Name -Harshil Patel
Roll No - 202012055
Mentor – Lavneet Singh

Summer Internship Report

Source code: https://github.com/202012055/summer_internship/tree/master/snake-on-a-tree

Project - Snake-on-a-Tree Algorithm

Introduction:

- It is a novel algorithm for making only some files available for accessing.
- It works by manipulating the file permissions on any unix-like OS.
- To make the process efficient we only change the minimum number of file permissions.
- For Example: if no file in the sub-tree of a dir is public then we can just remove the executable permission on the dir and then no process will be able to climb down that dir.
- The end result looks like a snake(a series of revoked permissions) on a tree(file-heirerchy) so i named it snake-on-a-tree.

Implementation:

- The algorithm is implemented as a bash script library.
- It is needed to be sourced by the user script.
- It exports 1 global variable:
    1. SCOPE

       Unix-permissions have three parts:

       (a) user (u)

       (b) group (g)

(c) other (o)

It represents the part of permissions on with the functions act.

Its value can be [u][g][o].

Default value: ‘o’.

- It exports 3 functions:
  1. setROOT

     Sets the root of the tree on which other functions act.

     Takes 1 arg, a path to dir.
  2. makePublic

     Makes that dir/file public.

     Takes 1 arg, a path relative to ROOT.

     Returns: 0-made public,1-already public.
  3. makePrivate

     Makes that dir/file private.

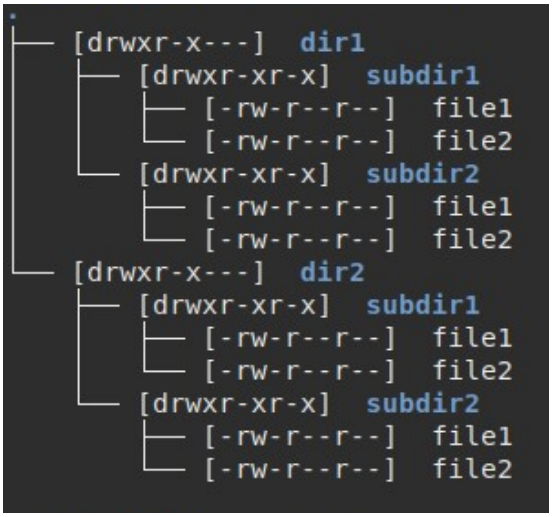
     Takes 1 arg, a path relative to ROOT.

     Returns: 0-made private,1-already private,2-under a public ancestor.

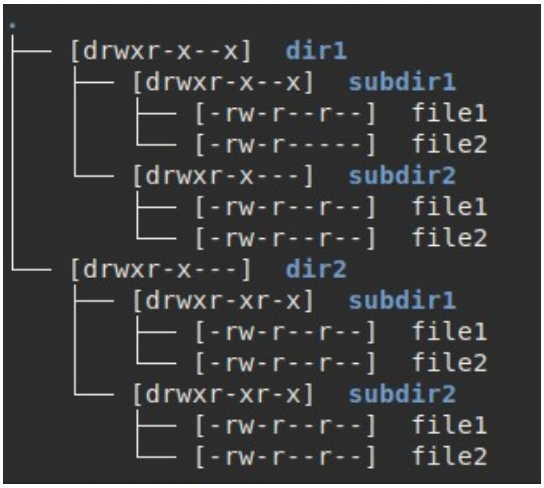
Testing:

- I made another project (usync) that uses this algorithm as its sub-module.
- usync is my personal project that can be used to effieciently sync the files across multiple devices.
- For usync, snake-on-a-tree provides a way to restrict certain files from being shared with untrustworthy devices.
- Bellow i will use usync to test the sanke-on-the-tree algorithm.
- To see the results of our actions i m also showing the file-heirerchy and file permissions produced by ‘tree -p’ command.

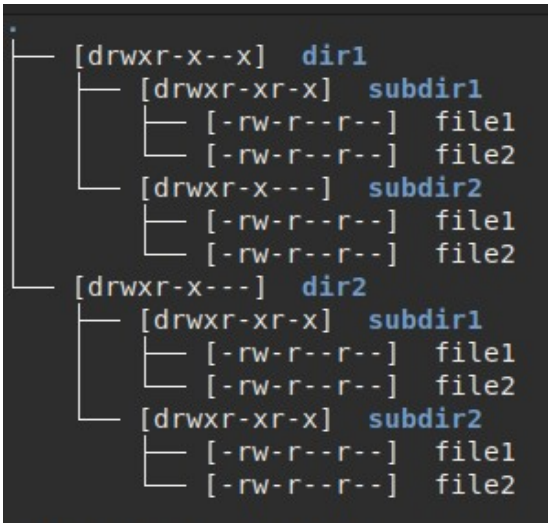
1. Initial Tree: every thing is private



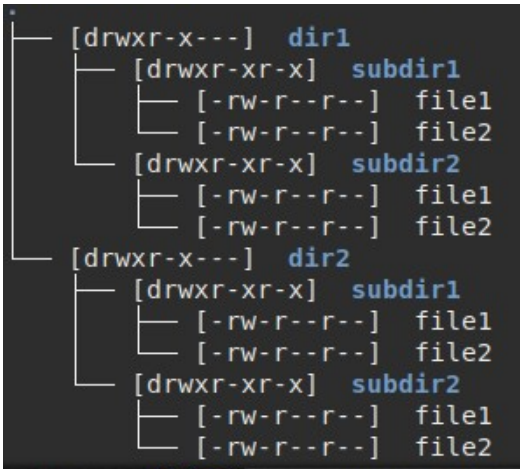
2. after making dir1/subdir1/file1 public



3. after making dir1/subdir1 public



4. after making dir1 private



Limitations:

- This algorithm only works on static file-heirerchy.
- If the tree's structure is changed then the resulting structure might not be secure and most likely will not be understood by the later runs of the algorithm.
- To circumvent this problem usync provides an option that reapplies all the permissions and make the tree consistent again, but it is a very expensive opperation.