6.1.4 System logs

All files referenced in the `results.config` file, (described below in section 6.2 will be collected into the artifact archive.

6.1.5 Capturing information about the environment

Some labs require the student to alter system configuration settings, e.g., using the sysctl command to affect ASLR. A *precheck.sh* script in:

```
$LABTAINER_DIR/labs/[labname]/[container name]/_bin
```

is intended to contain whatever commands are necessary to record the state of the system at the time a program was invoked. The stdout of the precheck.sh script is recorded in a timestamped *precheck.stdout* file. The timestamp of this file will match the timestamp of the stdin and stdout artifacts associated with the command that caused precheck.sh to run. The precheck.sh is passed in the full path of the program as an argument, thereby allowing the designer to capture different environment information for different commands.

As another example, consider the file-deletion lab *precheck.sh* script. It mounts a directory, lists its content, and unmounts it. This all occurs transparently to the student, and, in this example, helps confirm a specific file was in fact deleted at the time of issuing a command to recover deleted content from the volume.

In other situations, you may wish to capture environment information when selected commands are executed, even though you have no interest in stdin or stdout of those commands. For example, imagine you want to capture the file permissions of /usr/bin/tcpdump whenever that command is executed. This can be achieved by including /usr/bin/tcpdump in a list within a file at:

```
$LABTAINER_DIR/labs/[labname]/[container name]/_bin/forcecheck
```

and then include `ls -l /usr/bin/tcpdump` in the precheck.sh script. Note that the *forcecheck* list of programs must include the full path name. The *forcecheck* file can be used instead of a *treataslocal* file entry for those cases where stdin and stdout are not required for goal assessment. An example of the use of forcecheck can be found in the *capabilities* lab.

6.1.6 Capturing file access events

File creation, reading and modification events can be recorded using a combination of a `notify` file and an optional `notify_cb.sh` script at:

```
$LABTAINER_DIR/labs/[labname]/[container name]/_bin/
```

The `notify` file will name directory or file paths and the access modes of interest, one entry per line, having this format:

```
<file_path> <mode> [output file]
```

where the `file_path` is the absolute path to the file of interest, and `mode` is one of the following:

- `CREATE` Assumes the path is to a directory. This will capture any file or directory creation within the named directory.
- `ACCESS` will capture any read of the file named by the path.
- `MODIFY` will capture any write to the file named by the path.