Incredibuild Technology
Overview

INCREDIBUILD

Contents

Overview 3

Five Steps to Start 3

Drill Down: Virtualized Distributed Processing™ 4

Virtualized Distributed Processing™ Flow 5

Incredibuild Advantages 13

About Incredibuild 14

Overview

Incredibuild dramatically accelerates compilation, tests, and other compute-intensive tasks by seamlessly and concurrently distributing workload processes across idle CPUs in your local network or in the cloud. Essentially, Incredibuild seamlessly transforms each compute host into a supercomputer with hundreds of cores and gigs of memory.

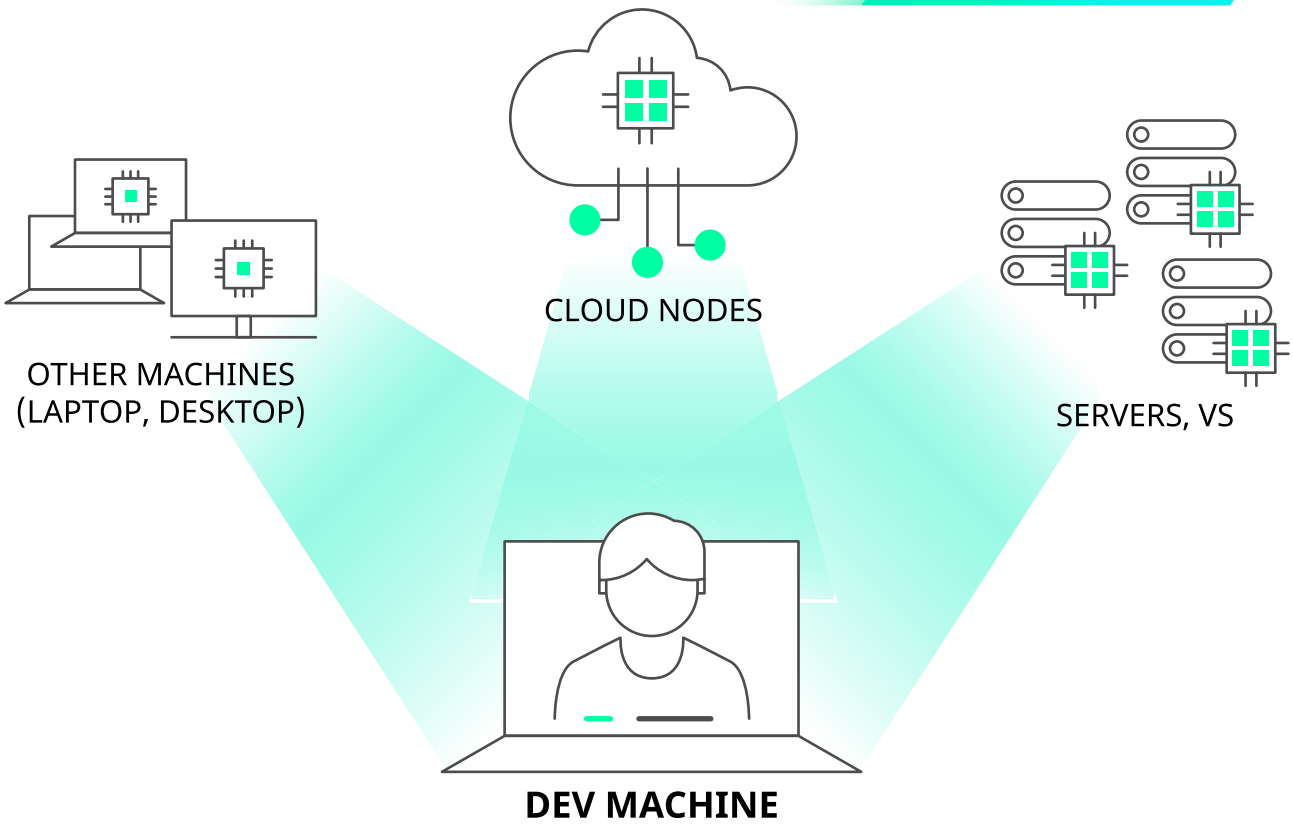
Five Steps to Start

Benefitting from Incredibuild's performance boost is super easy and simple, owing to our Virtualized Distributed Processing™ technology - which combines the genericness of virtualization with the horsepower of distributed processing. This allows Incredibuild to seamlessly distribute a process from a host machine to a remote machine - emulating the relevant parts of the host environment in real-time. All this without installing anything on the remote machine, accept for a lightweight Incredibuild Agent.

What's more - since Incredibuild's technology is generic - it can distribute almost any kind of process to remote machines: Compilations, tests, simulations, or your own custom workloads.

Here's how easy it is to start accelerating your workloads with Incredibuild:

1. Choose a machine on which to install an Incredibuild Coordinator and load an appropriate Incredibuild license to it. The coordinator is the component that coordinates between the hosts in an Incredibuild environment.
2. Install an Incredibuild Agent on each host you want to be part of the environment.
3. Enter the process names you'd like to distribute across remote machines or start working immediately with built-in integrations, like the Incredibuild for Visual Studio add-on.
4. Run your workload with Incredibuild.
5. Use Incredibuild's graphical visualization to see your processes being seamlessly distributed to hundreds of additional cores.



IIncredibuild architecture: Agents installed on each host are connected to a centralized coordinator. Each host machine with an agent can use the aggregated idle power of all the other machines in the Incredibuild environment.

Drill Down: Virtualized Distributed Processing™

The backbone of Incredibuild's offering, Virtualized Distributed Processing™ enables a workload that consists of multiple, concurrent processes to be automatically and dynamically distributed to idle CPUs on remote machine across your network or public cloud.

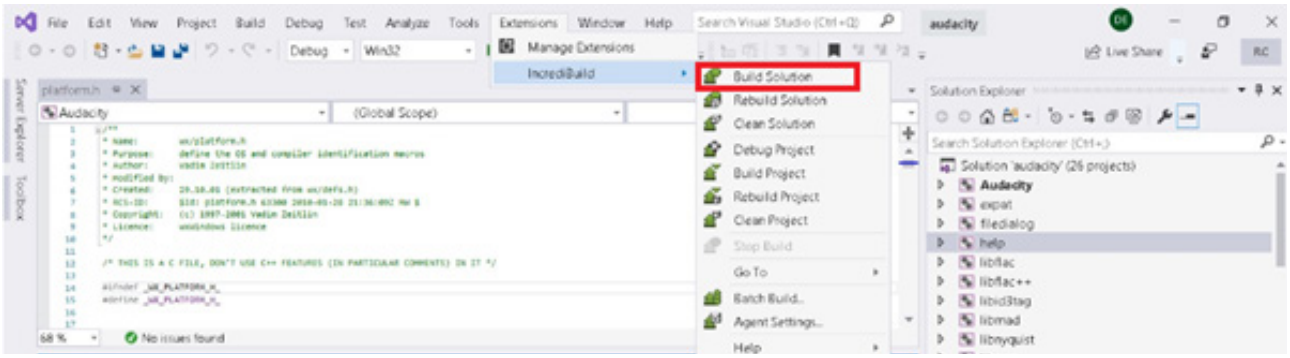
Virtualized Distributed Processing™ can even use idle CPUs on remote machines - while users are working on them - operating in the background, without interfering with work. In organizations that have hundreds of machines, the aggregated number of idle CPUs in any given moment can easily be in the thousands. These are wasted cores that Incredibuild recaptures to accelerate time-consuming workloads in need of computing power.

Incredibuild runs processes on remote machines in a secure sandbox. Everything each process requires to run properly is dynamically emulated from the local host to the remote machine. Any output generated by the process - std output, errors, return codes, files generated, etc. - is automatically synced back to the local host, as if the process had been executed locally.

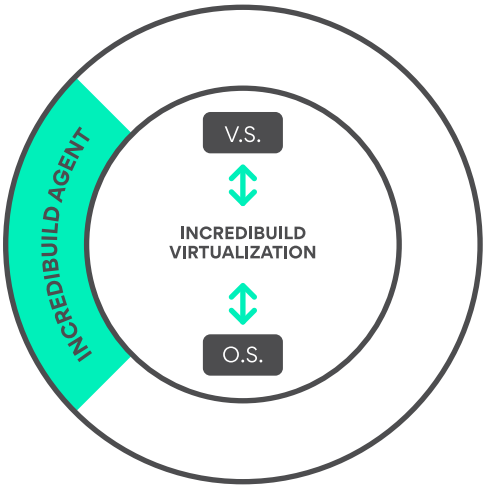
Virtualized Distributed Processing™ Flow

Here's an example of how Virtualized Distributed Processing™ accelerates Visual Studio C++ compilations by distributing the many compilation tasks to remote machines across a network. Note that this high-level explanation omits some details that occur during the process. However, the steps illustrated are basically identical for any other build system, compiler, or workload accelerated with Incredibuild.

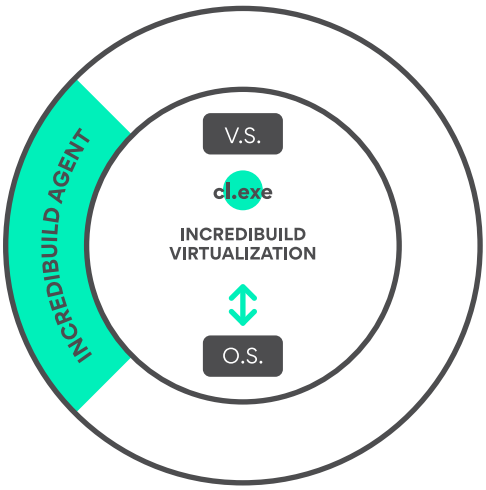
1. The Visual Studio C++ build is executed through the Incredibuild addon for Visual Studio.

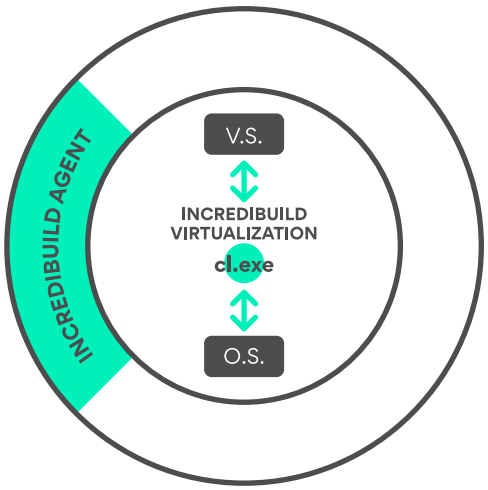


2. Incredibuild forwards the build request to Visual Studio IDE and injects a virtualization layer between Visual Studio and the operating system. All operating system calls from Visual Studio now pass through Incredibuild.

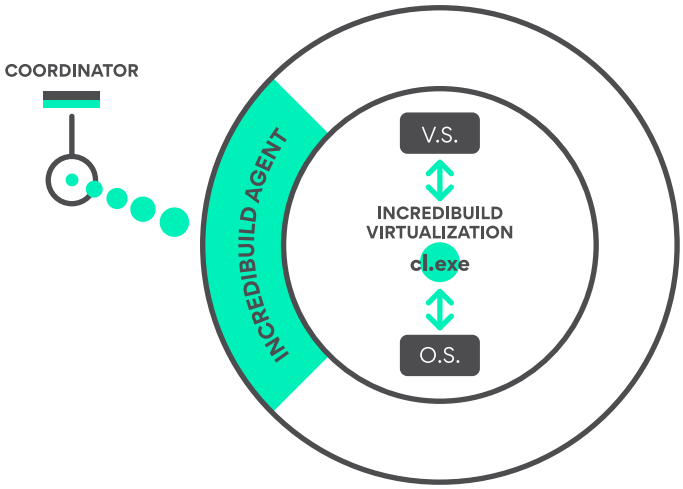


3. Incredibuild watches for CreateProcess calls from Visual Studio to the operating system, which ask the OS to create processes as part of the compilation workload. When Visual Studio executes a compilation process (the Visual Studio native C++ compiler process name is cl.exe), Incredibuild intercepts the CreateProcess call before it reaches the operating system.





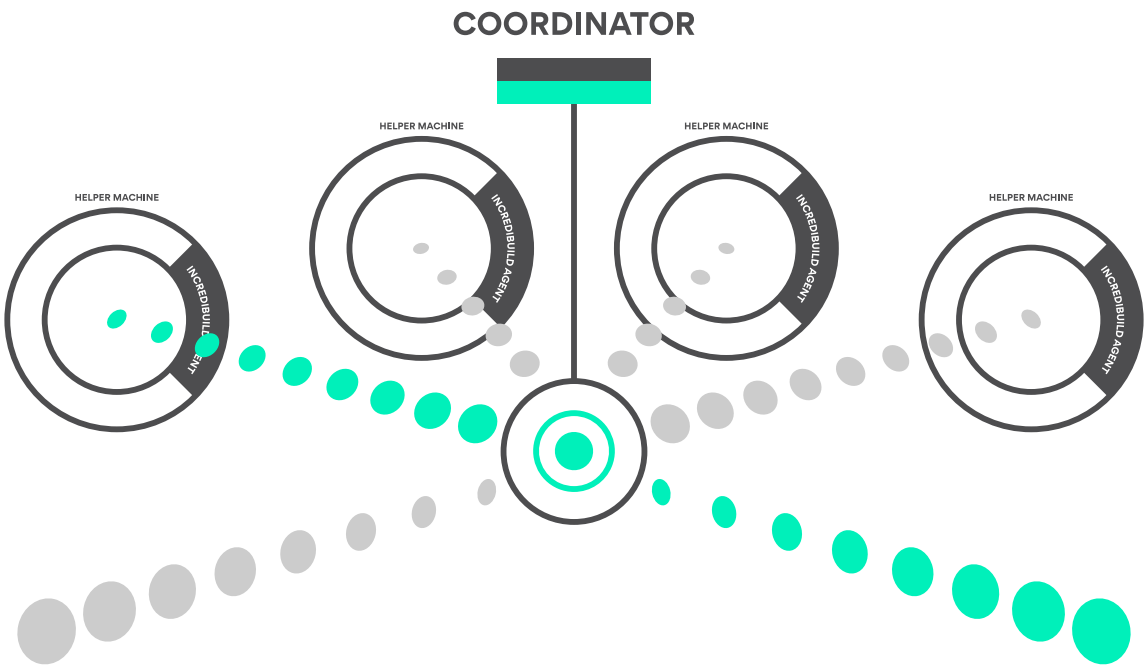
4. Incredibuild detects cl.exe as a process predefined to run on remote machines if possible, and does not allow the request to create the compilation process to reach the operating system.



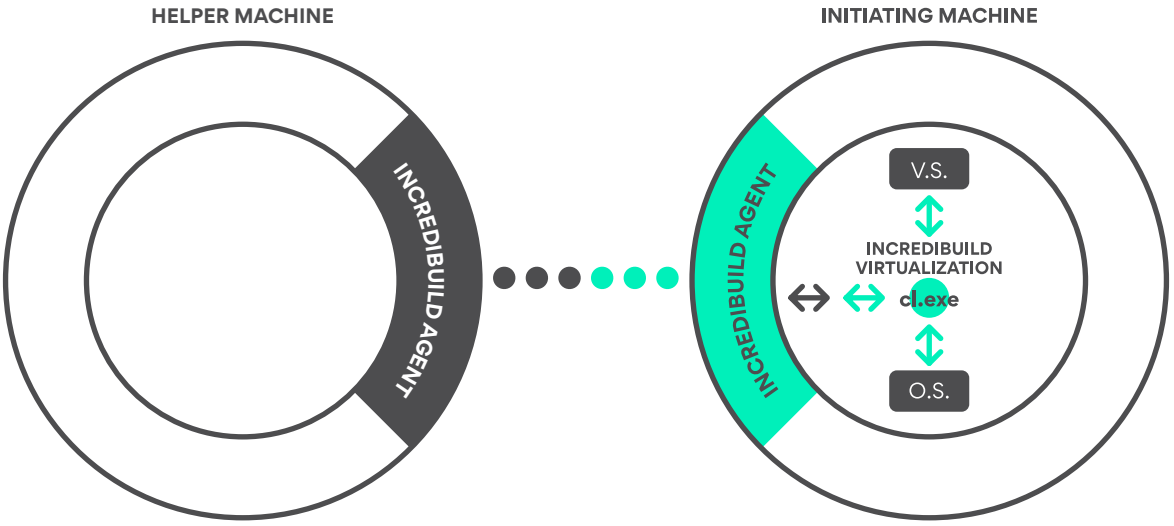
5. Instead of letting the operating system execute the cl.exe process as Visual Studio instructed, Incredibuild requests the coordinator component for an available remote machine on which an agent is installed. At least one of its cores is available, on which Incredibuild can remotely execute the compilation process.

Each Incredibuild environment contains a single coordinator responsible for managing the environment and deciding which machines are best suited to help accelerate a given workload.

6. The Incredibuild Coordinator, which has many agents connected to it, locates the best available remote machine(s) in the Incredibuild environment. Then it creates a handshake between the remote machine and the local host machine.



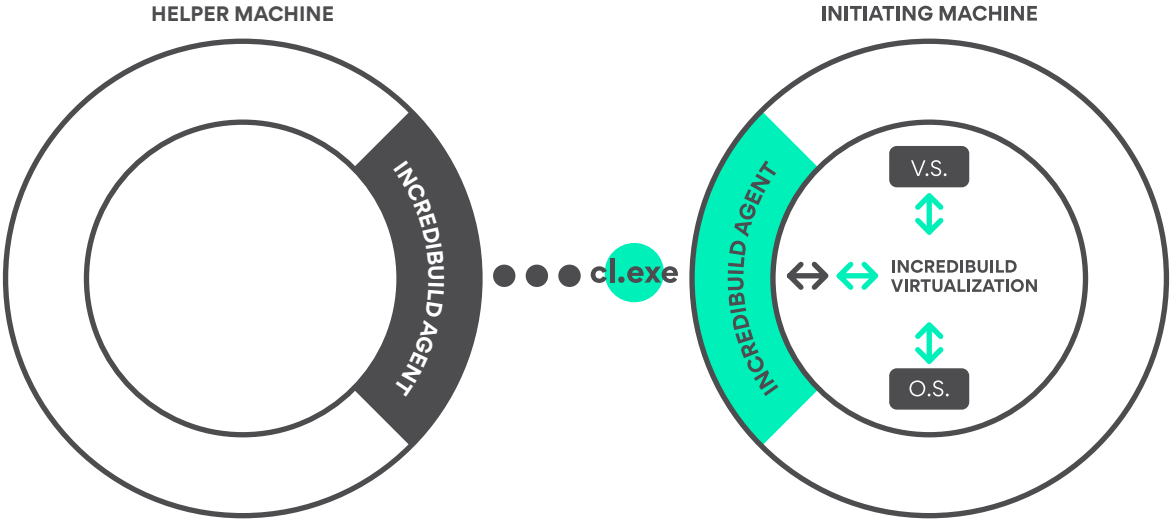
7. Once the Incredibuild Coordinator creates the handshake between the local host machine and the remote machine, it is no longer required for the flow of distributing compilation tasks to remote machines. The Incredibuild service running on the local host machine communicates directly with the service running on the remote machine.



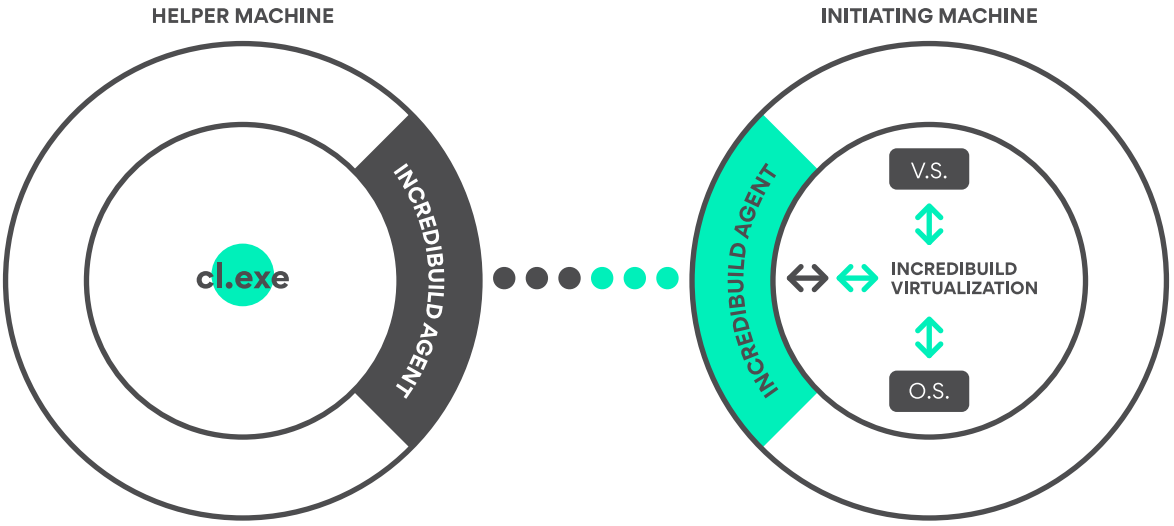
8. Once the Incredibuild Agent on the local host machine gets a remote machine on which it can execute the compilation process, Incredibuild informs Visual Studio that the compilation process it requested was executed. Yet, instead of a handle to the real process, Incredibuild returns a dummy process handle to Visual Studio. Visual Studio now understands that a compilation process is running on the local host machine.

In parallel, the Incredibuild Agent on the local host machine sends the compilation command (CreateProcess), the local host environment variables, and the compilation process file itself (cl.exe) to the remote machine using a TCP/IP communication protocol.

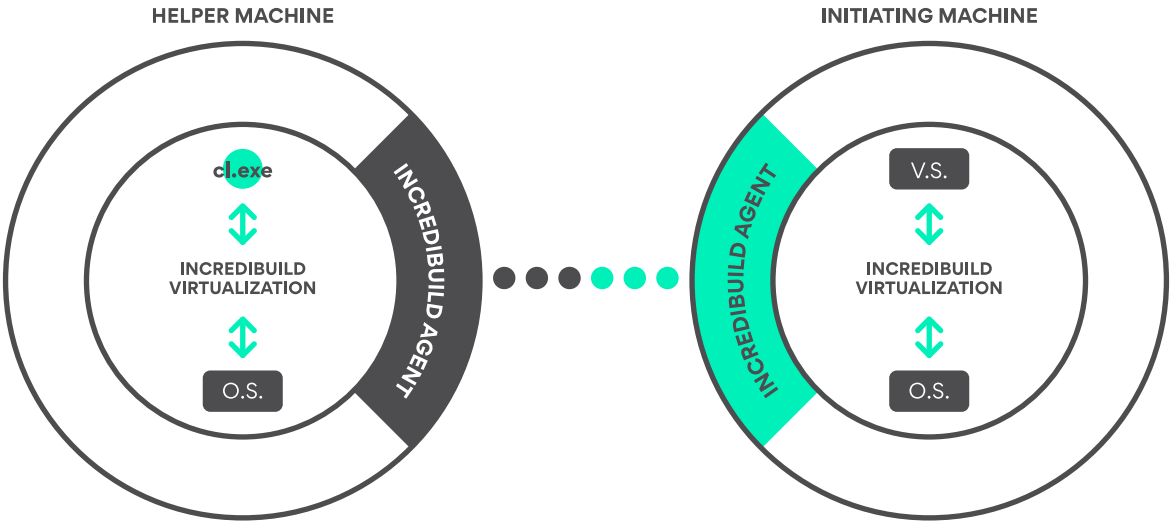
Files sent to remote machines are stored in a special Incredibuild cache so that the next time a compilation task is executed on that remote machine, the files won't need to be resent if they haven't changed. This means that after the Incredibuild environment is used for a while, the remote machines' cache becomes warm, and only a minimal number of files need to be re-synced.



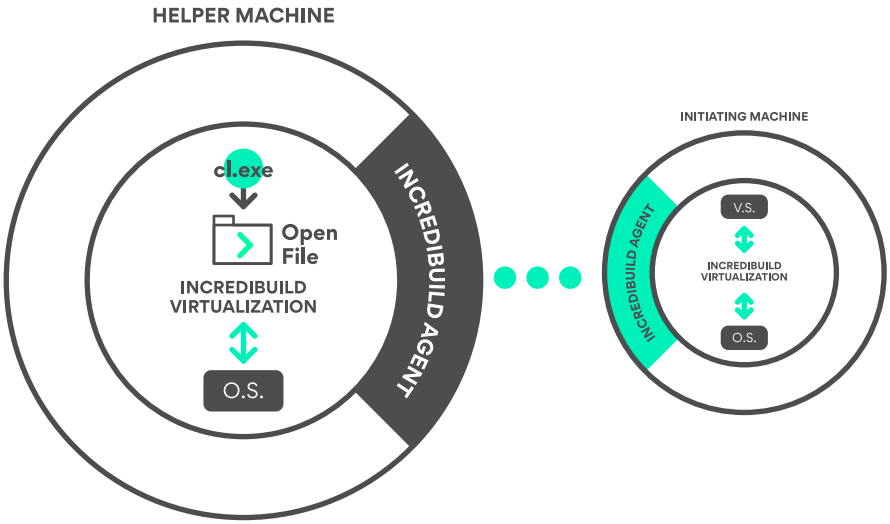
9. The Incredibuild Agent on the remote machine executes the cl.exe CreateProcess command in a special sandbox, using the local host machine's environment variables. If a user is currently working on the remote machine while Incredibuild is harvesting its unused CPU power, they won't even know that Incredibuild is accelerating another workload. If the user of the remote host executes a task that requires all of the machine's CPU cycles, Incredibuild seamlessly re-executes the compilation task on another available machine.

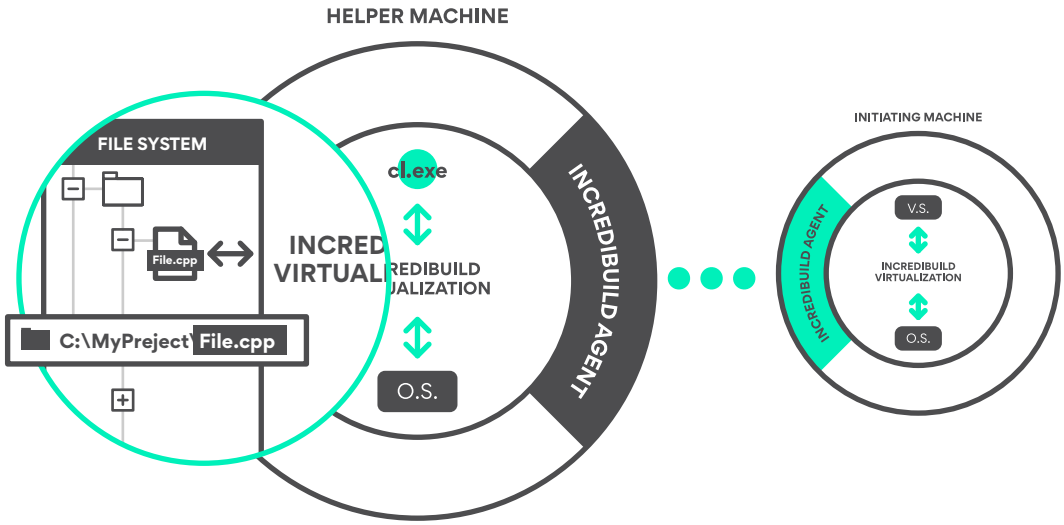


10. When the Incredibuild Agent on the remote machine executes the compilation process, the virtualization layer is once again injected into the process (cl.exe). It intercepts all calls that the cl.exe process sends to the OS.

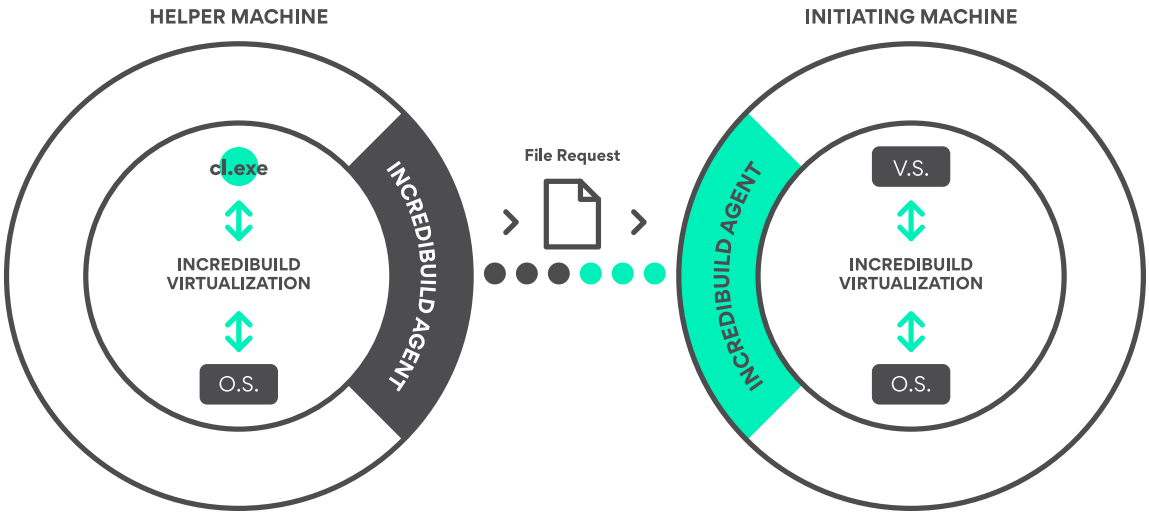


11. On the remote machine, the compilation process sends many calls and requests to the remote machines' operating systems. Each file-system-related call to the OS is intercepted by Incredibuild. It cannot be performed directly by the OS because the files that the cl.exe process requests are not available on the remote machine. For example, when the compilation process sends an Open File request, it passes through Incredibuild. Incredibuild knows that the file (and the path to the file) that needs to be opened is not available on the remote machine but rather on the local host machine that initiated the process execution.

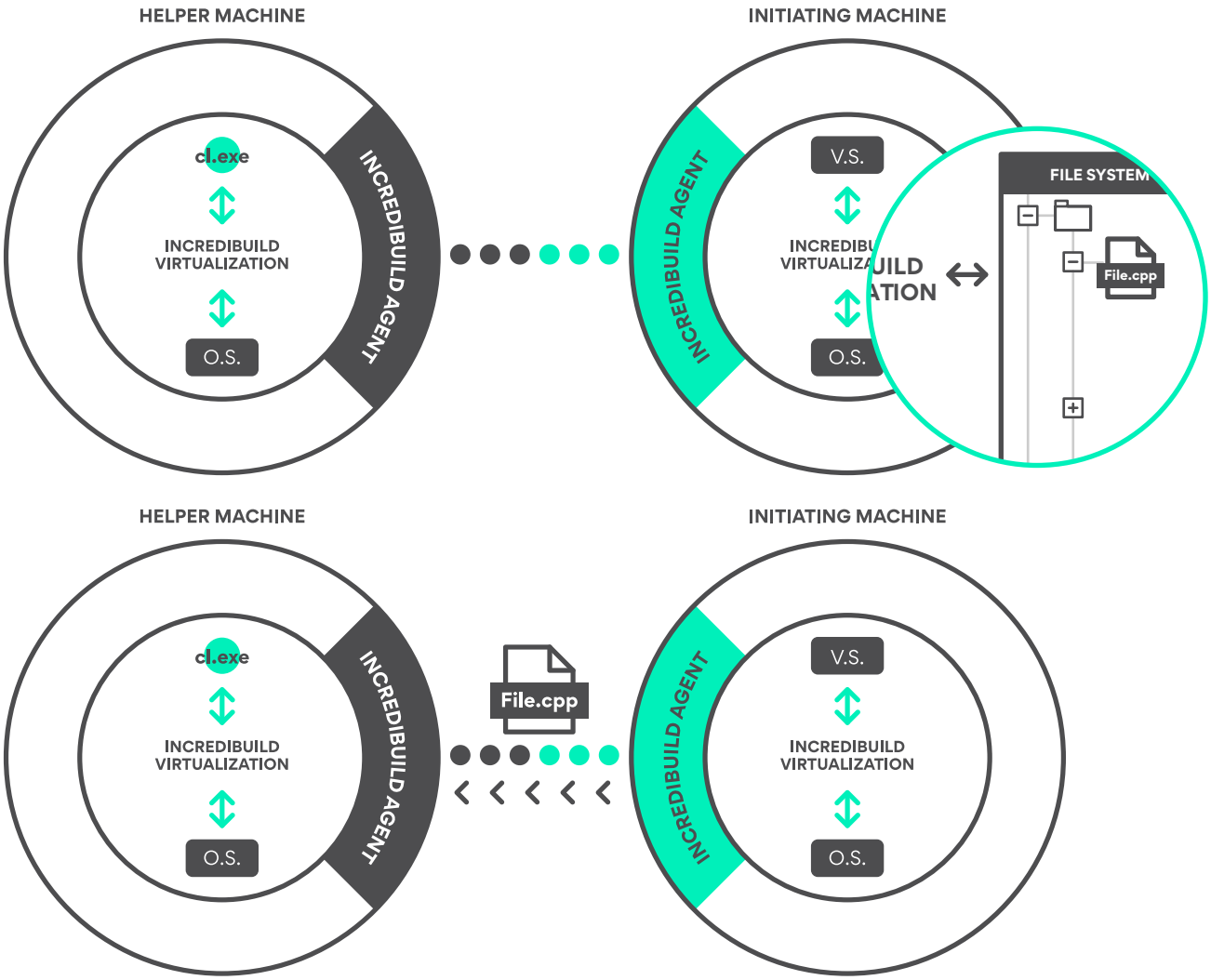




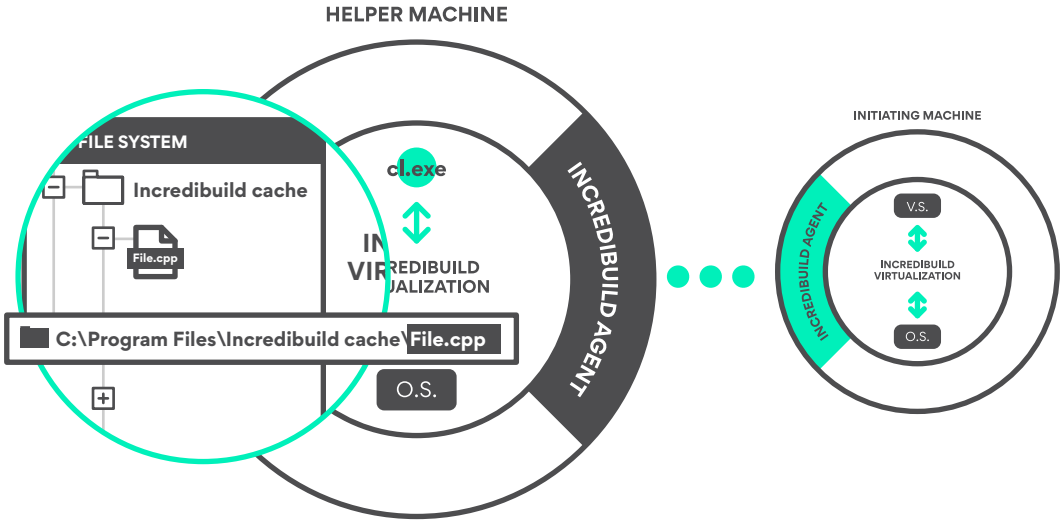
12. The Incredibuild Agent on the remote machine then requests the local host machine's agent to provide the file.



13. The requested file is copied from the local host machine's file system to an Incredibuild cache folder on the remote machine.

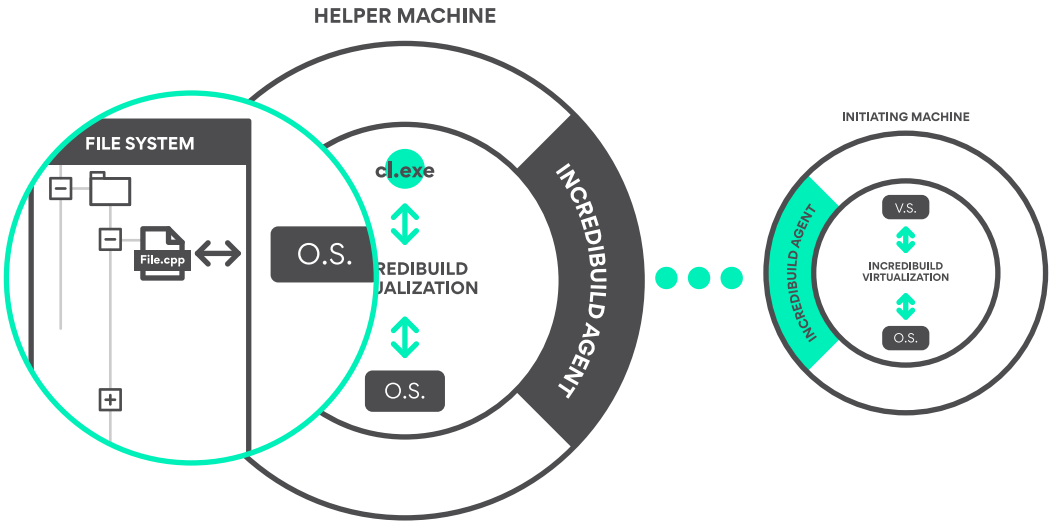


14. Once the file is successfully copied to the Incredibuild cache folder on the remote machine, Incredibuild manipulates the Open File call. The file the OS is asked to open is the file in the Incredibuild cache folder, rather than the original file location.

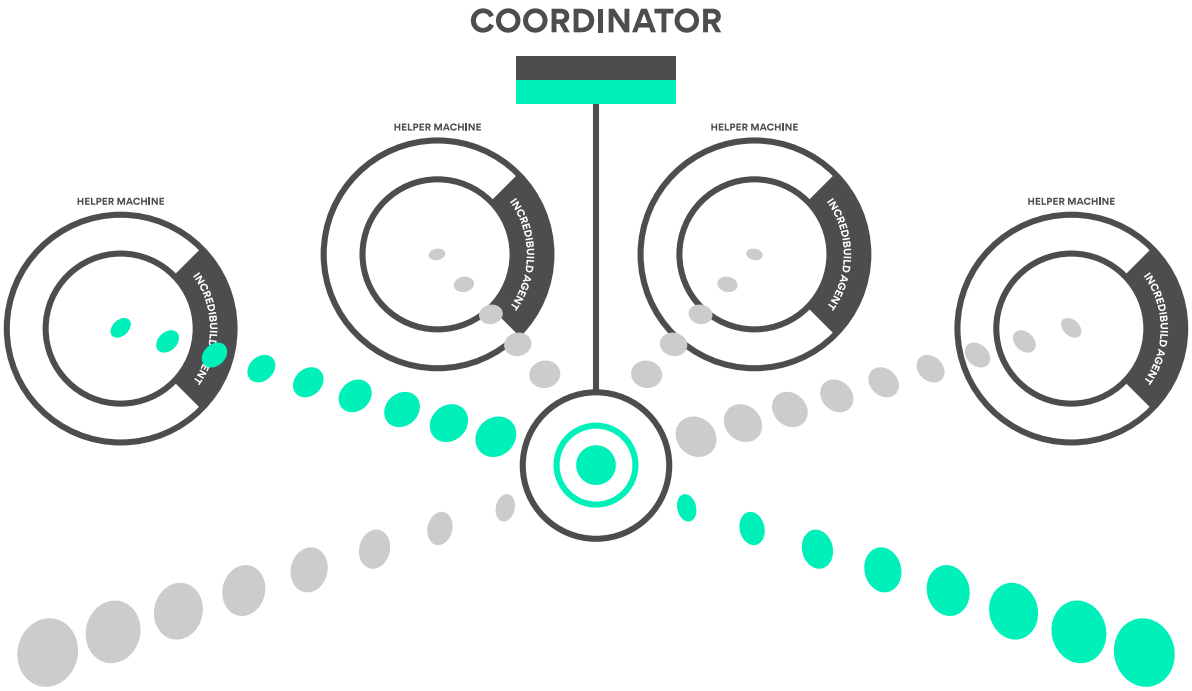


15. The operating system locates the file in the Incredibuild cache folder, opens it, and returns a file handle to Incredibuild, which forwards it to the compiler process. The result: The OS opens a valid file, the compilation process receives a valid handle for the file it requested, and the compilation process continues – all while Incredibuild continues to emulate the local machine's environment.

The same virtualization layer is applied for other file-system events such as Registry calls, DLL loading, the running of executables, and other OS calls. Incredibuild handles StdOut, StdErr, and StdIn for remote processes in a similar manner, along with any files generated by the processes executed on remote machines.



16. From the user's perspective - when distributing many compilation processes in parallel to many remote machines and cores, it's as though the host machine is a supercomputer with hundreds of cores – and the compilation workload is executed dramatically faster.



The flow above illustrates how Incredibuild's Virtualized Distributed Processing™ technology is generic and can be applied to any kind of process - not just compilation. This genericness allows users to accelerate many different types of compute-intensive workloads, as long as they consist of multiple processes that can run in parallel. This allows users to use the same Incredibuild environment and their existing hardware to accelerate a wide variety of the compute-intensive, concurrent, multi-process workloads that are highly common in the dev ecosystem: Compilations, unit testing, integration tests, QA scripts, packaging, data conversion, compression, code generation, simulations, graphical tasks, and many others.

Incredibuild Advantages

Virtualized Distributed Processing™ makes working with Incredibuild "IT Heaven" – with maintenance effort literally near zero. There is no need to retain VM image banks or copy files, scripts, or toolchains to remote machines. You can even work simultaneously with different OS flavors. Incredibuild does all the work for you, seamlessly.

Incredibuild also keeps both hardware and software costs in check. Despite the massive gain in performance and time to market, **with Incredibuild, there's no need to install anything on remote machines; expect a lightweight Incredibuild Agent**. This means if you accelerate Visual Studio compilations, you don't need to install Visual Studio, its toolchain, or your source code on the remote machines. These machines will contribute their idle CPU cycles to your Incredibuild environment regardless of what software they have installed. Moreover, since Incredibuild's technology is generic, you can use the same environment to accelerate many different compute-intensive workload types.

Here are a few more incredible Incredibuild productivity advantages:

- There's no need to change your tools, processes, or the way you work to benefit from Incredibuild. It will simply charge your workloads with additional compute power.
- Different flavors of OS can co-work together - such as Windows 10, Windows Server 2019, or Linux, Ubuntu, and Fedora.
- In the same way that you don't need to install anything on the remote machines, if a remote machine has Visual Studio 2015 installed, it can seamlessly be used to accelerate a local host's Visual Studio 2019 compilations. This is because the Visual Studio 2015 installed on the remote machine will not be considered when Incredibuild operates inside its special remote sandbox.
- To upgrade an Incredibuild version, you only need to upgrade the Incredibuild Coordinator component, which will upgrade all the Agents on your Incredibuild environment automatically.

See Incredibuild in action!

Learn from our customers and get ideas on how you can use Virtualized Distributed Processing™to boost your performance, productivity, and time to market.

About Incredibuild

Incredibuild turbocharges development from compilations to testing and release automation by turning every host into a super computer with hundreds, even thousands, of cores. World leading brands like Microsoft, Amazon, Citibank, Adobe, Disney, Intel, Nvidia, EPIC Games, Nintendo among others rely on Incredibuild to deliver better products to market, radically faster.