Securely replacing VDI for development workflows

Why VDI is not appropriate for development workflows, and how to replace or augment them to improve developer productivity without compromising security.

Table of content

Optimizing remote desktop access for development workflows

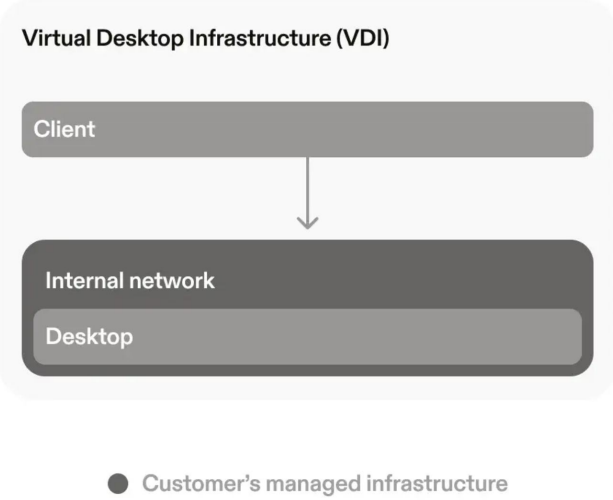
To put it bluntly, VDIs damage developer experience. They are a general-purpose security solution that have never been tailored to software development practices. Yet so many developers are forced to use VDIs, especially among regulated industries like financial services, health care and telco.

The answer doesn't have to be to petition to remove your VDI or worse, engage in insecure software practices. It's entirely possible to both improve developer productivity and keep security teams happy without compromising compliance certification such as HIPAA or ISO27001.

This white paper will explore what makes VDI solutions the wrong choice for software development, when to augment your VDI solution, and why Gartner says that **by 2027, 40% of organizations in highly regulated industries will mandate CDEs**. Let's first get aligned on what a VDI actually is.

What's a VDI?

VDIs offer a virtualized solution for accessing your desktop from a different device, focusing solely on delivering a consistent desktop experience. They are not designed to enhance developer experience or productivity in application use or coding. They are known for providing consistent desktop user experiences across different devices and locations, making particularly useful beneficial for organizations with BYOD (bring your own device) policies.



Organizations leverage VDIs for several reasons

- Remote work
- Desktop management
- Enhanced security
- Onboarding and off-boarding employees and contractors

Although primarily used for remote work, VDIs are also valuable for both onsite and offsite developers due to their security benefits:

- **Centralized data management:** VDIs centralize data storage, rather than on individual local devices, aiding in better data security and regulatory compliance.
- **Enhanced admin / access controls:** organizations can enforce strict access controls, allowing only authorized users to access sensitive data.
- **Improved compliance management:** VDIs provide a controlled environment where data handling and processing can be monitored and audited more effectively.
- **Secure remote access:** VDIs ensure secure, encrypted connections between remote devices and the central data center they are accessing, appealing to onsite teams engaging in hot desking.
- **Reduced endpoint security risks:** with VDIs, the attack surface is significantly reduced because the actual computing and data storage occurs in secure data centers, not on local devices.

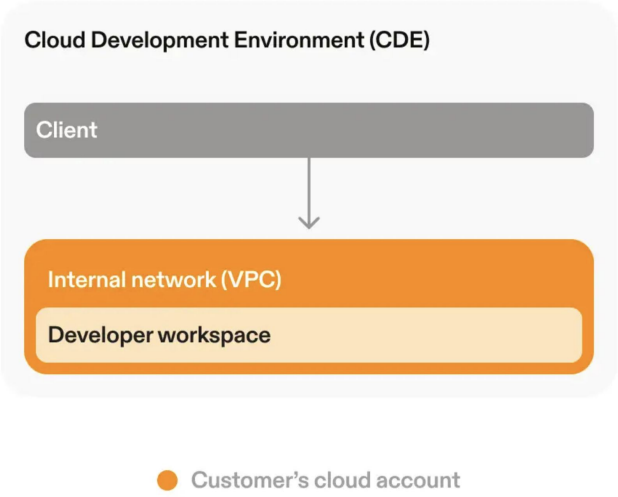
Now for a quick reminder on CDEs.

What's a CDE?

Cloud development environments are secure, standardized and automated development environments. They can act as your distribution channel for

development tools, ensuring developers only have access to 'business approved' tools. They integrate seamlessly with a developer's preferred IDE, automatically apply your organization's configurations across all environments and optimize resources efficiently.

CDEs are the only tool secure enough to replace your VDI, without compromising developer experience.



'It's anti-developer. Period. Full stop.'

As discussed, virtual desktop infrastructures are a form of virtualization that enables remote access to a full desktop environment. VDIs host the operating system, applications and data from a desktop environment on a virtual machine and stream them to end-users over a network.

In theory, VDI setups should not interfere with developer experience. Their purpose is to duplicate a desktop in order to provide remote access to end users, or limit the actions that end users are able to take when accessing sensitive information (i.e. preventing users from downloading information to their laptops, etc.). But in practice, developer experience is significantly impacted.

Developer experience challenges when using VDIs

Below are some of the challenges developers face with their VDI setups. Some related to inherent challenges with VDI technology and others coming from how organizations set up their VDI solutions, and the policies they have in place around them.

Constantly losing state

VDIs can be ‘persistent’ and ‘non-persistent’. This refers to how user data and customization settings are managed and stored across sessions. In persistent setups, VDIs maintain state across sessions.

In non-persistent setups, environments do not save user states between sessions. The advantages to this are mainly cost savings but as you can imagine, for developers, removing the customizations they make such as installing tools or configuring their editors can lead to hours wasted on a daily basis.

Incorrect operating systems

VDI solutions are often provisioned with windows machines while many development tools are often optimized for Linux-based operating systems (OS). If the developer applications being used depend on Linux as the OS, workarounds can be put in place, but this will often lead to ‘works on my machine’ issues where the developers’ local environments end up leaking bugs to production because they are different from the upstream environments.

Inability to use dual monitors or larger screens

Using multiple screens, or larger screen resolution, is often limited by the screen resolution provided from a VDI.

Application installation restrictions

Developers may encounter restrictions on installing software or marking certain system changes when using VDIs. This is usually due to:

- **Security and compliance:** random installations can compromise compliance by potentially improperly storing sensitive data or not following required security protocols
- **Resource management:** VDIs typically operate a shared resource environment where CPU, memory and storage are allocated based on the needs of multiple users. Uncontrolled installations can lead to resource hogging, affecting performance for others.
- **Maintenance and support:** custom installations can lead to an increase in support tickets, more complicated troubleshooting, and longer resolution times.
- **Data integrity and backups:** VDIs are often configured to revert to a known ‘good state’ at the end of a session, helping to maintain system integration and reducing the risk of persistent malware (see above on non-persistent environments). Installations could lead to data loss or inconsistencies as installed applications may be removed after a session

Latency and resourcing issues

Latency in a VDI context refers to delays between a user’s action and the response from the VDI environment. They are the number one complaint from VDI users as they are the most impactful to development workflows. Some factors that contribute to latency are:

- **Network latency:** the most significant contributor – this represents the time it takes for data to travel across the network from a client device to the VDI servers and back. This can depend on distance, network quality, and bandwidth and because many developers using a VDI are accessing servers from long distances, this becomes a noticeable issue.
- **Protocol efficiency:** In reality, VDIs are really just streaming video. They use protocols like PCoIP, HDX and RDP to transmit screen images, mouse clicks, and keystrokes between the client device and server. The efficiency of these protocols in compressing and decompressing data impacts performance.
- **Server performance:** if the servers hosting VDI environments are under high load or not properly configured, they may not process inputs and send outputs as quickly as they need. This also applies to resource contention – if the infrastructure is not adequately provisioned, users might experience increased delays during peak usage.
- **Client device performance:** although less crucial, on occasion a client device's ability to quickly decode screen updates and handle the VDI client software will impact developer experience.

The pain developers feel related to these latency issues show up in routine development activities:

- **Real-time typing feedback delays:** delays in seeing what you've typed or commands being executed .
- **Interactive development environment features:** latency can slow down features like autocompletion, syntax checking, and other real-time feedback functionality.
- **Testing changes:** VDI setups often limit the ability to access local test environments or require users to fully push to staging or pre-prod environments to see the impact of their changes – which ends up in a loop of long feedback loops waiting for your CI automation cycle adding hours to writing code.

- **Debugging:** debugging often requires stepping through code line-by-line – latency makes this process sluggish as the delay in response times when setting breakpoints or inspecting variable values can make it difficult to maintain a mental map of what you've already reviewed.
- **Code reviews:** slow screen sharing.
- **Late integration:** developers can experience disruptions to their workflows because of long wait times, developers committing changes to the same repo will experience merge conflicts.
- **Accessing remote resources:** executing database queries and waiting for responses, or accessing cloud-based APIs, storage etc can run slower.

To mitigate the issues, enterprises can adopt strategies like optimizing network infrastructure, understanding which protocols are best for their workloads, server performance and sizing, etc. However none of these mitigations will address the core of the problems surrounding developer experience because VDIs were not built for development. Enter cloud development environments.

CDEs as a replacement for VDI

CDEs are similar to VDIs in that they are both virtualized technologies enabling a remote experience. Their key difference: **CDEs were purpose-built to improve developer experience.**

CDEs can be used for:

- Faster onboarding
- Consistency and reproducibility of development environments
- Better collaboration between peers
- Faster build times for large codebases

They can also be used for the same security benefits as a VDI setup:

- **Centralized data management:** CDEs store data centrally rather than on individual local devices, helping organizations to maintain better control over data security and compliance with regulations.
- **Enhanced admin and access controls:** organizations can implement robust access control mechanisms, ensuring only authorized personnel can access sensitive data.
- **Improved compliance management:** CDEs provide a controlled environment where data handling and processing can be monitored and audited more effectively.
- **Secure remote access:** employees can access sensitive data from any location because CDEs provide a secure, encrypted connection between a remote device and the central data center they are accessing. This becomes attractive to onsite teams who are hot desking at their offices
- **Reduced endpoint security risks:** with CDEs, the attack surface is significantly reduced because the actual computing and data storage occurs in secure data centers, not on local devices.

Given these benefits, CDEs are the recommended solution for enabling secure, remote development in a way that doesn't hinder typical developer workflows. If you absolutely must use VDIs because of organizational mandates, keep reading.

When should VDIs and CDEs be used together?

The rule of thumb is simple: your organization should rely exclusively on VDIs only if it does not support any developers. In all other scenarios, your organization should rely on both CDEs and VDIs if they require use of a remote desktop. If they do not require use of a remote desktop, CDEs without VDIs check all the security boxes and provide a better overall experience.


Gitpod