



Institute for High Energy Physics named by A.A. Logunov of National
Research Centre "Kurchatov Institute"

НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ ЦЕНТР
«**КУРЧАТОВСКИЙ
ИНСТИТУТ**»

Development of a knowledge base system for administration of the NRC 'Kurchatov Institute' – IHEP computing center based on Linux history tools

Author:

Maria Shemeiko

Co-authors:

Viktor Kotliar, Valerii Morozov

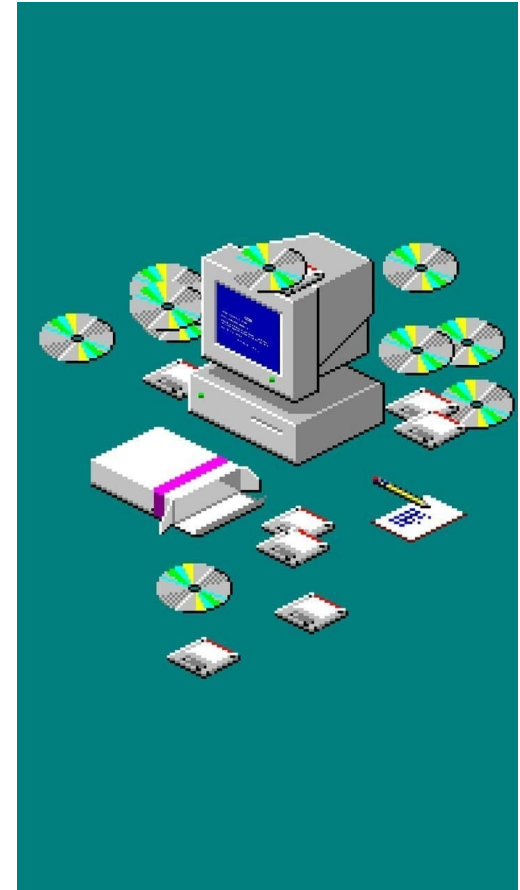
Introduction

Problem: System administrators need to maintain knowledge of numerous server commands across different systems.

Objective: To establish a unified knowledge base for collecting and storing command execution history.

Challenges: Current solutions prove unsuitable for various reasons. Examples:

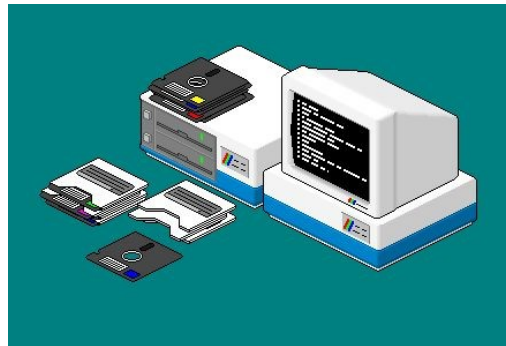
- Command Cache
- JC (JSON Convert)



Linux history tools

Three most common tools for logging activity:

- Sudo Logging – Audits privileged commands (sudo).
- Process Accounting – Logs detailed information about every command executed on the system (auditd, acct).
- Bash History – Tracks user command history (.bash_history).



Configuration for .bashrc

For the history collection script to work properly, its output must meet the following requirements:

- Preserve lines beginning with a space
- Include timestamps
- Retain history across multiple sessions



To ensure compliance and automate .bashrc modifications on worker nodes, an Ansible playbook was developed.

Ansible playbook

- Backup .bashrc file
- Remove the rule that ignores lines with spaces
`HISTCONTROL=ignorespace`
- Add configurations to .bashrc
 - Add timestamps `HISTTIMEFORMAT="%d/%m/%y %T "`
 - Increase history size `HISTSIZE=10000000`
 - Add history saving across multiple sessions

```
shopt -s histappend
```

```
PROMPT_COMMAND="history -a;$PROMPT_COMMAND"
```



Work process

Goal: Automate .bash_history upload to OpenSearch.

Script:

Pre-checks:

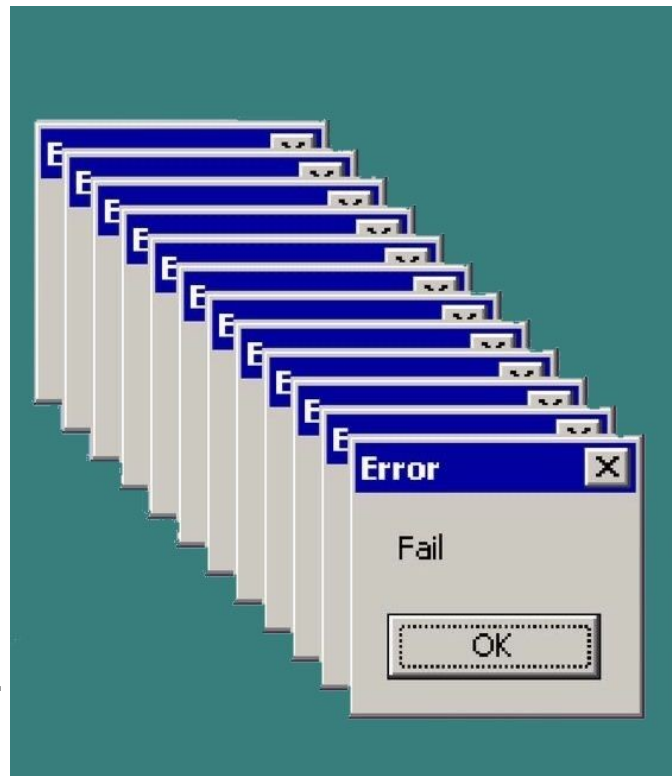
- Lock file (prevents duplicate runs).
- OpenSearch server availability (test request).

Two-Pass Processing:

- Pass 1: Extract commands + timestamps.
- Pass 2: Assign timestamps to untimed commands.

Output:

- Commands with ISO 8601 timestamps (timezone-aware).
- Progress logs.



Script output

=== [Tue 24 Jun 2025 03:35:03 PM MSK] Starting script execution ===

✓ Successfully connected to OpenSearch

Last posted timestamp: 2025-06-23 16:02:52

Posted: vi history/run_hist.sh (Time: 2025-06-24T15:29:57+03:00)



Posted: curl -s -XGET "http://localhost:9200/history/_search?pretty" (Time: 2025-06-24T15:34:26+03:00)

Updated last timestamp to: 2025-06-24 15:34:26

Posted 2 new commands.

=== [Tue 24 Jun 2025 03:35:05 PM MSK] Script completed successfully ===

Result

Columns		Sort fields	
Time (timestamp)	Source		
 Jun 26, 2025 @ 11:04:33 +03:00	hostname mon0003.m45.ihep.su command crontab -e timestamp Jun 26, 2025 @ 11:04:33 +03:00 _id ZdREq5cBymgYLNW3_EK _type - _index historian _score -		
 Jun 26, 2025 @ 11:03:09 +03:00	hostname mon0003.m45.ihep.su command vi history/historian.py timestamp Jun 26, 2025 @ 11:03:09 +03:00 _id ZNREq5cBymgYLNW3vGI _type - _index historian _score -		
 Jun 26, 2025 @ 11:03:03 +03:00	hostname mon0004.m45.ihep.su command vi history/historian.py timestamp Jun 26, 2025 @ 11:03:03 +03:00 _id aNRGq5cBymgYLNWdvGV _type - _index historian _score -		
 Jun 26, 2025 @ 11:02:44 +03:00	hostname mon0003.m45.ihep.su command vi history/run_hist.sh timestamp Jun 26, 2025 @ 11:02:44 +03:00 _id Y9REq5cBymgYLNW3vFp _type - _index historian _score -		
 Jun 26, 2025 @ 11:02:27 +03:00	hostname mon0004.m45.ihep.su command vi history/historian.py timestamp Jun 26, 2025 @ 11:02:27 +03:00 _id Z9RGq5cBymgYLNWdvFE _type - _index historian _score -		
 Jun 26, 2025 @ 11:02:14 +03:00	hostname mon0003.m45.ihep.su command crontab -e timestamp Jun 26, 2025 @ 11:02:14 +03:00 _id YtREq5cBymgYLNW3PH4 _type - _index historian _score -		

Conclusion & Future Work

Implemented:

- Ansible playbook for integrating settings into the .bashrc file
- Python script for collecting command history and uploading data to the OpenSearch database

Development Plans:

- Implementation of an algorithm to filter duplicate commands
- Integration with a task manager
- Deployment of AI-based functionality
- Adding a mechanism to detect potentially dangerous commands



Thank you for attention!