CSE-509 Spring 2020

Assignment 2 Discussion

ptrace, ltrace and strace

- Read the man pages carefully
- ***Read the man pages carefully!***
  - Don’t try to remember every item on the man page, but know enough that you know where to look for specifics.
  - Few ptrace flags are more useful for your assignment. Identify those. “ptrace” supports operations to PEEK, POKE data into the registers, and also have flags to mark, or conditionally run the monitored process (flags like TRACEME or CONT)
- Try ltrace and strace with different input flags to get a better understanding
  - Think about how these tools can be implemented
  - You can look at their source code, but you need to figure out how to get to the most relevant parts of the code
- Read the Linux journal article on ptrace.

Library vs System-call Interception

- Review lecture notes and recordings on this topic
  - See Topic 5 under “Lectures” on the course web page
- Study a bit about dynamic loaders and see what are the different options of loading library files in your system, like LD_PRELOAD or LD_LIBRARY_PATH.
- Review the two links provided on this topic (at netspi.com and samanbarghi.com)
- Without good knowledge of the tools and techniques discussed so far, and a good deal of hands-on practice in the Warmup, you won’t even fully understand Part 2 of this assignment.
- You should also know how the registers are used during system calls, and how you can identify when a specific system call has been invoked or exited.

Option 1: (Access remote URLs as if they were local files)

- Identify the system calls that are used to open, read and write local files.
  - Try this out with a few command line applications (e.g., cp or rm) and one or two GUI applications (e.g., gedit)
- Find a convenient application (e.g., wget) that can download a web page and store it in a local file.
  - Sometimes it is not just a single file, decide how you will handle this
- Intercept and modify the system calls identified in the first step so that:
  - you recognize a system call associated with opening an URL
  - fork/exec wget to fetch and save the web page locally
  - redirect the system call to this local copy.

Option 2: (Logger extension)

- Identify the system calls and calls to dynamic libraries needed to open a file or network connections using “strace” and “ltrace” commands.
  - Try it out with applications like “gedit” or “Firefox”, or commands like “ping, wget, vi, cat” etc.
  - For simplicity, only concentrate on calls made by an application itself, not by its child processes (if any).
- Use “ptrace” with appropriate flags to get the contents of different registers during the system call.
  - You should know which all registers are used to make system calls, and read the contents of those registers using ptrace.
  - Once you trace those, open a logger file in append mode, and write the information into it. Please carefully read the assignment description, and see what all information you are required to record in order to receive credits.
- Resume the monitored application. You need to figure out the needed “ptrace” flags for identifying a monitored process to be traced, and halting and continuing execution based on conditions.

Option 3: (Automatic backup)

- Similar to the first two cases, make a comprehensive list of system calls that can possibly open and modify/ change a file, with strace and ltrace command.
- Use ptrace to halt and modify such system calls which intends to overwrite a file.
  - You need to find out the conditions to identify such system calls.
  - Check the registers contents to fiigure out the input parameters (e.g. the file path, file name).
  - Create a backup file in a .backup directory. If the directory is already present (already containing other backup files), you should not re-create the directory.
- Also, since you are needed to store unlimited backups, think of a version control system, so you do not overwrite older backups with newer ones.
  - For instance, you can rename old backup files with numbered extensions, like .BAK.1, .BAK.2 etc.