作为 Office Interop 部门的核心工具，承担跨项目人员和文档的数据仓库同步，和员工培训管理两大关键职能。旧系统存在运行缓慢，技术与安全架构落后等问题。

项目贡献：

- 原有基于 Task Scheduler 的固定时间执行模式每天运行超过 12 小时，用 Power Shell 实现服务依赖脚本和碎片化执行
- 升级项目安全架构以满足最新安全标准，将 CodeQL 集成至 Azure Pipeline，实现代码安全的自动化检测与持续集成
- 领导项目微服务化转型，将单体应用解耦为独立服务组件，通过 Azure Container 技术实现从.NET Framework 向.NET

项目成果：

- 将 Azure 资源费用降低 85%，调度系统将运行时间降低 50%，提升吞吐量等和稳定性
- 全面达成微软最新安全合规要求，微服务架构彻底消除 VM 维护相关的安全风险
- 微服务的灰度迁移方案实现系统零中断升级，保障用户体验持续平稳流畅

开源项目 - UpgradeAll 全栈应用更新器 - 项目发起人 2019.4 - 至今

项目链接: UpgradeAll (Kotlin/Rust 客户端)，Server (Python 服务端，2020.3-2022.6.5)

领导六人团队进行协作开发免费开源软件 UpgradeAll，解决传统更新存在的软件发布碎片化的问题。简化 Android 应用（包括未安装的应用）、Magisk 模块等的更新查找过程。项目致力于提供高速且易用的应用更新体验。客户端获得 1k+Star

客户端亮点（**Kotlin+Rust**）:

- 采用 Kotlin 开发前端，实现 Material Design 界面及相关组件
- 使用 Rust 开发高性能后端库，采用模块化代码设计，内核可独立使用
- 实现高度可自定义设置，支持通过 Json 配置更新来源。内嵌 JavaScript 引擎实现应用热更新能力

服务端亮点（**Python**）:

- 提供客户端 gRPC 和 REST 接口，支持从 GitHub、GitLab、F-Droid、Play Store 等多个源获取应用更新
- 使用 ZeroMQ 实现微服务架构与服务发现，设计可横向扩展的多层缓存服务架构
- 采用 Redis 实现分布式数据缓存，使用 Docker 容器化技术部署服务

项目成果：

- 将 30 分钟更新时间缩短至 2 分钟显著缩短用户应用更新时间，将原本需要手动查找的过程自动化
- 构建一站式应用更新平台，集成多个更新源

清华大学实验室项目 - 基于 IPFS 的文件分享应用 - Android 客户端开发 2021.4 - 2021.5

项目背景：传统的文件传输方式存在带宽限制、服务器依赖性高等问题。该项目旨在利用 IPFS (星际文件系统) 的分布式特性，构建一个同时支持面对面高速传输和远距离稳定共享的文件分享应用。

指导老师：赵黎

项目贡献：

- 设计并开发 Android 客户端原型，实现核心功能和用户界面
- 集成 IPFS 协议，构建高效的 P2P 文件传输网络
- 实现基于 Wi-Fi Direct 的面对面传输功能，大幅提高近距离传输速度
- 开发端到端加密系统，确保文件传输安全性
- 设计直观的文件预览界面，优化用户体验

项目成果：

- 近距离传输速度达到传统云存储解决方案的 3-5 倍，达到 1GB/S
- 成功实现不依赖中心服务器的 P2P 文件分享系统，提高性能和稳定性。
- 作为研究生课题的核心实现部分，获得指导老师高度评价

教育经历

华北理工大学 - 计算机科学与技术 - 本科 2023.6

课程：网络原理，计算机原理，软件工程，算法设计与分析，面向对象程序设计，数据库原理，操作系统（助教）

获奖经历

ASC18 世界大学生超级计算机竞赛 - 二等奖

Power by LaTeX and Emacs. Written by Xiangzhe and AI

Xiangzhe Zeng

xzos.net | github.com/xz-dev | ✉ xz@xzos.net | ☎ +86 15530859511

Professional Skills

- Backend Development: Azure Serverless(Function/Container/Logic APP), .NET, Azure VM, Redis, RPC, Protocol Buffers, Python Web Server, Blockchain infrastructure development, IPFS blockchain storage, JAVA Spring,
  Test-driven development, Behavior-driven development, CI/CD continuous integration, Embedded development
- DevOps: Docker, K8s, Proficient with GNU/Linux operating systems, Familiar with Linux kernel, Understanding of internal principles of ArchLinux, Gentoo, openSUSE, Fedora
- Frontend Technology: React, Material Design, Android Jetpack Compose, QT
- Programming Languages: Kotlin, Java, Python3, C#, Rust, C, PowerShell, Bash Script, JavaScript, SQL, LISPs
- Development Tools: Emacs, (Neo)Vim
- Certifications: Red Hat Certified Engineer (285/300 points), Red Hat Certified System Administrator (perfect score)

Open Source Contributions and Key Skills

- Created and maintained over 10 open source projects including UpgradeAll (1k+★)
- Active participation in open source communities:
  1. Armbian: PR#210 fix: config DNS for hotspot
  2. InfiniTime (Embedded): PR#2143 cmake: fix python path with use multi-version python, Chinese support branch
  3. Ansible-runner: PR#1306 Disable –tty for subprocess when parent process is non-tty
  4. DistroBox (Containerization): PR#1622 init: fix runtime sync fallback
  5. CachyOS-kernels (Gentoo repository for CachyOS kernel): Maintaining Szowisz/CachyOS-kernels and added automation tools to simplify maintenance
- Proficient in unit testing and continuous integration, skilled at automating project workflows
- Experienced with Git version control system, led team collaborations, familiar with and participated in agile development
- Adaptable to cross-timezone collaborations
- Selected for the Linux Foundation Talent Incentive Program

Work Experience

Microsoft (Vendor) - Office365 Interoperability and Compliance Project - Full Stack Engineer 2023.10 - Present

Responsible for Office document system development and automated operations testing, maintaining security models and updates for GenDox and InteropTools, while creating demonstration projects for M365 events. Rewrote services to Azure Serverless and implemented CI and automated operations tools through Azure cloud services. Participated in Microsoft software GB character certification for China, developed automated testing platforms and document processing tools, and optimized testing processes and error detection using AI technology (GPT models and vision models).

- Designed and implemented security-compliant system architecture
- Built complete CI/CD pipelines to improve deployment efficiency
- Developed automated testing platform, reducing manual testing work by 80%
- Trained custom AI models for GB character rendering error detection

Talkweb Information - China Unicom A-Share Portal Website - JAVA Backend Developer 2022.2 - 2022.7

The Unicom A-Share portal website, due to years of neglect and outdated technology, was difficult to deploy and maintain. Responsible for the microservice design and code rewrite of the project backend, implementing k8s pipeline deployment and automated scaling.

- Restructured the project using microservice architecture
- Adopted agile development methods
- Implemented dynamically scalable k8s deployment

Project Experience

Microsoft - Office Word Automated Testing Platform - Python Development 2024.3 - Present

Project Background: Office Word requires extensive character compatibility and rendering tests to ensure product quality. Traditional manual testing methods are inefficient and consume significant resources from the testing team. To address this issue, I was responsible for developing an automated testing platform to execute hundreds of test cases weekly.

Project Contributions:

- Designed and implemented a Python automation testing framework specifically for Office Word software compatibility verification
- Innovatively adopted a ”code as configuration” test system architecture, allowing testers to efficiently define test rules through AI assistance and IDE environment
- Integrated Windows API interfaces to simulate user operations, building end-to-end automated testing processes
- Developed an OCR-based rendering result recognition system to improve the accuracy of character compatibility testing
- Implemented batch test scheduling in virtual machine environments to support large-scale parallel test execution

Project Outcomes:

- Achieved automated execution of 800+ test cases weekly, significantly improving test coverage. Reduced manual testing time by approximately 50%
- Improved product quality through systematic testing, reducing user feedback rates for character rendering issues

Microsoft - GB Document Verification and Correction Tool - Python Development 2024.6 - Present

Project Background: Microsoft Office suite needs to pass GB certification, requiring standardized test result documents from the testing team. When merging documents, issues such as inconsistent terminology and format disruption often occur. To solve these problems, I developed an automated tool to verify and correct documents, ensuring they meet standard requirements.

Project Contributions:

- Thoroughly researched the internal structure of Word document Open XML, mastering document format control mechanisms
- Implemented cut, copy, and paste functionalities not supported by python-docx
- Utilized LLM for intelligent replacement of professional terminology, ensuring document terminology consistency
- Used python-docx and oxml libraries for OXML low-level operations, solving formatting errors that occurred after merging multiple documents

Project Outcomes:

- Reduced document verification and correction time by 96.67%, from an average of 2.5 hours to 5 minutes per person
- Ensured terminology consistency and format standardization for all GB certification documents, improving certification pass rates
- Reduced the testing team’s workload on document formatting adjustments, allowing them to focus on the test content itself

- Wrote a technical blog ”Cut and move Runs via python-docx”, contributing solutions to the open source community

Microsoft - Gendox Document Management System - C# Development 2023.10 - Present

Project Background: Gendox is Microsoft’s internal document management tool, which automatically converts to wiki in the form of a Word plugin. It provides product managers with a structured document writing environment, adopting the ”build menu first, then fill in content” method, supporting cross-document sharing and synchronous modification of document fragments. The project covers the entire process from editing to publishing, integrating core functions such as version control, automated building, and security assurance.

Project Contributions:

- Researched GenDox plugin loading and running efficiency, developed Python-based and image recognition-based automated testing tools
- Built an automatic publishing system based on Azure Pipeline, achieving continuous delivery of new versions. Refactored the Release tool
- Designed an Azure Function automatic archiving solution, integrated PowerBI automated telemetry data collection to generate real-time dashboards and email alert systems
- Upgraded the security model, migrating from password-based authentication to Azure Managed Identity, and wrote standardized migration documentation
- Developed Azure Serverless-based automation tools to automatically apply Patch Tuesday updates for VMs

Project Outcomes:

- Efficiently processed up to 30GB and 400,000 files weekly for releases
- Simplified team collaboration processes, reduced new version testing time, eliminated cross-team manual handovers, saving approximately 3 person-days of work per release
- Implemented automatic log checking, eliminating oversight risks from manual reviews, reducing potential delays and losses
- Automated monthly maintenance work, saving 1 person-day per month for system inspection and updates
- Completed Microsoft Q3 quarter security requirements, enhancing overall system security

Microsoft - Interop Department Data Synchronization and Training Management System Improvement - C# Development 2023.10 - Present

Project Background: This system serves as a core tool for the Office Interop department, undertaking two key functions: cross-project personnel and document data warehouse synchronization, and employee training management. The old system had issues such as slow operation, outdated technology, and security architecture.

Project Contributions:

- The original Task Scheduler-based fixed-time execution mode ran for over 12 hours daily; implemented service dependency scripts and fragmented execution with PowerShell
- Upgraded the project security architecture to meet the latest security standards, integrated CodeQL into Azure Pipeline to achieve automated detection and continuous integration of code security
- Led the project’s microservice transformation, decoupling the monolithic application into independent service components, implementing migration from .NET Framework to .NET through Azure Container technology

Project Outcomes:

- Reduced Azure resource costs by 85%, decreased scheduling system runtime by 50%, improved throughput and stability
- Fully met Microsoft’s latest security compliance requirements; microservice architecture completely eliminated VM maintenance-related security risks
- The microservice gray migration solution achieved zero-downtime system upgrades, ensuring a continuously smooth user experience

Open Source Project - UpgradeAll Full Stack Application Updater - Project Initiator 2019.4 - Present

Project Links: UpgradeAll (Kotlin/Rust client), Server (Python server, 2020.3-2022.6.5)

Led a six-person team in collaborative development of the free open-source software UpgradeAll, solving the fragmentation problem in traditional software updates. Simplified the update search process for Android applications (including uninstalled apps), Magisk modules, etc. The project aims to provide a fast and user-friendly application update experience. The client has received 1k+ Stars.

Client Highlights (Kotlin+Rust):

- Developed frontend with Kotlin, implementing Material Design interface and related components
- Developed high-performance backend library using Rust, with modular code design; the kernel can be used independently
- Implemented highly customizable settings, supporting update sources configuration via Json. Embedded JavaScript engine enabling application hot update capability

Server Highlights (Python):

- Provided client gRPC and REST interfaces, supporting application updates from multiple sources including GitHub, GitLab, F-Droid, Play Store, etc.
- Used ZeroMQ to implement microservice architecture and service discovery, designed a horizontally scalable multilayer cache service architecture
- Used Redis to implement distributed data caching, deployed services using Docker containerization technology

Project Outcomes:

- Significantly reduced user application update time from 30 minutes to 2 minutes, automating what was previously a manual search process
- Built a one-stop application update platform integrating multiple update sources

Tsinghua University Laboratory Project - IPFS-based File Sharing Application - Android Client Development 2021.4 - 2021.5

Project Background: Traditional file transfer methods face bandwidth limitations and high server dependency. This project aimed to leverage the distributed nature of IPFS (InterPlanetary File System) to build a file sharing application that supports both face-to-face high-speed transmission and stable long-distance sharing.

Supervisor: Li Zhao

Project Contributions:

- Designed and developed Android client prototype, implementing core functionalities and user interface
- Integrated IPFS protocol, building an efficient P2P file transfer network
- Implemented Wi-Fi Direct-based face-to-face transfer functionality, significantly improving short-distance transfer speed
- Developed end-to-end encryption system, ensuring file transfer security
- Designed intuitive file preview interface, optimizing user experience

Project Outcomes:

- Short-distance transfer speed reached 3-5 times that of traditional cloud storage solutions, achieving 1GB/S
- Successfully implemented a P2P file sharing system without relying on central servers, improving performance and stability
- Served as the core implementation part of a graduate thesis, highly praised by the supervisor

Education

North China University of Science and Technology - Computer Science and Technology - Bachelor's Degree 2023.6

Courses: Network Principles, Computer Principles, Software Engineering, Algorithm Design and Analysis, Object-oriented Programming, Database Principles, Operating Systems (Teaching Assistant)

Awards & Honors

ASC18 World University Supercomputer Competition - Second Prize

Power by LaTeX and Emacs. Written by Xiangzhe and AI