CSE 5472: Data Provenance Lab

Objective

Learn about the merits and challenges in using data provenance to enforce network security and conduct forensic investigations.

Deliverable

1. Completed `graph.py` **with source comments marking the code for each task.**
2. `task1.dot`
3. `task2.dot`
4. `task3.dot`
5. `task4.dot`
6. `question-sheet.txt`

Environment

Please use a Linux environment (e.g., Debian, Ubuntu). A virtual machine is fine.

Provided Materials

- `audit.log.gz`: A gzip compressed Linux audit log.
- `graph.py`: A template python script for completing the tasks.
- `question-sheet.txt`: A text file for writing your solutions to the tasks.

Recommended Tools

This lab will require writing Python code that uses the NetworkX library.

Tasks

1. Using `graph.py` as a starting point, parse the provided audit log and create a directed graph where each node is a process and edges represent parent/child relationships. The `label` attribute of each node should contain both the process' PID and its main executable (e.g., `/bin/sh`). Save and submit this graph as a DOT file named `task1.dot`.
2. Extend `graph.py` to also graph files and directories accessed by the processes. I.e., every process node should have edges pointing to file/directory nodes that the process accessed, if any. The new nodes should be labeled with the **full** file/directory path. Save and submit this graph as `task2.dot`.
3. Bob has a sensitive database file in his home directory named `database.db`. Modify `graph.py` to produce a *subgraph* that shows the *reverse* provenance for this file. Save and submit this subgraph as `task3.dot`.
4. Bob wants to know what other files and directories were accessed by the bash shell executing with PID 1654. Modify `graph.py` to produce a *subgraph* that shows the *forward* provenance for this process. Save and submit this subgraph as `task4.dot`.
5. Complete and submit `question-sheet.txt` using the graphs you created to help answer the questions.

Note: We are not picky about graph edge orientation so long as it is consistent. E.g., for Task 1, the directed edges can be either `parent -> child` or `child -> parent`. Both will be accepted for full credit.

Grading

- Task 1 (10 points)
- Task 2 (10 points)
- Task 3 (10 points)

- Task 4 (10 points)
- Task 5 (10 points, see `question-sheet.txt` for grading breakdown)
- **Total Points:** 50

Hints

- Task 1 can be completed using only the `SYSCALL` messages.
- In Task 2, to correctly identify the *full* paths for files and directories, you will need to keep track of both `CWD` and `PATH` messages and combine their contents appropriately.
- The provenance of an object or subject can be determined using a breadth-first or depth-first search. NetworkX provides implementations for both.
- Two ways to visualize a DOT file are to convert it into an image using the `dot` utility on Linux or by installing `xdot`. The latter has the advantage of being interactive and searchable.