Untouchables

Learning Objectives

- Design and implement a C program using structured programming techniques
- Design and implement a C program applying secure cyber design
- Design and implement a C program that modifies a file on disk but leaves the accessed and modified times unchanged
- Design and implement a C program that creates a file on disk that has the same modify and access times as its parent directory

Before Class

In preparation for in class activities, complete the following activities:

- None.

In Class

Assignment Information

in operations: File Stamps File stamps are commonly used as simple indicators of whether a file has been modified or not. The question you should be asking yourself is whether time stamps are sufficient or not to indicate whether data has been modified? Time stamps are insufficient for determining the integrity of data, since if the data can be modified (write access) then the last accessed and last modified time stamps can be modified (i-node write access).
Regardless of the inability for time stamps to provide integrity, time stamps are naively used for integrity checks. Verifying time stamps are useful in detecting malicious activities; low capability threats will likely not change time stamps, humans make mistakes, it may only take one piece of evidence to indicate a compromise and cause operators to take a deeper look.
Similarly, we need to understand the forensic evidence our actions leave, and what steps can be taken to remove that forensic evidence.
As you learn more details of a system, you learn the breadth and depth of steps needed to infiltrate a system, maintain persistence on a system, and remain undetected; or the breadth and depth of steps needed to operate and defend a system.

note: Useful Time Stamps The utility `make` uses time stamps to determine if a target file is up to date or note. The course website is managed using `make`, the course calendar entries are in a source file that is processed and used to generate the source HTML.

Lab 0x03: Untouchables

Assignment Type:	Programming - Laboratory	Collaboration Policy:	Default
Assignment		Lab 0x03: Untouchables	

Due	
Section	Course Server Time / Date
1121	Pseudo Code: Friday, 1800 03 Mar 2018 Complete: Tuesday, 0755 07 Mar 2018
3122	Pseudo Code: Friday, 1800 03 Mar 2018 Complete: Tuesday, 0955 07 Mar 2018
5123	Pseudo Code: Friday, 1800 03 Mar 2018 Complete: Tuesday, 1330 07 Mar 2018

General Comments:

- Make use of in-class time to the fullest extent possible
- Read the entire assignment before you begin
- You are given more time to complete programming assignments because they are expected to take you longer to complete
  - It is expected that you will need to work on programming assignments outside of scheduled class
  - Do not procrastinate on starting a programming assignment
- General Instructor Feedback from Previous Labs:
  - General & Password Complexity

Given Material:

- Starter Code: `untouch.c`
- Test Script: `run-test.sh`
- Compiled Solution: `untouch` (x86-64)
  SHA-256: `911d63de8f9d61c353c984676412296ef5fdd12c4af8ba00bae7dbbb15c0d53f`

Test

The following are examples of how your program may be tested from the command line.

```
$ ./untouch  # No command line argument (All Parts)
NAME
        untouch - Modify regular file but do not change access time or
                  modify time
SYNOPSIS
        untouch FILE
DESCRIPTION
        Modify FILE (regular file) but do not change the last accessed and
        last modified times. If FILE does not exist, create FILE assigning
        last accessed and last modified times of the parent directory.

        If the command line argument is not a regular file, no actions are
        taken, and the program exits normally.

$ echo $?  # Normal exit
0

$ ls
myDir/    myFile.txt   myLink@   myPipe|

$ ./untouch myDir  # Invalid argument (All Parts)
untouch-ERROR 1-myDir: Invalid argument

$ echo $?  # Error exit
1
```

```
$ ./untouch badFile  # Argument does not exist (Part 1, Part 2)
untouch-ERROR 2-badFile: No such file or directory

$ echo $?  # Error exit
2

$ cat myFile.txt
howdy

$ ls -la
drwxr-x--x 3 hoffmeis scs 4096 Feb 16 11:10 ./
...
-rw------- 1 hoffmeis scs    5 Feb 17 11:10 myFile.txt

$ ls -lua
drwxr-x--x 3 hoffmeis scs 4096 Feb 16 11:10 ./
...
-rw------- 1 hoffmeis scs    5 Feb 17 11:10 myFile.txt

$ date
Mon Feb 17 13:20:00 EST 2014

$ ./untouch myFile.txt  # Argument is regular file (All Parts)

$ echo $?
0

$ ls -l
...
-rw------- 1 hoffmeis scs   10 Feb 17 11:10 myFile.txt

$ ls -lu
...
-rw------- 1 hoffmeis scs   10 Feb 17 11:10 myFile.txt

$ cat myFile.txt
howdy
-h0ff  # -h0ff is the text appended at the end of myFile.txt; i.e the modification to myFi

$ ./untouch myFile2.txt  # Argument does not exist (Part 3)

$ echo $?
0

$ ls -la
drwxr-x--x 3 hoffmeis scs 4096 Feb 16 11:10 ./
...
-rw------- 1 hoffmeis scs    5 Feb 16 11:10 myFile2.txt

$ ls -lua
drwxr-x--x 3 hoffmeis scs 4096 Feb 16 11:10 ./
...
-rw------- 1 hoffmeis scs    5 Feb 16 11:10 myFile2.txt

$ cat myFile2.txt
-h0ff

$ ls -la myDir/
drwx------ 2 hoffmeis scs 4096 Feb 16 13:30 ./
drwxr-x--x 3 hoffmeis scs 4096 Feb 16 11:10 ../

$ ls -lua
drwx------ 2 hoffmeis scs 4096 Feb 16 13:30 ./
drwxr-x--x 3 hoffmeis scs 4096 Feb 16 11:10 ../

$ ./untouch myDir/myFile3.txt  # Argument does not exist (Part 4)

$ echo $?
0

$ ls -la myDir/
drwx------ 3 hoffmeis scs 4096 Feb 16 13:31 ./
...
```

```
-rw------- 1 hoffmeis scs    5 Feb 16 13:30 myFile3.txt

$ ls -lua myDir/
drwx------ 3 hoffmeis scs 4096 Feb 16 13:31 ./
...
-rw------- 1 hoffmeis scs    5 Feb 16 13:30 myFile3.txt

$ cat myDir/myFile3.txt
-h0ff
```

After Class

Activities

Furthering the in class activities, complete the following activities:

- Turn in the worksheet to your instructor after you have submitted your final source code.
- Submit your pseudo code in a single file named `untouchPseudo.c`, following the submission directions on the course information web page.
- Submit your final source code in a single file named `untouch-#.c`, where `#` is the part of the implementation that you completed (e.g. `untouch-4.c`), following the submission directions on the course information web page.

`submit` Assignment (project): *lab03*

Resources

Assignment Solutions:

- Worksheet Solution: Untouchables
- Source Code Solution: untouch.c

++

Precision matters. Once graded, review the test results output and think about the default (common) precision regarding times displayed by `ls(1)`. The default long listing format of `ls(1)` only goes to the minute, *varStat*`.st_atime` is in seconds, but the system keeps time down to the nanosecond. In order to see time information down to the nanosecond via the command line you need to use `stat(1)`.

> **note:** See also: `stat(2)` man page Notes section.