System call use case

Imagine you're a software architect encountering performance issues in your software. Determined to address these issues, you decide to gain insights from the system to inform your decision-making process. Upon investigation, you discover that various commands in the user space serve similar purposes but may exhibit differences in performance. Some examples include:

- `ls` **vs.** `find`
- `cp` **vs.** `rsync`
- `diff` **vs.** `cmp`
- `sort` **vs.** `uniq`
- `grep` **vs.** `sed`

Your Main Tasks:

Time Measurement:
- Measure the execution time for each command to assess their performance.

System Interaction Identification:
- Identify which system stack (e.g., file system, network) each command interacts with during execution.

Syscall Time Breakdown:
- Break down the total execution time of each command into time per system call to understand their resource consumption.

Performance Evaluation:
- Analyze the results to determine which command performs better than the other based on execution time, system interaction, and syscall breakdown.

Hints:

- Utilize the `tldr` command for each command to find practical use cases and examples (`tldr ls`, `tldr find`, etc.).
- `To Measure total time please use` **`time`** `command.`