$ mkdir $HOME/workdir  
$ cd $HOME/workdir

### NOTE: glibc source code and following patch is availble only with in Intel.  
$ wget <https://github.com/intel-sandbox/iodlr/blob/pkb_wp_php_2gb_fix/containers/wordpress/pkb_wp_opt/files/0001-x86-64-Restore-LD_PREFER_MAP_32BIT_EXEC-support-BZ-2.patch>

$ cd $HOME/workdir && git clone <https://github.com/intel-innersource/applications.compilers.source.gnu-toolchain.glibc.git> glibc

$ cd $HOME/workdir/glibc  
$ git checkout f4fe72a  
$ git apply $HOME/workdir/0001-x86-64-Restore-LD\_PREFER\_MAP\_32BIT\_EXEC-support-BZ-2.patch  
$ cd $HOME/workdir  
$ git clone <https://github.com/intel-innersource/applications.compilers.source.infrastructure.tools-build.git> tools-build  
$ cd $HOME/workdir/tools-build/glibc  
$ echo "GLIBC-SOURCE-DIR=$HOME/workdir/glibc" > config.make  
$ make glibc -j  
$ cd $HOME/workdir/tools-build/glibc-runtest  
$ echo "GLIBC-SOURCE-DIR=$HOME/workdir/glibc" > config.make  
$ make glibc -j

### How to use this library with your application.  
$ cat $HOME/wrapper.sh  
#!/usr/bin/bash  
LD\_PREFER\_MAP\_32BIT\_EXEC=1 $HOME/workdir/tools-build/glibc-runtest/build-x86\_64-linux/testrun.sh <absolute path to application.exe> $@

$ chmod +x $HOME/wrapper.sh

$ $HOME/wrapper.sh

Smaps data collection step

1. Launch your application
2. Get the “process id” of the launched application
3. Execute following commands as “root” user or with “sudo”

$ sudo cat /proc/<process id>/smaps > app.smaps.txt

1. Analyze smaps output to identify application’s “.text” region address range and all the other objects loaded at various addresses to have a high level understanding of the memory mapping.