Project: File System Attributes

Team members: Aakash Basnet, Ranjan Khadka, Elijah Carter

For this project, we were supposed to create a mechanism that allow users to attach specific attributes to files and directories. In order to do that, we were supposed to write several system call functions, modification to some existing file system code and the creation of new user-space programs to display the attributes. There were several guidelines that were provided to us which we adhered to in order to create such mechanism.

System Calls:

We wrote four system call methods for our purpose.

Set_attribute: The first system call was set attribute method where we were supposed to set the attribute for a specific file. We were given character pointer to filename, character pointer to attribute name, character pointer to attribute value and some size. We allocated some space in memory and stored the directory name in there. And then we created a character pointer to store the name of the directory containing the filename. We then opened the directory of that path and created another directory that stored the attribute. Within that directory we created a file and wrote some value in it and finally returned 0.

Get_attribute: In this system call, we were supposed to get the attribute value nad we were only passed character pointer to filename, attributename, buffer and bufsize of data type integer. We

just allocated memory to store a character pointer to directory name. We first the opened the directory containing the file attribute. After making sure that the directory exists, we then proceed to find the value of the attribute contained within the test attribute directory. Inside the if condition, we opened the attribute file and read the value of that file.

Get_attribute_name: In this system call, we were supposed to get the name of all the attribute names contained within the attribute directory. Here we were just passed a character pointer to filename, buffer and an integer. We allocated some space in memory to store a character which we created a pointer to point to. We then also stored the directory path of our filename. After that we created another directory for storing test file attribute. After creating that checked if the directory named as such is empty. If it not we created a while loop to read through all the attribute names in the directory. While reading through we also stored the attribute name in buffer.

Remove_attribute: In this system call, we were provided a character pointer to filename and attribute name. Like before, we allocated some space in memory to store attribute name. We also created a character pointer to point to that space in memory. After that we created a path to the attribute name that was supposed to be deleted. We read through the directory through the while loop and upon finding the attribute name, we removed that attribute by removing the attribute path that we created earlier. There is no test program to verify this works.

Syscall Number for all four syscall function are as follows:

- 355 i386 fa cs401_set_attribute
- 356 i386 fa2 cs401_get_attribute

- 357 i386 fa3 cs401_get_attribute_names
- 358 i386 fa4 cs401_remove_attribute

User Space Test Programs:

We used two user space programs to test our system calls. In order to make things simple, we created two different user space programs, setattr and listattr. The setattr program takes input a single name = value pair and a list of files to apply the attribute. Another test program that we wrote was listattr. This test program is supposed to take as input an attribute name and a list of files and output the values and names of the attributes requested for those files.

The setattr program functions by stripping the “attribute name=attribute value” pair and calling set_attribute for each file in the command line list passed in. The set_attribute syscall has issues with directory permissions and does not create and write in the same call.

The listattr program with the LISTALL option returns a list of all attributes, calling the get_attribute syscall for each one.

However, get_attribute does not return the proper attribute value when reading form the file so standard listattr calls do not display properly.

- Step to run User Space test Program:

I. cd /path/to/user_space_test

II. Make listattr

III. Make setattr

IV. For seattr, use

./setattr “AttrName=AttrValue” <filename>

V. For listattr, use

./listattr AttrName <filename>

Here are a screenshots of test program results:

Problem faced:

- We failed to properly implement the necessary modifications to our kernel support modules: sys_rename and sys_unlink. It was tasking to locate the files in the kernel. The lx browser recommended in the project pdf might have been useful.
- Debugging and compiling time took most part of the development process. Classmates provided a means to more quickly compile the kernel through caching, however.
- Proper Understanding of how the user-space program and kernel modules program’s codes are written.
- Performance testing of the programs was planned, but, for sake of time, not implemented.
- Generating a patch file (disk space insufficient)