How to Use the Linux watch Command with Examples

August 11, 2021

COMMANDS LINUX

Introduction

To repeatedly run a command or job in regular time intervals while working in Linux, you can use cron jobs or bash scripts. However, Linux also offers a more straightforward, built-in solution - the `watch` command.

In this tutorial, you will learn the `watch` command syntax, how it works, and the different things it can help you do.

phoenixNAP GLOBAL IT SERVICES
How to Use the Linux Watch Command with Examples
#watch

Prerequisites

- A system running a Linux distribution (learn how to install Ubuntu 20.04, how to install CentOS 7, or how to install Arch Linux)
- An account with sudo privileges
- Access to the terminal window/command line

Linux watch Command Overview

The `watch` command is a built-in Linux utility used for running user-defined commands at regular intervals. It temporarily clears all the terminal content and displays the output of the attached command, along with the current system date and time.

By default, the `watch` command updates the output every two seconds. Press **Ctrl+C** to exit out of the command output.

The `watch` command is useful when you need to monitor changes in a command output over time. This includes disk usage, system uptime, or tracking errors.

Linux Watch Command Syntax

The `watch` command uses the following syntax:

```
watch [option] [command]
```

Where:

- `[option]`: Adding an option changes the way the `watch` command behaves. Available options are listed below.
- `[command]`: A user-defined command you want to run repeatedly.

The `watch` command options include:

`-n`, `--interval`	Allows you to specify the interval between output updates.
`-d`, `--differences`	Highlights the differences between output updates.
`-g`, `--chgexit`	Exits the `watch` command when the output of the user-defined command changes.
`-t`, `--no-tit`	Removes the header showing the interval, command, and current time and

`le`	date.
`-b`, `--beep`	Plays a sound alert (beep) if the command exits with an error.
`-p`, `--precise`	Attempts to run the command after the exact number of seconds defined by the `--interval` option.
`-e`, `--errexit`	Stops output updates on error and exits the command after a key press.
`-c`, `--color`	Interprets ANSI color and style sequences.
`-x`, `--exec`	Passes the user-defined command to `exec`, reducing the need for extra quoting.
`-w`, `--no-linewrap`	Turns off line wrapping and truncates long lines instead.
`-h`, `--help`	Displays help text and exits.
`-v`, `--version`	Displays version information and exits.

Linux Watch Command Examples

Here are some of the ways you can use the `watch` command options to achieve different results:

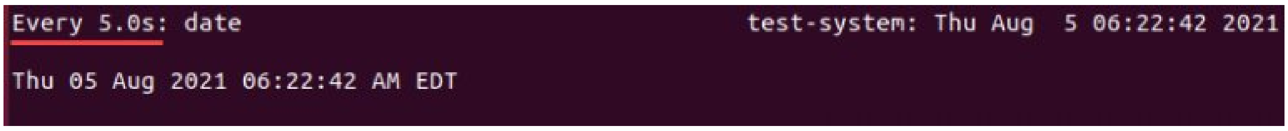
Run Command with a Custom Interval

Set a custom interval to run a user-defined command and show the output by using the `-n` or `--interval` option:

```
watch -n [interval in seconds] [command]
```

For instance, to display the system time and date every 5 seconds, run:

```
watch -n 5 date
```



```
Every 5.0s: date                              test-system: Thu Aug  5 06:22:42 2021

Thu 05 Aug 2021 06:22:42 AM EDT
```

Note: The `-n` option allows you to use fractions of a second, with a minimum interval of 0.1 seconds. When entering decimals, both a period (.) and a comma (,) work for any locale.

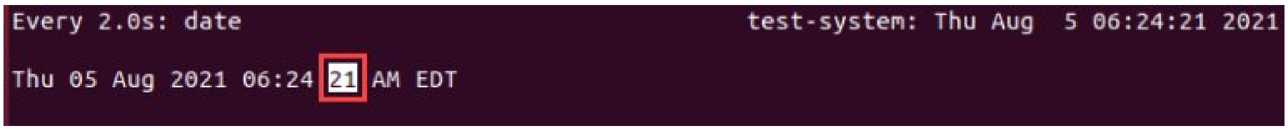
Highlighting Changes Between Updates

Use the **`-d`** or **`--difference`** option to highlight changes between successive output updates:

```
watch -d [command]
```

For example, display the system date and time in the default 2-second interval with the changes highlighted:

```
watch -d date
```



```
Every 2.0s: date                                   test-system: Thu Aug  5 06:24:21 2021

Thu 05 Aug 2021 06:24:21 AM EDT
```

Pass **`=cumulative`** to the **`-d`** option if you want all the values that have ever changed to stay highlighted:

```
watch -d=cumulative date
```

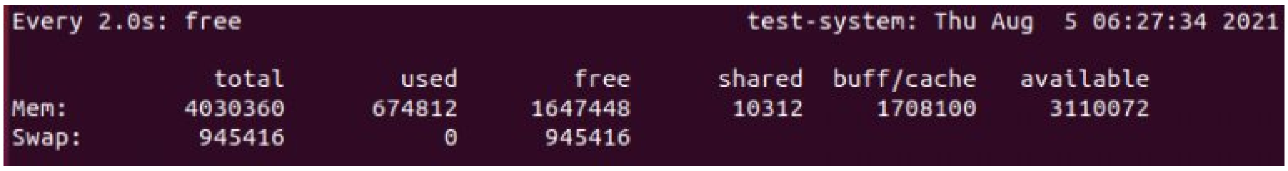
Exit on Change

The **`-g`** or **`--chgexit`** option causes the watch command to exit if there is a change in the output:

```
watch -g [command]
```

As an example, adding the free command monitors your system's memory consumption and exits if the value changes:

```
watch -g free
```



```
Every 2.0s: free                                   test-system: Thu Aug  5 06:27:34 2021

              total        used        free      shared  buff/cache   available
Mem:        4030360      674812     1647448       10312     1708100     3110072
Swap:        945416           0      945416
```

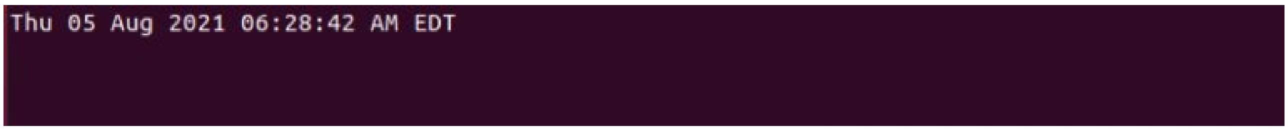
Hide the watch Command Header

Turn off the header containing the interval time, user-defined command, and current system time in the **`watch`** command output by using the **`-t`** or **`--no-title`** option:

```
watch -t [command]
```

Returning to the example of displaying the system date and time, this time without the header:

```
watch -t date
```



```
Thu 05 Aug 2021 06:28:42 AM EDT
```

Alert on Error

The `watch` command uses the beep package to play a sound alert if the output update fails due to an error. To do this, use the `-b` or `--beep` option:

```
watch -b [command]
```

Note: if you don't have the beep package installed, add it with `sudo apt install beep` command.

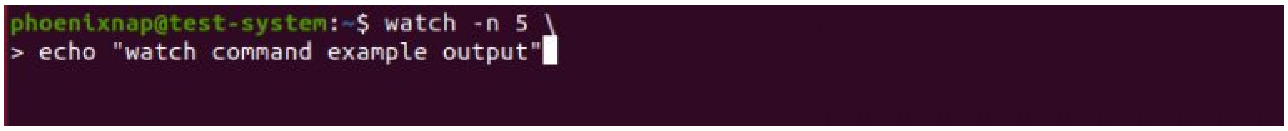
Using Complex Commands

The `watch` command also allows you to use more complex user-defined commands, with their own arguments and options. One way to do this is to use the backslash (**'\'**) symbol:

```
watch [options] \
```

Using the command above brings you to the next line in the terminal, where you need to add the user-defined command. Once you hit **Enter**, it executes the command. For instance:

```
watch -n 5 \
echo "watch command example output"
```



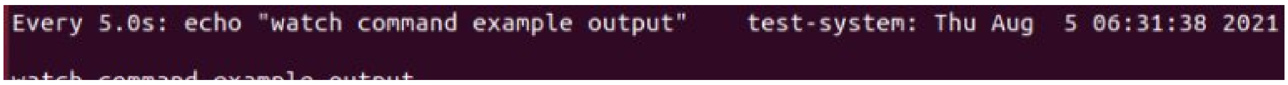
```
phoenixnap@test-system:~$ watch -n 5 \
> echo "watch command example output"
```

Another option is to add the user-define command in single quotation marks:

```
watch [options] '[command]'
```

Using the example above, the command would be:

```
watch -n 5 'echo "watch command example output"'
```



```
Every 5.0s: echo "watch command example output"    test-system: Thu Aug  5 06:31:38 2021
watch command example output
```

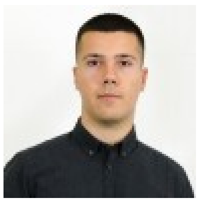
```
watch command example output
```

Conclusion

After reading this tutorial, you should have a better understanding of how the `watch` command works and what you can use it for.

For a more comprehensive overview of commands, check out our ultimate list of Linux commands.

Was this article helpful?
Yes No



Aleksandar Kovačević

With a background in both design and writing, Aleksandar Kovacevic aims to bring a fresh perspective to writing for IT, making complicated concepts easy to understand and approach.

Next you should read

SysAdmin

Linux Source Command with Examples

July 22, 2021

The Linux source command allows you to execute a list of commands from a text file or script. Learn the different ways you can use the source command in this tutorial.

READ MORE

SysAdmin

man Command in Linux with Examples

March 31, 2021

The man command is a built-in Linux utility that allows users to search for any available command and displays a user manual for the specified command.

READ MORE

SysAdmin

Echo Command in Linux (With Examples)

February 11, 2021

The echo command prints out a text string you provide as the output message. This tutorial covers the echo command syntax and different ways you can use it.

READ MORE

DevOps and Development, SysAdmin, Web Servers

How to Use the Linux sleep Command with

Examples

February 3, 2021

The sleep command is used when it is necessary to postpone the execution of commands on the command line or in shell scripts.

READ MORE

RECENT POSTS

- Nala apt Frontend for Linux
- String Slicing in Python
- How to Upgrade Debian 10 to Debian 11
- How to Use Python Struct Functions
- What is a vCPU and How to Calculate vCPU Requirements?

CATEGORIES

- SysAdmin
- Virtualization
- DevOps and Development
- Security
- Backup and Recovery
- Bare Metal Servers
- Web Servers
- Networking
- Databases

COLOCATION

- Phoenix
- Ashburn
- Amsterdam
- Atlanta
- Belgrade
- Singapore

PROMOTIONS

SERVERS

CLOUD SERVICES

- Dedicated Servers
- Database Servers
- Virtualization Servers
- High Performance Computing (HPC) Servers
- Dedicated Streaming Servers
- Dedicated Game Servers
- Dedicated Storage Servers
- SQL Server Hosting
- Dedicated Servers in Amsterdam
- Cloud Servers in Europe
- Big Memory Infrastructure

- Data Security Cloud
- VPDC
- Managed Private Cloud
- Object Storage

SERVERS

- Disaster Recovery
- Web Hosting Reseller
- SaaS Hosting

BUY NOW

INDUSTRIES

- Web Hosting Providers
- Legal
- MSPs & VARs
- Media Hosting
- Online Gaming
- SaaS Hosting Solutions
- Ecommerce Hosting Solutions

COMPLIANCE

- HIPAA Ready Hosting
- PCI Compliant Hosting

NEEDS

- Disaster Recovery Solutions
- High Availability Solutions
- Cloud Evaluation

COMPANY

- About Us
- GitHub
- Blog
- RFP Template
- Careers

CONNECT

- Events
- Press
- Contact Us

PhoenixNAP HomeBlogResourcesGlossaryGitHubRFP Template

- Live Chat
- Get a Quote
- Support | 1-855-330-1509
- Sales | 1-877-588-5918

Contact UsLegalPrivacy PolicyTerms of UseDMCAGDPRSitemap
Contact UsLegalPrivacy PolicyTerms of UseDMCAGDPRSitemap