面向云桌面环境的安全运维管理平台架构设计

王中华[1,2]，张　鹏[1,2]，刘中一[1,2]，黄向平[1,2]

（1. 中国民航信息网络股份有限公司，北京　101318；2. 北京市民航大数据工程技术研究中心，北京　101318）

摘　要：运维管理平台是业务系统在开发、测试、运维过程中的重要工具。近年来，随着信息安全形势逐步严峻，越来越多的企业选用云桌面这一安全加固方式，实现办公、开发测试、生产三种网络环境的隔离。为了在网络隔离环境下安全、便捷地访问各个环境的业务系统，结合客户端/服务器和浏览器/服务器架构的优点，设计了一种面向云桌面环境的安全运维管理平台架构方案。该架构不占用服务器资源，可以降低系统信息安全隐患，并可在各个环境之间迁移复用，极大地提升了运维管理平台的开发效率及升级灵活性。

关键词：网络隔离；云桌面；信息安全；运维管理平台；架构设计

中图分类号：TP311；TP309　　**文献标识码：**A　　**DOI：**10. 19358/j. issn. 2097 - 1788. 2024. 12. 002

引用格式：王中华，张鹏，刘中一，等. 面向云桌面环境的安全运维管理平台架构设计［J］. 网络安全与数据治理，2024，43(12)：10 - 18.

Architecture design of secure operation and maintenance management platform for cloud desktop environment

Wang Zhonghua[1,2], Zhang Peng[1,2], Liu Zhongyi[1,2], Huang Xiangping[1,2]

(1. TravelSky Technology Limited, Beijing 101318, China; 2. Beijing Engineering Research Center of Civil Aviation Big Data, Beijing 101318, China)

Abstract: The operation and maintenance management platform is a crucial tool in the development, testing, and operation processes of business systems. In recent years, as the information security situation has become severe, more and more enterprises adopt cloud desktops as a means of security reinforcement, enabling the isolation of three distinct network environments for office work, development and testing, and production. To enable secure and convenient access to business systems in various environments under a network isolation setting, an architecture solution of a secure operation and maintenance management platform for cloud desktop environment has been designed, combining the advantages of both Client/Server (C/S) and Browser/Server (B/S) architectures. This solution does not occupy server resources, minimizes potential information security risks within the system, facilitates migration and reuse across environments, and significantly enhances the development efficiency and upgrade flexibility of the operation and maintenance management platform.

Key words: network isolation; cloud desktop; information security; operation and maintenance management platform; architecture design

0　引言

运维管理平台可以对业务系统中的请求、结果、日志、数据进行分析，为产品、开发、测试、运维人员提供诸如场景追溯、日志抓取、用户模拟、结果比对、数据查询、状态监控等功能，它包含多种工具和资源，以支持问题的高效处理和解决。通常，运维管理平台只为企业内部用户提供服务，且对业务系统是只读的，不会对业务系统数据进行任何写入或修改。专业的运维管理平台将极大地提升各类人员的工作效率，最终达到增强业务系统稳定性、加固生产安全的目的，其重要性不言而喻[1]。

近年来，随着信息安全形势逐步严峻，众多企业选用云桌面技术实现办公、开发测试、生产三种网络环境的隔离。云桌面在实现网络隔离、提升安全性的同时，也带来了一些问题，例如，在办公网的个人计算机无法直接访问部署在开发测试网、生产网的运维管理平台，

且运维管理平台无论是以客户端/服务器（Client/Server，C/S）架构的桌面应用部署在云桌面里，还是以浏览器/服务器（Browser/Server，B/S）架构的 Web 应用部署在开发测试网或生产网，都不能完全满足运维管理平台在开发效率、部署复杂度、升级灵活性、跨平台性方面的需求。

本文结合 C/S 架构和 B/S 架构的优点，设计了一种面向云桌面环境的安全运维管理平台的架构方案。该设计不占用服务器资源，可以降低系统信息安全隐患，并可在各个环境之间迁移复用，极大地提升了运维管理平台的开发效率及升级灵活性。

1 应用场景分析

1.1 网络环境限制

传统的运维管理平台部署在公司内网，产品、开发、测试、运维人员可以通过个人计算机访问运维管理平台，运维管理平台可以直接访问开发测试系统以及生产系统，如图 1 所示。

近年来，随着信息安全形势逐步严峻，国家及企业对信息安全的意识和要求不断增强，越来越多的企业选用了云桌面[2-3]这一信息安全加固方式，实现办公、开发测试、生产三种网络环境的隔离。云桌面技术是一种基于虚拟化技术的解决方案，该技术通过深度融合前后端软硬件，整合服务器虚拟化、桌面虚拟化和存储虚拟化，实现桌面应用环境与终端设备的有效分离。这种架构使得用户可以通过终端设备不受物理位置限制远程访问其个性化桌面。同时，云桌面技术提供了安全的数据保护机制，有效保障用户数据在传输和存储过程中的安全性和保密性[4]。在网络隔离的环境中，部署在办公网的个人计算机无法直接访问开发测试网、生产网，只有通过专用的云桌面，才可以访问相应的开发测试网、生产网，如图 2 所示。

1.2 架构方案对比

为了适应网络环境的变化，在上述网络隔离的环境中，运维管理平台有如下几种架构方案。

1.2.1 C/S 架构下的桌面应用

C/S 架构[5]是指将客户端与服务端分开分发、部署、提供计算机服务的系统架构。如图 3 所示，运维管理平台为 C/S 架构的桌面应用，直接运行在云桌面中。这种架构的优点是能够充分利用客户端计算资源，不占用服务器资源；客户端与服务器之间的直接通信减少了数据泄露的风险，安全性较高。这种架构也存在着一些缺点。首先，通信协议、编程框架多为自定义，使得系统间的兼容性和复用性大打折扣，当需要集成新系统或功能时，往往需要重新编写接口代码，大大增加了开发成本。其次，客户端的升级过程繁琐，每当有新版本发布，所有用户均需手动更新客户端软件，这不仅增加了运维难度，

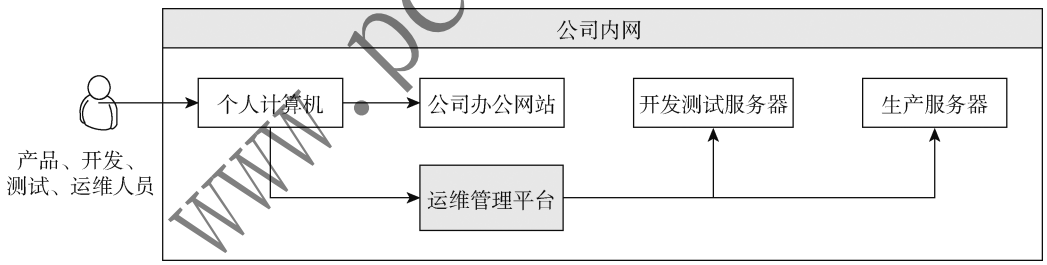


图 1 网络未隔离时的运维管理平台

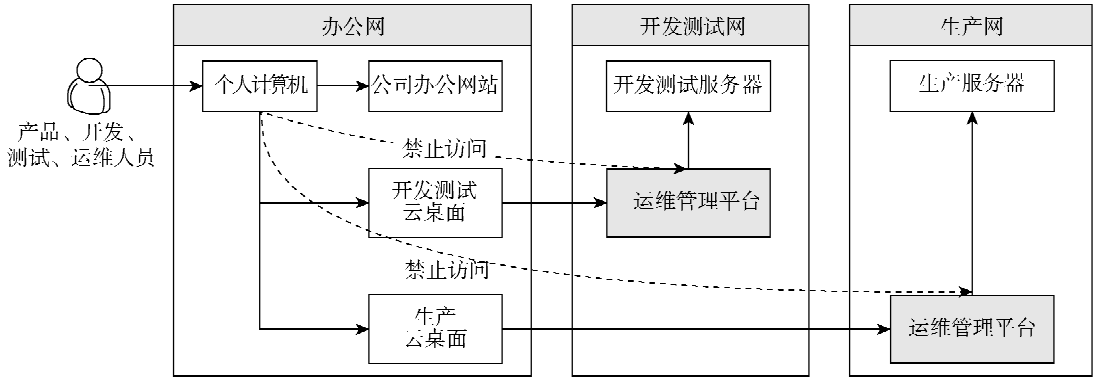


图 2 网络隔离环境下的运维管理平台

也可能因用户未及时更新而引入安全漏洞。再者，客户端软件的高度定制化意味着它往往受限于特定的操作系统平台，限制了运维管理平台在不同环境下的灵活部署和使用，难以满足日益增长的跨平台操作需求。

1.2.2 B/S 架构下开启外部端口的 Web 应用

B/S 架构[6-7]是构建网络应用程序的重要模式，它由客户端和服务器端组成，客户端只需要安装浏览器，与具备丰富功能的服务器进行交互，服务器端承担主要的数据处理与计算任务。如图 4 所示，运维管理平台为 B/S 架构的 Web 应用，在云桌面中通过浏览器进行访问。这种架构的优点是通信协议、编程框架的复用性高，开发成本低、效率高；维护和升级方式简单，对客户透明；跨平台性强，支持多种操作系统，客户端只需安装浏览器，无需安装额外软件。缺点是部署在生产网络中的运维管理平台需要开启并保持必要的外部网络服务端口，这些网络端口增大了被网络嗅探和恶意攻击的风险，给生产环境带来不必要的安全隐患，在某些特殊的保障时期，需要关停系统以提升安全级别，在此期间无法使用运维管理平台进行生产问题排查与分析，严重影响工作效率；运维管理平台的需求来自于产品、开发、测试、运维人员，需求的更新频率高、灵活性大，运维管理平台部署在生产网络中，任何的修改、升级都必须遵循严格的生产系统上线流程，需要进行长时间的功能、性能测试，这与运维管理平台灵活调整的初衷相背离，严重影响其更新迭代的速度；此外，部署在生产网络中的运维管理平台需要符合生产系统安全的各项规范，如必须具备严格的身份鉴别、数据加密、防篡改、防抵赖等特性，这些都给系统开发带来相当多的额外工作量，且运维管理平台不直接参与生产服务，不服务外部客户，不符合生产系统的定位，却占用着生产资源与相应级别的保障服务，是一种软硬件及人力资源的浪费。

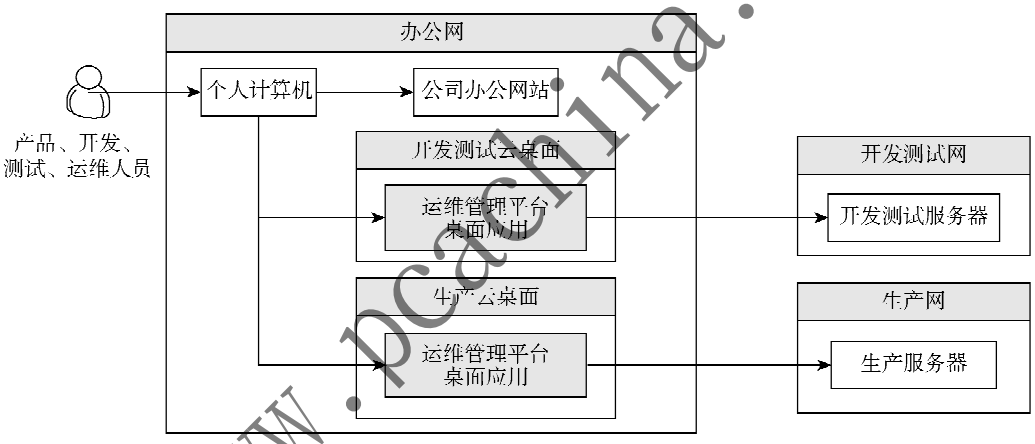


图 3 运维管理平台桌面应用

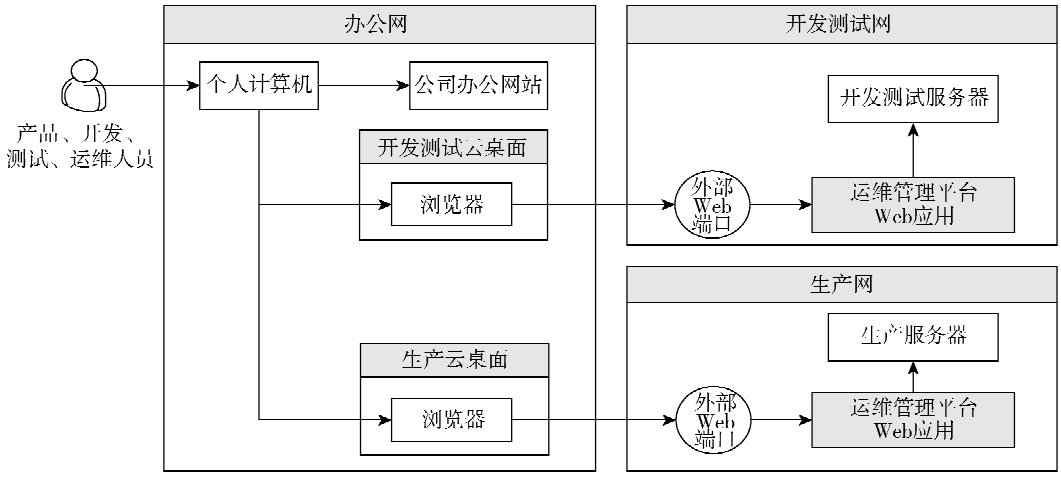


图 4 运维管理平台 Web 应用

1.2.3 B/S 架构下开启内部端口的 Web 应用

为了解决上述开启外部网络服务端口带来的安全性问题，一种替代方式是运维管理平台不再暴露外部 Web 端口，而是开启内部 Web 端口，经过安全的 SSH 服务端口转发[8]，如图 5 所示。这种架构虽然可以有效提升 Web 应用的安全性，但其在部署复杂性、访问便捷性和运维管理难度等方面均存在不容忽视的缺点。首先，从部署角度来看，内部 Web 端口的设置要求开发者或运维人员具备较高的网络配置能力，需要确保内网环境的安全与稳定，同时配置 SSH 端口转发规则，这增加了部署的复杂度和时间成本。对于多环境（如开发、测试、生产）的部署，每个环境都需单独配置，进一步增大了工作量。其次，在访问体验上，用户或开发者需要通过 SSH 跳转指令来访问 Web 应用，这不仅要求用户熟悉 SSH 的使用，还可能导致访问速度受限于 SSH 连接的稳定性和带宽。对于需要频繁访问或高效协作的场景，这种访问方式不够便捷和高效。再者，从运维管理的角度来看，内部 Web 端口的部署增加了监控和故障排查的难度。由于服务被隐藏在内网，外部安全扫描和监控工具无法直接应用，需要依赖内部监控机制，这要求运维团队具备更强的内部网络管理能力和应急响应能力。

1.2.4 B/S 架构下通过容器部署的 Web 应用

为了解决部署在生产网络中占用生产资源，修改升级必须遵循严格的生产系统上线流程，更新迭代速度慢等问题，可以将运维管理平台 Web 应用生成镜像，运行于云桌面里的 Docker[9-10] 容器，通过云桌面里的浏览器访问，如图 6 所示。容器技术基于虚拟化技术，可以实现程序从一个计算环境快速可靠地转移到另一个计算环境运行。Docker 是一种开源的应用容器引擎，让开发者可以将他们的应用及依赖包打包到一个可移植的镜像中，然后发布到任何版本的 Linux 或 Windows 操作系统的机器上[11]。这种架构提高了应用的可移植性，加快了环境的一致性配置，但也伴随着一些不可忽视的缺点。首先是安全性和隔离性挑战。尽管 Docker 提供了相对隔离的环境，但容器间的隔离程度相较于虚拟机仍有所不足，存在潜在的安全风险，如容器逃逸等问题。此外，若云桌面本身存在安全漏洞，也可能间接影响容器内应用的安全性。其次，管理复杂度增加。随着容器数量的增加，如何有效管理这些容器的生命周期、资源分配、监控及日志收集等变得更为复杂，特别是在多用户、多项目的云桌面环境中，如何确保资源的合理分配与隔离，避免资源竞争和冲突，成为运维团队面临的重要挑战。再者，依赖性和兼容性风险。Docker 镜像的构建和运行高度依赖于 Docker 引擎及其配置，以及底层操作系统的兼容性，任何一环的更新或变化都可能影响应用的稳定性和性能，需要持续跟踪和测试以确保兼容性。

表 1 从多个维度对上述各种架构进行了对比，可以看出各种架构互有优劣。

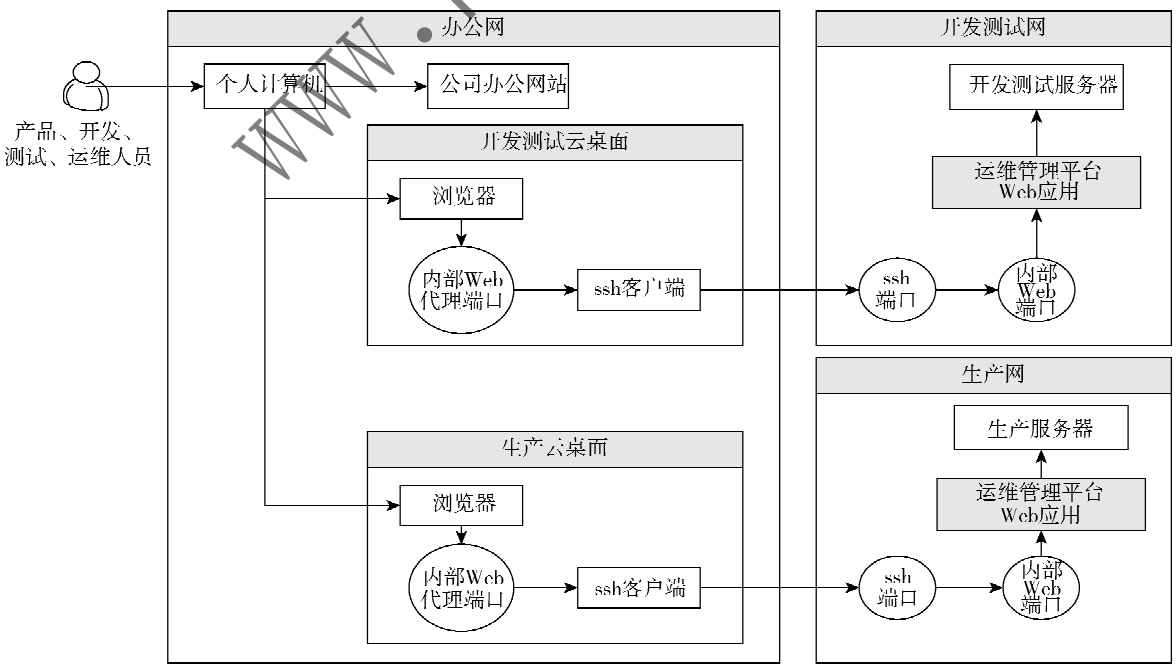


图 5 Web 应用开启内部端口

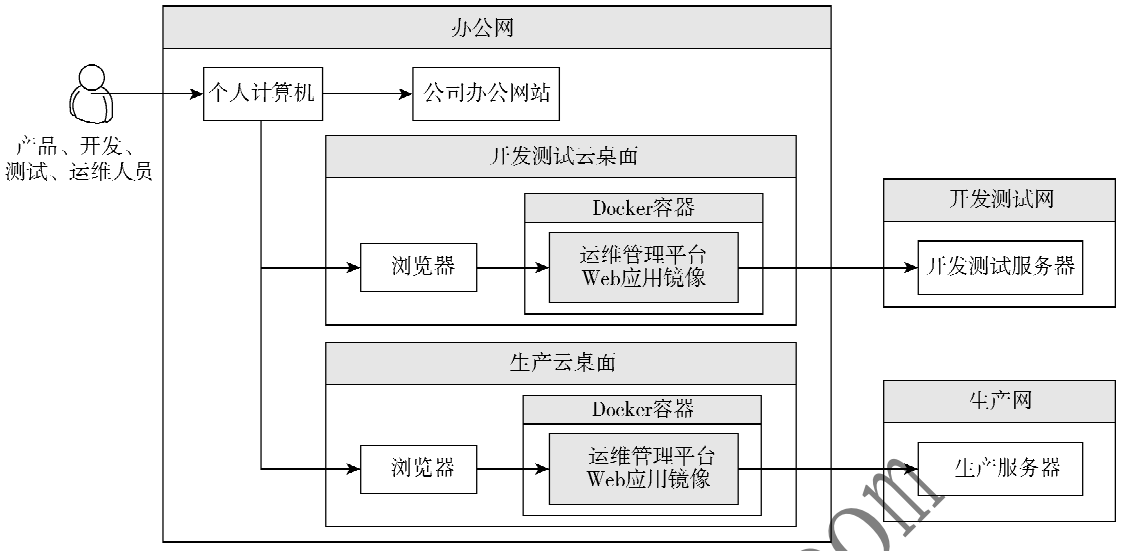


图 6　Web 应用生成镜像运行于云桌面里的 Docker 容器

表 1　各种架构对比

对比维度	C/S 架构	B/S 架构		
		开启外部端口	开启内部端口	生成镜像
部署位置	云桌面	开发测试网/生产网	开发测试网/生产网	云桌面
安全性	高	低	高	高
开发效率	低	高	高	高
部署复杂度	低	低	高	高
升级灵活性	高	低	低	高
跨平台性	弱	强	强	强

2　系统架构设计

2.1　设计思路

针对运维管理平台的应用场景，企业需要一种部署在云桌面里，有较高的安全性、开发成本低、效率高、部署简单、迭代升级灵活且跨平台性强的架构。在网络隔离环境下，C/S 架构有着较高的安全性、部署简单、升级灵活性较高，但是开发成本高、效率低、跨平台性较弱。B/S 架构开发成本低、效率高、跨平台性强，在开启外部端口时部署简单但是安全性较低，在开启内部端口或者生成镜像时部署复杂但是安全性高，升级灵活性一般较低，只有在生成镜像时较高。可见 C/S 架构或者 B/S 架构本身都无法满足上述需求，因此，本文考虑将两者有机结合，扬长避短，充分发挥各自的优势。

2.2　整体架构

如图 7 所示，FastAPI[12-13] 是一个高性能的 Web 框架，专为快速构建 Web 应用而设计。Pywebview 是一个轻量级跨平台的浏览器控件，它允许在图形化桌面应用中显示 HTML、CSS 和 JavaScript 内容。PyInstaller 是一个功能强大且易于使用的 Python 程序打包工具，它能够将 Python 程序转换为可以在目标机器上独立运行的可执行文件，无需事先安装 Python 解释器或其他依赖项。

首先基于 FastAPI 框架进行 B/S 架构的 Web 应用开发，然后由 PyInstaller 打包器将运维管理平台网站及 Pywebview 浏览器控件打包为 C/S 架构的运维管理平台桌面应用。其中浏览器控件及打包器均为业界流行的可用组件。整个运维管理平台利用了浏览器的特性，整体展现为一个“本地应用程序”，部署由开发测试网或生产网转移至云桌面中。

这种架构兼具了 B/S 架构和 C/S 架构的优点。一方面，运维管理平台采用 B/S 架构，基于 FastAPI Web 框架开发，开发成本低、效率高。另一方面，将开发好的 B/S 架构的运维管理平台封装成 C/S 架构的桌面应用，不部署到生产网络，而是部署在云桌面中，无需开放外部端口，降低了生产安全隐患，在某些特殊的保障时期也无需关停；而且，部署简单，在各个环境的云桌面中安装方式统一；此外，由于部署在云桌面中，不在生产网络中部署，无需遵循生产系统严苛的上线流程，迭代升级灵活性高、速度快；最后，浏览器控件 Pywebview 和打包器 PyInstaller 都具备很强的跨平台兼容性，可以根据需要打包在不同操作系统上运行的桌面应用。

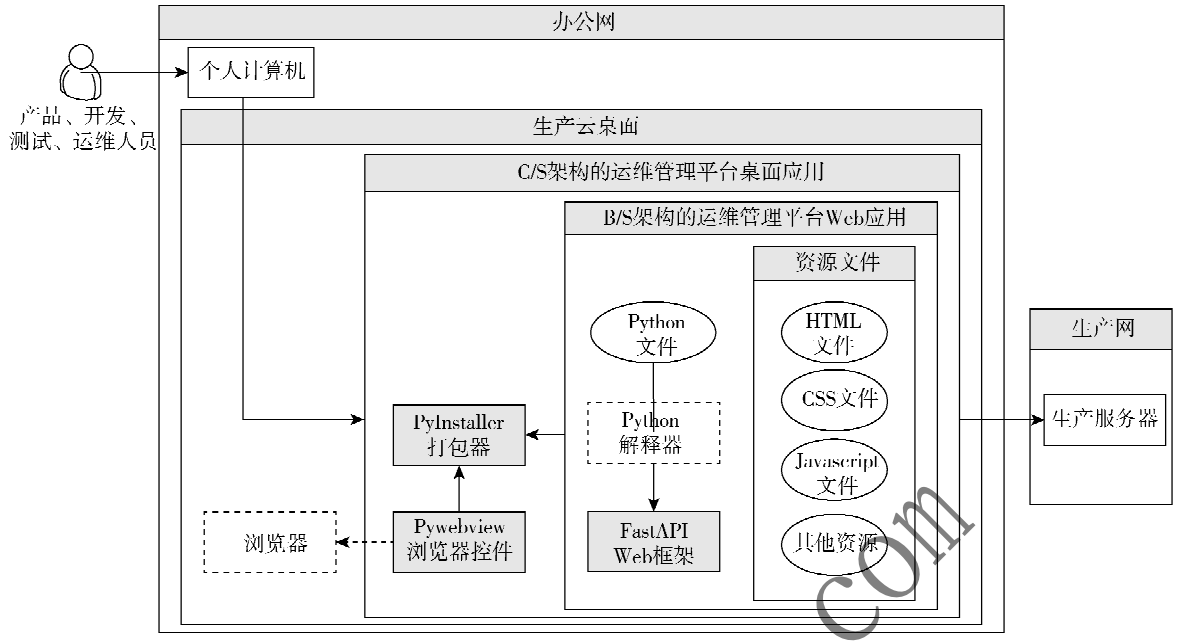


图 7　运维管理平台整体架构

2.3　开发工具链

2.3.1　FastAPI

FastAPI 是一个现代、快速且高效的 Web 框架，用于构建应用程序接口（Application Program Interface，API），使用 Python 编写。它基于标准 Python 类型提示，通过自动生成交互式 API 文档、数据验证和序列化/反序列化，极大地简化了 API 开发过程。FastAPI 利用 Pydantic 进行数据验证和设置管理，以 Starlette 作为底层 ASGI 框架，确保了应用的性能和可扩展性。其设计哲学是快速编码、低学习曲线和高生产效率，让开发者能够迅速构建出既快速又安全的 RESTful API。FastAPI 还提供了对 WebSockets 的原生支持，以及易于集成的认证和依赖注入系统，是构建现代 Web 应用的理想选择。

2.3.2　Pywebview

Pywebview 是一个轻量级跨平台的浏览器控件。它让桌面程序拥有了 Web 技术的强大功能，而隐藏了其是基于浏览器的事实。Pywebview 的核心功能基于各种操作系统的 WebView 组件，如 Windows 的 MSHTML（IE）或 Edge HTML（Edge），Linux 的 WebKitGTK，以及 macOS 的 WebKit（Safari）等。Pywebview 的核心优势在于它弥合了桌面应用和 Web 技术之间的鸿沟，可以将 Pywebview 与 FastAPI、Django 等 Python Web 框架结合使用，从而轻松地构建具有复杂后端逻辑的桌面应用，这种结合使得开发者能够充分利用 Web 技术的优势，同时保持桌面应用的灵活性和可扩展性。

2.3.3　PyInstaller

PyInstaller 是一个功能强大且易于使用的 Python 程序打包工具，它能够将 Python 程序及其所有依赖项打包成独立的可执行文件，使得这些程序可以在没有安装 Python 解释器的环境中直接运行，用户无需关心 Python 环境的配置问题，极大地简化了 Python 应用程序的分发和部署过程[14-15]。PyInstaller 支持在多个操作系统（如 Windows、Linux 和 macOS）上打包 Python 程序，以满足不同用户群体的需求。此外，它还提供了许多选项和参数，用户可以根据需要对打包过程进行定制，如指定输出目录和图标文件、添加资源文件、排除特定模块等。

3　工程实例验证

上述架构设计方案已经在某业务系统的运维管理平台上完成了实践验证，本文以此为例进行验证说明。

3.1　开发过程

首先，基于 FastAPI 框架进行 B/S 架构的 Web 应用开发、测试，得到运维管理平台的网站。

然后，使用 PyInstaller 把基于 FastAPI 框架的 Web 网站与 Pywebview 进行打包，生成可执行的桌面程序，如图 8、9 所示。执行完毕后，源文件所在目录将生成 dist 和 build 两个文件夹。其中 build 是 PyInstaller 存储临时文件的目录，可以安全删除。最终的打包程序在 dist 文件夹中，目录中其他文件是可执行文件的动态链接库[16]。

最后，将生成的运维管理平台桌面程序发布至云桌面投入使用，如图 10 所示。运维管理平台桌面程序的界面与之前运维管理平台网站的界面完全一致。

```
D:\ipric_v1_fe_guitools>pyinstaller -i app.ico ^
--hiddenimport requests --hiddenimport json --hiddenimport json_tools --hiddenimport xmltodict ^
--hiddenimport xml.dom.minidom --hiddenimport paramiko --hiddenimport scp --hiddenimport subprocess ^
--hiddenimport shutil --hiddenimport multiprocessing --hiddenimport etcd3 --hiddenimport redis ^
--add-data DOTNET/System.Collections.NonGeneric.dll;. ^
--add-data DOTNET/System.Collections.Specialized.dll;. ^
--add-data DOTNET/System.Collections.dll;. ^
--add-data DOTNET/System.Threading.Overlapped.dll;. ^
--add-data DOTNET/System.Threading.Tasks.Parallel.dll;. ^
--add-data DOTNET/System.Threading.Tasks.dll;. ^
--add-data DOTNET/System.Threading.Thread.dll;. ^
--add-data DOTNET/System.Threading.ThreadPool.dll;. ^
--add-data DOTNET/System.Threading.Timer.dll;. ^
--add-data DOTNET/System.Threading.dll;. ^
--add-data DOTNET/api-ms-win-crt-runtime-l1-1-0.dll;. ^
--collect-all tornado --collect-all webssh ^
-n fareAnalyzer main.py
```

图 8　PyInstaller 打包指令

```
25052 INFO: Graph cross-reference written to D:\ipric_v1_fe_guitools\build\fareAnalyzer\xref-fareAnalyzer.html
25069 INFO: Appending 'datas' from .spec
25100 INFO: checking PYZ
25106 INFO: Building because toc changed
25107 INFO: Building PYZ (ZlibArchive) D:\ipric_v1_fe_guitools\build\fareAnalyzer\PYZ-00.pyz
26011 INFO: Building PYZ (ZlibArchive) D:\ipric_v1_fe_guitools\build\fareAnalyzer\PYZ-00.pyz completed successfully.
26024 INFO: checking PKG
26026 INFO: Building because D:\ipric_v1_fe_guitools\build\fareAnalyzer\PYZ-00.pyz changed
26026 INFO: Building PKG (CArchive) fareAnalyzer.pkg
26040 INFO: Building PKG (CArchive) fareAnalyzer.pkg completed successfully.
26041 INFO: Bootloader z:\python37\lib\site-packages\PyInstaller\bootloader\Windows-64bit\run.exe
26042 INFO: checking EXE
26042 INFO: Building because console changed
26043 INFO: Building EXE from EXE-00.toc
26043 INFO: Copying bootloader EXE to D:\ipric_v1_fe_guitools\build\fareAnalyzer\fareAnalyzer.exe
26048 INFO: Copying icon to EXE
26049 INFO: Copying icons from ['D:\\ipric_v1_fe_guitools\\app.ico']
26050 INFO: Writing RT_GROUP_ICON 0 resource with 20 bytes
26050 INFO: Writing RT_ICON 1 resource with 67624 bytes
26052 INFO: Copying 0 resources to EXE
26052 INFO: Embedding manifest in EXE
26053 INFO: Updating manifest in D:\ipric_v1_fe_guitools\build\fareAnalyzer\fareAnalyzer.exe
26054 INFO: Updating resource type 24 name 1 language 0
26055 INFO: Appending PKG archive to EXE D:\ipric_v1_fe_guitools\dist\fareAnalyzer\fareAnalyzer.exe 可执行文件的路径
26059 INFO: Fixing EXE headers
26215 INFO: Building EXE from EXE-00.toc completed successfully. 可执行文件构建成功
```

图 9　PyInstaller 打包生成可执行的桌面程序

图 10　运维管理平台桌面程序

3.2 效果比对

将基于新架构的运维管理平台桌面程序与传统的运维管理平台 Web 应用，从部署位置、安全性、开发效率、部署复杂度、升级灵活性及跨平台性等方面进行效果比对，比对结果如表 2 所示。

综上，基于新架构的运维管理平台桌面程序与传统的运维管理平台 Web 应用相比，虽然在开发效率和跨平台性方面有微弱劣势，但在安全性、部署复杂度、升级灵活性方面都有巨大的优势。

4 结论

本文针对安全云桌面的使用场景，提出了一种运维管理平台的架构设计方案，通过本方案，可以在保证信息安全的基础上，实现一个易开发、易使用、易迁移、易维护的运维管理平台。具体来说，该架构具有如下优点：

（1）系统安全。将不承担生产功能的运维管理平台由生产网络剥离至非生产网络，将需要一直开启的网络端口改为“随用随开、随走随关”的方式，最大限度地降低了信息安全风险。

（2）节能高效。运维管理平台从生产网络剥离，意味着对各种人力物力资源的节约。开发采用最常用的 B/S 架构，意味着种类繁多的程序组件库可以使用，同时高效成熟的框架将带来高效的开发效率。

（3）迁移复用。组合使用业界已有的多种成熟工具，具有很强的跨平台兼容性，可以轻松地迁移到不同操作系统的云桌面环境，极大地提高了代码的复用性，缩小了迁移的工作量。

（4）灵活易用。部署于非生产网络、不依赖端口常驻，该运维管理平台更加方便访问，其开发及版本迭代过程也更加轻量级和易于施行，并且在高安全等级时期其使用也不会受到影响。

表 2 比对结果

比对维度	传统的运维管理平台 Web 应用	基于新架构的运维管理平台桌面程序
部署位置	服务器端，开发测试网/生产网（占用服务器资源）	用户终端的云桌面环境（不占用服务器资源）
安全性	需考虑服务器安全、网络传输安全等多层面保护	受云桌面安全机制保护，减少了外部网络攻击面
开发效率	专注于后端逻辑（约 16 人月）	在传统架构的基础上，仅需增加打包时间 0.5 人月（约 16.5 人月）
部署复杂度	涉及服务器配置、网络配置及安全加固等（约 4 小时）	用户只需下载安装即可（约 5 分钟）
升级灵活性	需考虑服务器停机时间，用户访问中断影响	直接在用户终端的云桌面环境在线更新安装包
跨平台性	浏览器兼容性好	依赖 PyInstaller 和目标平台支持

参考文献

[1] 刘中一，黄向平，杨毅，等. 分析辅助工具，数据处理系统，辅助分析方法和相关设备：中国，CN202210719970.1 [P]. 2024 - 07 - 22.

[2] 胡钧超，黄海江，张悦. 基于零信任的“一机多网”云桌面设计 [J]. 微型电脑应用，2024，40（7）：249 - 252.

[3] FUZI M F M，HAMID R S，AHMAD M A. Virtual desktop environment on cloud computing platform [C]//2014 IEEE 5th Control and System Graduate Research Colloquium. IEEE，2014：80 - 84.

[4] 高金金，李潞洋，薛俊杰. 融合云桌面资源的高性能计算集群方案研究 [J]. 软件，2024，45（3）：13 - 17.

[5] 易任重，刘晓海，廖晓昕. 一个基于 C/S 与 B/S 混合架构的应用实例解析 [J]. 计算机工程与应用，2001，37（16）：159 - 161.

[6] 张文涛，常红星. 基于 ASP. NET 的 B/S 架构下的项目管理系统的网络安全模式设计 [J]. 计算机科学，2008，35（2）：101 - 103.

[7] TU J F，GUO R F. The application reseach of mixed program structure based on client-server，browser-server and web service [C]// 2011 International Conference on Business Management and Electronic Information. IEEE，2011，1：193 - 195.

[8] 戎力. SSH 命令行帮你实现 6 种“贴心”的安全应用 [J]. 计算机与网络，2018，44（5）：30 - 33.

[9] 林航. Docker 容器技术在微服务架构中的应用 [J]. 网络安全和信息化，2024（7）：59 - 62.

[10] 韩莉，李雪岗. 计算机软件开发中 Docker 技术的运用探究 [J]. 软件，2024，45（4）：74 - 76.

[11] 李磊. Docker 容器技术在构建计算机实验平台中的应用 [J]. 网络安全和信息化，2024（6）：114 - 116.

[12] PERALTA J H. Microservice APIs：Using Python，Flask，FastAPI，OpenAPI and More [M]. New York：Simon and Schuster，2023.

[13] LATHKAR M. High-performance web apps with FastAPI [M].

Apress, 2023.

[14] 赫特兰. Python 基础教程 [M]. 凌杰，陆禹淳，顾俊，译. 北京：人民邮电出版社，2010.

[15] LUTZ M. Programming python [M]. O'Reilly Media, Inc., 2010.

[16] 王瑞文，王圣辉，边润根，等. Python 程序打包成 exe 可执行文件的方法探究 [J]. 无线互联科技，2017 (12)：52 - 53.

(收稿日期：2024 - 08 - 22)

作者简介：

王中华 (1972 -)，男，博士，高级工程师，主要研究方向：民航信息化技术。

张鹏 (1978 -)，通信作者，男，硕士，高级工程师，主要研究方向：民航信息化技术。E-mail：pzhang@ travelsky. com. cn。

刘中一 (1987 -)，男，硕士，高级工程师，主要研究方向：民航客票运价系统、密集计算系统。

(上接第 9 页)

[37] ZHANG P Q, TIAN G K, DONG H Y, et al. Research on network intrusion detection based on whitening PCA and CNN [C]// 2023 the 7th International Conference on Smart Grid and Smart Cites, 2023：232 - 237.

[38] 张利隆. 基于高斯过程和混合模型的工控入侵检测技术研究 [D]. 太原：太原理工大学，2021.

[39] GUO S T, LIU Y. Network intrusion detection method combining class balance and CNN [J]. Computer Applications and Software, 2023, 40 (7)：326 - 332.

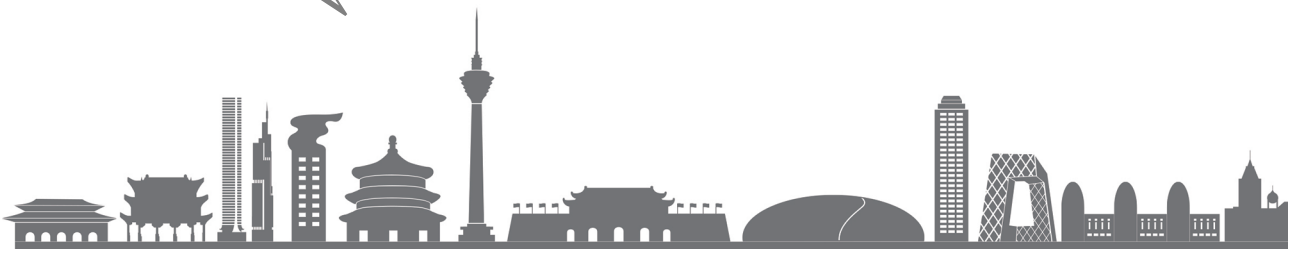
(收稿日期：2024 - 07 - 30)

作者简介：

张茜 (1992 -)，女，硕士，工程师，主要研究方向：网络安全。

王晓菲 (1990 -)，女，博士，高级工程师，主要研究方向：网络安全。

王亚洲 (1994 -)，男，硕士，工程师，主要研究方向：网络安全。



版权声明

凡《网络安全与数据治理》录用的文章，如作者没有关于汇编权、翻译权、印刷权及电子版的复制权、信息网络传播权与发行权等版权的特殊声明，即视作该文章署名作者同意将该文章的汇编权、翻译权、印刷权及电子版的复制权、信息网络传播权与发行权授予本刊，本刊有权授权本刊合作数据库、合作媒体等合作伙伴使用。同时，本刊支付的稿酬已包含上述使用的费用，特此声明。

《网络安全与数据治理》编辑部