



<https://access.redhat.com/articles/rhel9-abi-compatibility>

Compatibility exceptions

The following are exceptions to compatibility in RHEL.

SystemTap static probes

- No assurances are made at this time that integrated SystemTap static probes will continue to have the same probe name, probe location, or interpretation or number of arguments. Since the probes are primarily designed for deep analysis and debugging, the probes must be able to change as the underlying implementation changes.

Static linking with the C/C++ runtime

- Static linking with the C/C runtime is not supported. This includes linking with any files that are part of the ``glibc-static`` or ``libstdc`-static` packages. You may choose to link statically, but the resulting application binaries may fail to operate if any package in the installation is changed.

C/C++ application sanitizers

- C/C++ applications built with the compiler option `-fsanitize=[option]` cannot participate in the API or ABI guarantees provided in this document. The sanitizer



<https://access.redhat.com/articles/rhel9-abi-compatibility>

Guidelines for preserving binary compatibility

Red Hat recommends that application developers adopt the following principles in order to improve binary compatibility:

1. Use only libraries and applications listed in the compatibility level that suits your application needs.
2. Build applications using the published interfaces of a library. Non-published (internal) interfaces are subject to change at any time, which can cause instability in the dependent application if relied upon.
 - If the library provides a development package, you must install and use that development package, including any provided headers, for your own development. If the library does not provide a development package or does not provide a set of headers and shared object for linking, then the API and the ABI of that library is not covered by any compatibility guarantees.
 - You must link with all libraries that are required by your application. Failing to link against all required libraries is called “underlinking.” An underlinked application cannot take advantage of ABI compatibility.