Cloud Compiling for z/OS

Secure, Cloud-Based Compiling Cuts License Fees and CPU Usage

The need has never been greater for mainframe enterprises to reduce expenses while leveraging existing equipment and best utilizing workforce expertise. Cloud Compiling offloads compiling jobs to a private or external cloud, eliminating redundant licensing costs by 50% or more and slashing z/OS compiler-related CPU usage by up to 70%.

Cloud Compiling for z/OS has two compiling options. Customers can choose a cloud-based, off-site data center to process compiling jobs or an in-house compiler that uses an existing internal network.

Cloud Compiling transmits the source code and compile job to a virtualized z/OS language compiler and returns the object code and listing to the target data sets.

With no IPL, migration, or changes to source code, customers retain control over automated tasks without additional administration. The entire process is seamless and invisible to z/OS programmers.

Most importantly, instead of licensing compilers to run on multiple LPARs, Cloud Compiling for z/OS streamlines the process, reducing monthly license charges (MLC) by at least 50%. It is maintained on only one LPAR, which recovers capacity on all other LPARS that no longer need a compiler and further reduces costs.
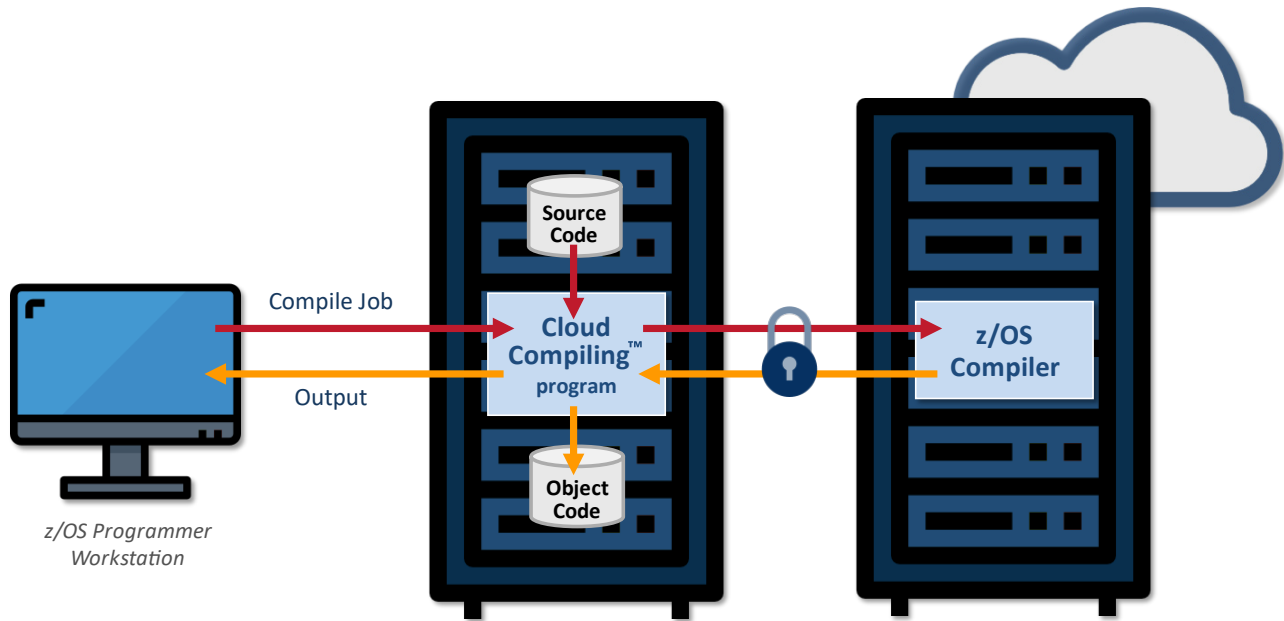
The single-destination compiling architecture eliminates multiple installs and the need for repeated testing. This allows the IT team to focus on other priorities.

Initial installation involves no up-front investment and is as easy as downloading a zip file. Typically, the trial version of the software is running within an hour; cost savings will be reflected in the first billing cycle.

Reliable Security from Cloud Compiling

- Access is controlled by standard security products – RACF, ACF2, Top Secret – and enhanced technologies such as PassTickets.

- A single user ID, restricted user privileges, and limited authority for data sets reduce hacking points.

- Customers can further secure data in transit with SSL/TLS and VPN.

- Old and failed jobs are deleted.

- The software does not access application files or databases.

- Customer source code is not stored on the compile machine.

- All access is initiated by the customer, not the cloud.

- Off-site data centers are hosted by world-class cloud service providers with advanced, audited security protocols.

- For the on-premises compiler option, source code never leaves the internal network.

Cloud Compiling eliminates redundant compiler licenses and reduces CPU usage
—and delivers those dollars to the bottom line with no up-front expenses and quick ROI.

Cloud Compiling Saves Time, CPU Resources – and Money

Cuts MLC and CPU Load by Half or More

Monthly license charges are reduced by a minimum of 50%. The local CPU compiling load can be reduced by up to 70% over current usage.

Flattens 4HRA

MSU spikes associated with COBOL 6 compiling are eliminated, which levels the 4-hour rolling average (4HRA).

Minimizes IT Resources

All maintenance and upgrades are managed by Cloud Compiling, so your IT staff is available for other projects.

No Additional Workload

Installing Cloud Compiling requires no migration or changes to the existing environment.

Fast Scaling

Capacity can be added with minimal notice. Cloud destination data centers can scale quickly to avoid additional licenses or stress on existing internal resources.

Support for Multiple Languages

Cloud Compiling supports all versions of COBOL, PL/1, Fortran, and C/C++ programming languages, and can be run on any version of z/OS.

For more information about Cloud Compiling for z/OS, please visit sdsusa.com/cloud-compiling.

Quality Mainframe Software Since 1982

Software Diversified Services delivers comprehensive, affordable mainframe and distributed software with a focus on cybersecurity and compliance. Hundreds of organizations worldwide, including many Fortune 500 companies, rely on SDS software. Our expert development and award-winning technical support teams are based in Minneapolis, MN. To learn more, please visit our website.