This document discusses the challenges encountered when adapting the **bpftime** project for **CentOS 7** and proposes some solutions.

一、Direct Adaptation on CentOS 7

1.Upgrading GCC

The default GCC version in CentOS 7 is **GCC 4.8**, which lacks support for **C++20**. Since GCC started preliminary support for C++20 from **version 9**, it was necessary to upgrade GCC to version **11**.

Steps:

1.Enable the Software Collections Library (SCL) repository.

```
sudo yum install centos-release-scl
```

2.Configure a mirror source to resolve network issues.

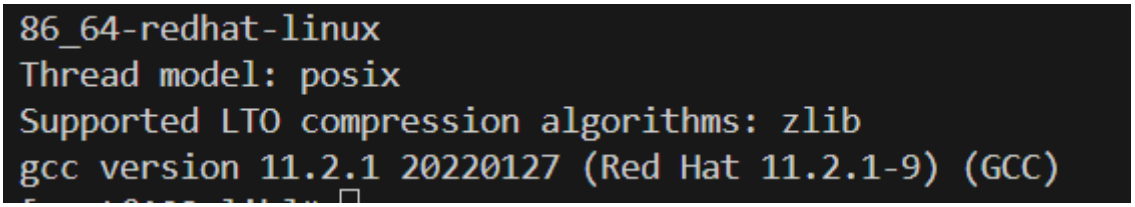
```
#Modify the following three files:
/etc/yum.repos.d/CentOS-Base.repo
/etc/yum.repos.d/CentOS-SCLo-scl-rh.repo
/etc/yum.repos.d/CentOS-SCLo-scl.repo
```

3.Address GPG key mismatches.

```
#Introducing GPGKEY
vi /etc/yum.repos.d/CentOS-SCLo-scl-rh.repo
gpgkey=https://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-
7,https://www.centos.org/keys/RPM-GPG-KEY-CentOS-SIG-SCLo
rpm --import https://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7
rpm --import https://www.centos.org/keys/RPM-GPG-KEY-CentOS-SIG-SCLo
```

4.Install GCC 11.

```
sudo yum install devtoolset-11-gcc*
scl enable devtoolset-11 bash
gcc -v
```



Successfully install GCC 11.

2.Upgrading Clang

Clang is required for compiling **eBPF programs**. Upgrading to **Clang 16** was achieved using precompiled binary packages (as source compilation resulted in errors).

Steps:

```
#Install Clang binary package.
wget https://github.com/llvm/llvm-project/releases/download/llvmorg-
16.0.3/clang+llvm-16.0.3-x86_64-linux-gnu-ubuntu-20.04.tar.xz
#Extract binary package
sudo tar -xf clang+llvm-16.0.3-x86_64-linux-gnu-ubuntu-20.04.tar.xz -C /usr/local/
sudo mv /usr/local/clang+llvm-16.0.3-x86_64-linux-gnu-ubuntu-20.04 /usr/local/clang-
16
#Configure environment variable/etc/profile
export PATH=/usr/local/clang-16/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/clang-16/lib:$LD_LIBRARY_PATH
#Verify installation
clang --version
```

There will be an error here:

```
clang: error while loading shared libraries: libtinfo.so.6: cannot open shared
object file: No such file or directory
```

Resolve shared library (`libtinfo.so`) errors by creating symbolic links.

```
sudo ln -s /usr/lib64/libtinfo.so.5 /usr/lib64/libtinfo.so.6
sudo ldconfig
```

The problem has been solved, but there will be an error message next:

```
[root@192 project]# sudo ln -s /usr/lib64/libtinfo.so.5 /usr/lib64/libtinfo.so.6
[root@192 project]# sudo ldconfig
[root@192 project]# clang --version
clang: /lib64/libtinfo.so.6: no version information available (required by clang)
clang: /lib64/libm.so.6: version GLIBC_2.27' not found (required by clang)
clang: /lib64/libm.so.6: version GLIBC_2.29' not found (required by clang)
clang: /lib64/libc.so.6: version GLIBC_2.34' not found (required by clang)
clang: /lib64/libc.so.6: version GLIBC_2.32' not found (required by clang)
clang: /lib64/libc.so.6: version GLIBC_2.33' not found (required by clang)
clang: /lib64/libstdc++.so.6: version GLIBCXX_3.4.30' not found (required by clang)
clang: /lib64/libstdc++.so.6: version GLIBCXX_3.4.29' not found (required by clang)
clang: /lib64/libstdc++.so.6: version CXXABI_1.3.13' not found (required by clang)
clang: /lib64/libstdc++.so.6: version CXXABI_1.3.11' not found (required by clang)
clang: /lib64/libstdc++.so.6: version GLIBCXX_3.4.26' not found (required by clang)
clang: /lib64/libstdc++.so.6: version GLIBCXX_3.4.20' not found (required by clang)
clang: /lib64/libstdc++.so.6: version GLIBCXX_3.4.21' not found (required by clang)
```

The error message indicates that the current versions of GLIBC and libstd c++ in the system are outdated and do not meet Clang's running requirements. GLIBC is the GNU C library and a core component for many programs to run. It needs to be upgraded to at least 2.34 version.

Here, the upgrade of GLIBC is carried out through source code compilation, but due to the outdated make and bison versions of the system, an upgrade is required. After the upgrade is complete, set the 'LD_LIBRRY-PATH' environment variable to use the new GLIBC library. But there will be an error here:

```
relocation error: /opt/glibc-2.34/lib/libc.so.6: symbol __tunable_get_val, version
GLIBC_PRIVATE not defined in file ld-linux-x86-64.so.2 with link time reference
```

This error indicates that the 'ls' command used a new' glibc 'library (`/opt/glibc-2.34/lib/libc. so. 6`) at runtime, but the library is not compatible with the current dynamic linker' ld-linux-x86-64. so. 2 '.

There is currently no better way to solve this problem.

```
clang: /lib64/libstdc++.so.6: version GLIBCXX_3.4.30' not found (required by clang)
#The above issue can be resolved by upgrading the GCC version to GCC 13.
```

After consulting the information, it was found that this machine can successfully install gcc 11 and clang 7, and upgrade the boost to version 1.78.0.

3.Start compiling:

Error 1:

```
During the compilation process, there were errors such as'__u32'and'__u64'with
undefined data types. These were resolved through the solutions provided in the
issue (Try replacing' __u32 'to'uint32_t','__u64'to'uint64_t', etc.). But there will
also be reports later that other types are undefined
```

Error 2:

```
link.c:579:10: error: ‘NFPROTO_NETDEV’ undeclared here (not in a function); did you
mean ‘NFPROTO_DECNET’?
  579 |         [NFPROTO_NETDEV] = "netdev",
      |          ^~~~~~~~~~~~~~
      |          NFPROTO_DECNET
link.c:579:10: error: array index in initializer not of integer type
link.c:579:10: note: (near initialization for ‘pf2name’)
make[4]: *** [link.o] Error 1
```

Modify the code and compile it.

Error 3:

```
Unable to find<boost/container_fash/ash. hpp>in the boost library
```

Upgrading the boost library can solve this problem.

Error 4:

```
In file included from /root/project/bpftime/daemon/kernel/bpf_tracer.bpf.c:6:
/root/project/bpftime/third_party/vmlinux/x86/vmlinux.h:5:15: error: attribute
'preserve_access_index' is not supported by '#pragma clang attribute'
#pragma clang attribute push (__attribute__((preserve_access_index)), apply_to =
record)
```

This error is caused by the clang version being too low.

Therefore, a higher version of Clang is still needed. Upgrade Clang to Clang 12 version here.

```
[root@192 local]# clang -v
clang: /lib64/libtinfo.so.5: no version information available (required by clang)
clang: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.20' not found (required by clang)
clang: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.21' not found (required by clang)
```

In response to this issue, libstdc++ has been converted into a library for GCC 13, which solves the problem.

The current environment is GCC 13 Clang 12， Continuing to compile: There were still many errors reported during the compilation process, especially when compiling libbpf, where there were many errors.

Turning off the compilation of libbpf will result in an error:

```
/root/project/bpftime/runtime/src/bpf_map/userspace/prog_array.cpp:8:10: fatal error: bpf/bpf.h: No such file or directory
    8 | #include "bpf/bpf.h"
      |          ^~~~~~~~~~~
compilation terminated.
gmake[2]: *** [runtime/CMakeFiles/runtime.dir/src/bpf_map/userspace/prog_array.cpp.o] Error 1
```

There is a lack of libbpf header files in CentOS 7. Introducing libbpf header files into the system will result in errors similar to the ones mentioned above (such as undefined types), and these issues have not been resolved yet.

二、Try compiling it in Docker and putting it into CentOS 7

Putting all the compiled library files and dependency libraries in Docker into CentOS 7 still results in an error:

```
[yys@192 lib]$ bpftime
bpftime: /lib64/libm.so.6: version `GLIBC_2.38' not found (required by bpftime)
bpftime: /lib64/libm.so.6: version `GLIBC_2.27' not found (required by bpftime)
bpftime: /lib64/libm.so.6: version `GLIBC_2.29' not found (required by bpftime)
bpftime: /lib64/libm.so.6: version `GLIBC_2.35' not found (required by bpftime)
bpftime: /lib64/libc.so.6: version `GLIBC_2.22' not found (required by bpftime)
bpftime: /lib64/libc.so.6: version `GLIBC_2.25' not found (required by bpftime)
bpftime: /lib64/libc.so.6: version `GLIBC_2.38' not found (required by bpftime)
bpftime: /lib64/libc.so.6: version `GLIBC_2.32' not found (required by bpftime)
bpftime: /lib64/libc.so.6: version `GLIBC_2.28' not found (required by bpftime)
bpftime: /lib64/libc.so.6: version `GLIBC_2.33' not found (required by bpftime)
bpftime: /lib64/libc.so.6: version `GLIBC_2.34' not found (required by bpftime)
bpftime: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.31' not found (required by bpftime)
bpftime: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.32' not found (required by bpftime)
bpftime: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.30' not found (required by bpftime)
bpftime: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.29' not found (required by bpftime)
bpftime: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.22' not found (required by bpftime)
bpftime: /lib64/libstdc++.so.6: version `CXXABI_1.3.9' not found (required by bpftime)
bpftime: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.26' not found (required by bpftime)
bpftime: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.20' not found (required by bpftime)
bpftime: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.21' not found (required by bpftime)
```

I continued to try upgrading the GLIBC library, but it still failed (there may be issues with commands in the system).

Deploying the bpftime project through the above two methods was unsuccessful. Therefore, try deploying in a container:

```
12:05:42                                                malloc called from pid 505
12:05:43                                                continue malloc...
12:05:44                                                malloc called from pid 505
12:05:45                                                continue malloc...
12:05:46                                                malloc called from pid 505
12:05:47                                                continue malloc...
12:05:48                                                malloc called from pid 505
12:05:49                                                continue malloc...
12:05:50                                                malloc called from pid 505
        pid=505         malloc calls: 9                 continue malloc...
12:05:51                                                malloc called from pid 505
        pid=505         malloc calls: 10                continue malloc...
12:05:52                                                malloc called from pid 505
        pid=505         malloc calls: 9                 continue malloc...
12:05:53                                                malloc called from pid 505
        pid=505         malloc calls: 10                continue malloc...
```

Deploying bpftime through container methods is not a problem.

三、Summary

Through the above adaptation process, it can be found that the main problem is the difficulty in upgrading the **GLIBC** library. GLIBC is the fundamental library of the system, and almost all programs and tools rely on it. Upgrading 'glibc' will directly affect the entire system. And many installed software are compiled based on old versions of glibc, which may cause compatibility issues after upgrading glibc. If the 'glibc' update fails, the system may not be able to complete the startup process, and may even be completely unable to enter the system. I will look for better ways to solve this problem in the future.

Therefore, in older versions of the system, the more recommended way to adapt to the bpftime project is to deploy it through a **container**. This method can better install some of the dependencies required by bpftime without conflicting with the system library. Moreover, this method can be more compatible with the eBPF features in the kernel and better utilize the functionality of bpftime.