. Linux Hardening Audit Tool

**What It Does (The Big Picture)**

The audit.py script is a **command-line tool** that acts as a basic security scanner for your Linux system.

Its main purpose is to automatically check for common security misconfigurations and vulnerabilities. Instead of you (the administrator) having to manually check 10-15 different things, this script runs all those checks in a few seconds.

At the end, it gives you two things:

1. A **human-readable report** in your terminal that shows what passed, what failed, and what's just informational.
2. A **compliance score** (e.g., 80%) to quickly tell you how secure the system is based on its checks.

**How It Works (The Technical Details)**

The script is broken down into several parts:

1. **run\_command(command) function:**
   * This is a "helper" function. Its only job is to run any terminal command (like systemctl is-active ufw) that you give it.
   * It captures the text output from that command and returns it.
   * If the command fails or produces an error, it prints a warning and returns None (nothing), so the script doesn't crash.
2. **check\_...() functions (The Core Logic):** Each of these functions performs one specific security check.
   * **check\_firewall():** Runs systemctl is-active ufw. If the output is "active", it's a **PASS**. If it's "inactive", it's a **FAIL**.
   * **check\_ssh\_config():** Uses grep to read the /etc/ssh/sshd\_config file. It looks for the line PermitRootLogin. If it finds PermitRootLogin no, it's a **PASS**. If it finds anything else (like PermitRootLogin yes or if the line is missing), it's a **FAIL**.
   * **check\_file\_permissions():** This one is a bit different. It uses Python's built-in os.stat() function (instead of a shell command) to get the permissions of /etc/passwd and /etc/shadow. It compares the permission numbers (e.g., 644) to the secure-recommended numbers.
   * **check\_cis\_umask():** Similar to the SSH check, it greps config files like /etc/profile to see if the default umask (a file permission setting) is set to a secure value like 027.
   * **check\_unused\_services():** Runs a systemctl command to list all services that are set to start on boot. It marks this as **INFO** because it can't know which services you need; it just provides the list for you to review manually.
   * **check\_rootkits():** First, it checks if the rkhunter tool is installed. If not, it reports an **ERROR**. If it is, it runs the rkhunter --check --rwo command (which can take a minute) and looks for the word "Warning:" in its output.
3. **report, score, and max\_score (The Data):**
   * report = []: This is just an empty list. Every time a check\_...() function runs, it adds a dictionary of its findings (like {"check": "Firewall", "status": "FAIL", ...}) to this list.
   * score & max\_score: These are counters. For every check that *can* be graded (like the firewall), it adds 1 to max\_score. If that check passes, it also adds 1 to score. This is how it calculates the percentage at the end.
4. **main() function (The "Conductor"):**
   * This is what ties everything together. When you run sudo python3 audit.py, the main() function is what runs first.
   * It first checks if you are root (using sudo). If not, it stops.
   * It then calls each check\_...() function one by one.
   * After all checks are finished and the report list is full of results, it prints the "Audit Report Summary" by looping through the report list.
   * Finally, it does the math (score / max\_score) \* 100 to print your final compliance percentage.





