Intro to Text Editors in Linux

Introduction:

- This workshop will introduce how to use the **vim** command-line text editor within a Linux environment.
- It will cover the basics on opening and updating a text file, how to navigate around the text, and how to search for terms.
- After the workshop, participants will have the skills to use a text editor to create basic scripts.
- Participants are expected to have an introductory level of experience using Linux from the command line – such as that provided by the Intro to Linux workshop.

Course Goals:

- To introduce users, using the Linux command line environment, to text editors: Specifically vim.
- Open, edit, update, save and quit a text editor.
- Use the keyboard and shortcuts to navigate around the text file.
- Use the command line to search for terms within the text file.

---

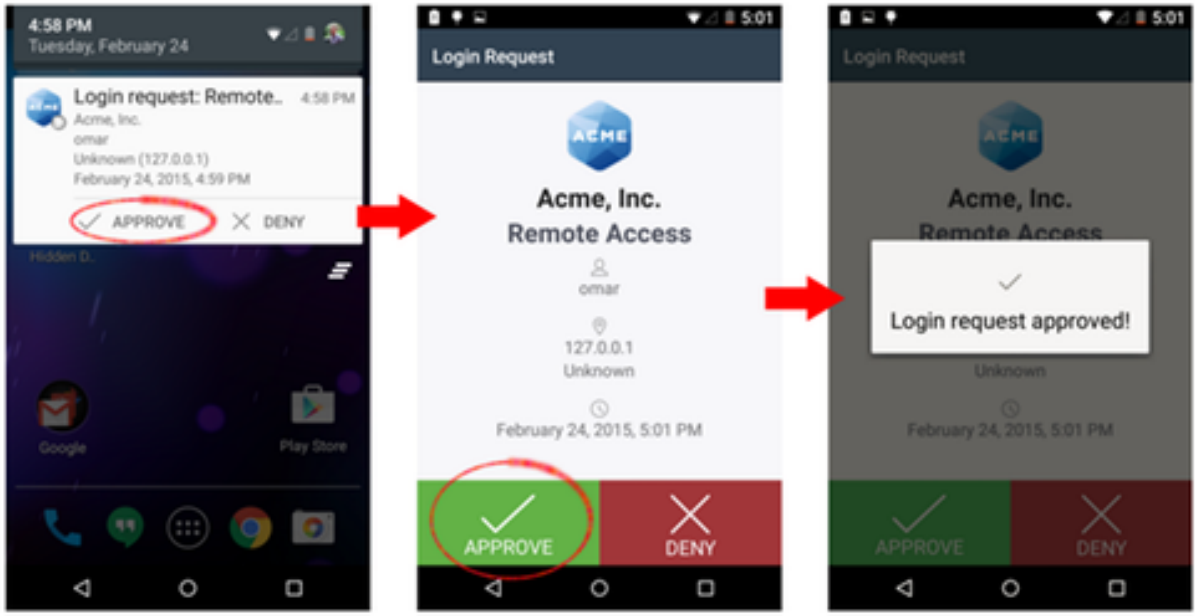
01 Getting Started

Login

1. Open up Chrome
2. Navigate to: MedicineBow OnDemand
3. Type in your provided username and password. Usually this will be your UWYO username and password, unless you are using an assigned training account.
4. Authenticate using your preferred 2 factor method (expandable directions below):

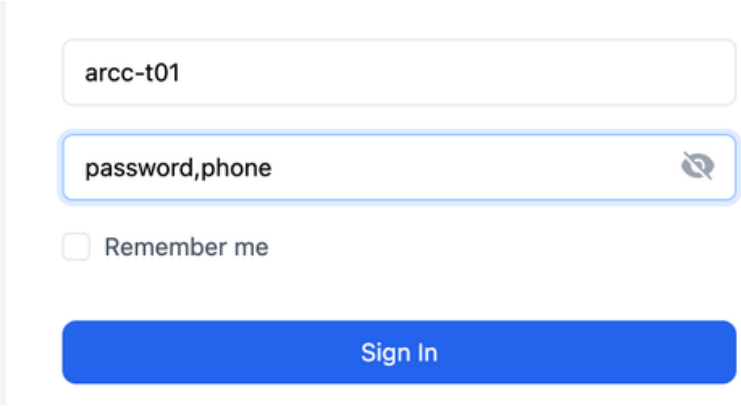
▾ Duo Mobile push:

If you usually get a two-factor push to your phone, just hit enter after entering your username and password, then complete authentication by approving the push on your device.



▾ Phone Call:

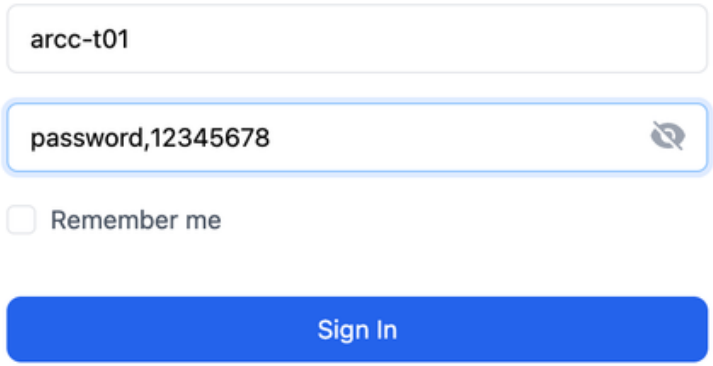
Without hitting enter after typing in your username and password, in the password text box, append a comma (,) to the end of your password, then append phone as shown in the screenshot below:



You should get a phone call on your main phone # associated with your two factor account. Answer this call and hit # to approve access.

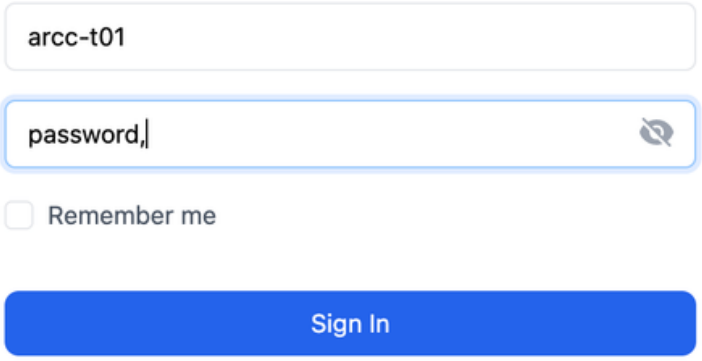
Duo Passcode

If you prefer to use a 2 factor passcode from your Duo Mobile app, without hitting enter after typing in your username and password, in the password text box, append a comma (,) to the end of your password, then append the multi digit passcode found in duo mobile as shown in the screenshot below:



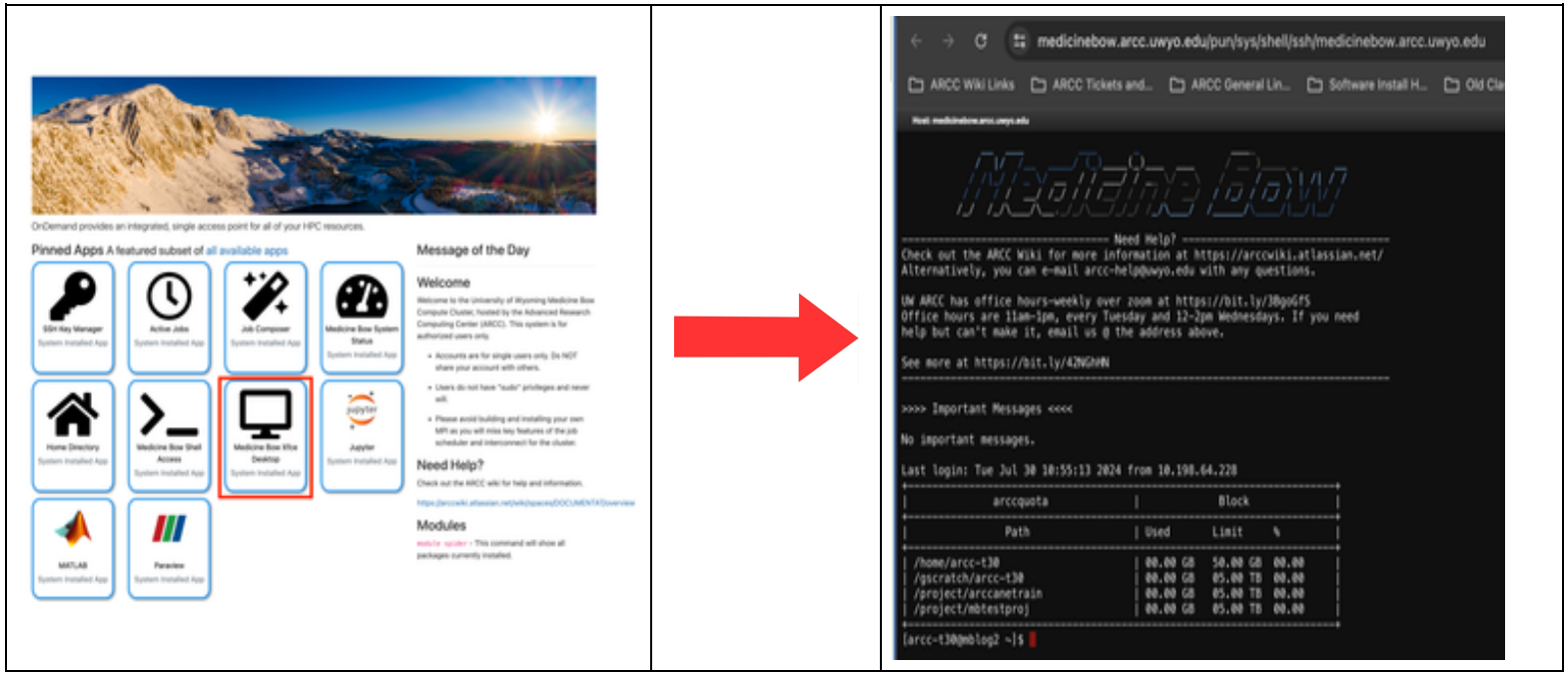
Yubikey:

Type in the account password, then, without hitting enter, append a comma (,) to the end of the password, then touch the light on the yubikey as shown in the screenshot and photo below:

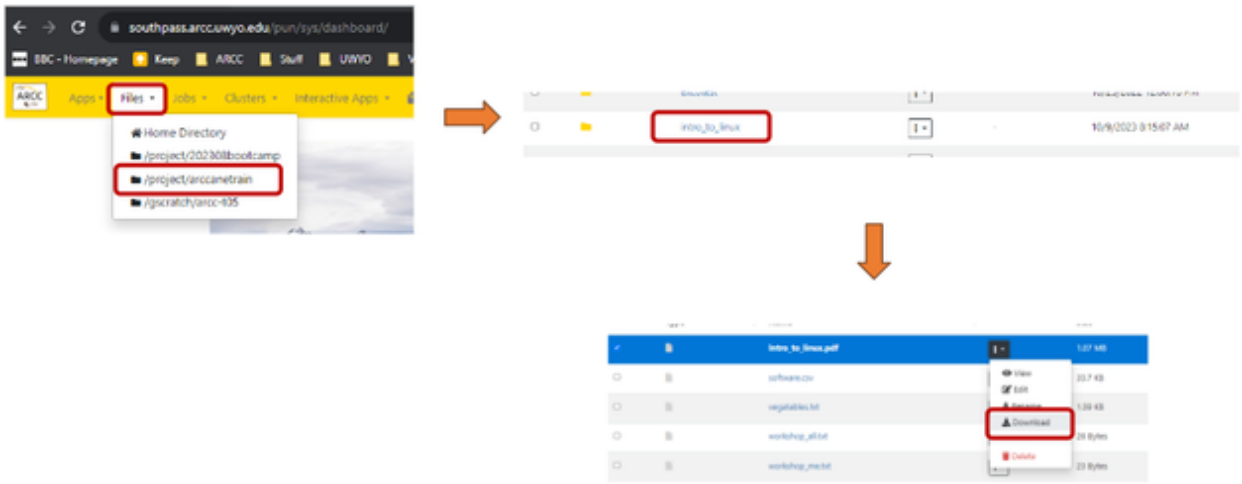


Then hit the green light on your yubikey to authenticate:

Start *MedicineBow Shell Access*



Download Slides



---

02 Why do we need a Text Editor?

02.01: Using the Terminal/Command-Line

- We have a non-GUI/non-Windows environment.
- Unable to start a GUI/Window text-editor type application.
- We're using the command-line.

02.03: Available Text Editors

- Just one of several available command-line based text editors.
  - ex: EXtended
  - emacs:

- vi:
- vim: Vi Improved: What are some of the improvements of Vim over Vi?
- nano:
- and others...

02.04: Why vim?

- **Considerations**:
  - What's available on the system you're using?
  - Can you install other editors?
  - What is your personal choice?
  - **Note**: On our Beartooth cluster, vi is an alias for vim – meaning if you start vi, vim will start.

Vim Tutor

Vim Tutor

- Using the vimtutor to "teach by use"
- Should be available on other systems you use vim upon.
- Can continue to learn.

```
[arc-t05@blog2 ~]$ vimtutor
===============================================================================
=    W e l c o m e   t o   t h e   V I M   T u t o r    -    Version 1.7      =
===============================================================================
     Vim is a very powerful editor that has many commands, too many to
     explain in a tutor such as this. This tutor is designed to describe
     enough of the commands that you will be able to easily use Vim as
```

```
     an all-purpose editor.
     ...
```

Vimtutor: Cursor Exit Delete Insert Append

```
Lesson 1.1: MOVING THE CURSOR
Lesson 1.2: EXITING VIM
Lesson 1.3: TEXT EDITING - DELETION
Lesson 1.4: TEXT EDITING - INSERTION
Lesson 1.5: TEXT EDITING - APPENDING
Lesson 1.6: EDITING A FILE
```

Summary

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                              Lesson 1 SUMMARY
  1. The cursor is moved using either the arrow keys or the hjkl keys.
        h (left)       j (down)       k (up)       l (right)
  2. To start Vim from the shell prompt type:  vim FILENAME <ENTER>
  3. To exit Vim type:     <ESC>   :q!   <ENTER>  to trash all changes.
             OR type:      <ESC>   :wq   <ENTER>  to save the changes.
  4. To delete the character at the cursor type:  x
  5. To insert or append text type:
        i   type inserted text   <ESC>         insert before the cursor
        A   type appended text   <ESC>         append after the line
NOTE: Pressing <ESC> will place you in Normal mode or will cancel
      an unwanted and partially completed command.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Vimtutor: Delete Move

```
Lesson 2.1: DELETION COMMANDS
Lesson 2.2: MORE DELETION COMMANDS
```

```
Lesson 2.3: ON OPERATORS AND MOTIONS
Lesson 2.4: USING A COUNT FOR A MOTION
Lesson 2.5: USING A COUNT TO DELETE MORE
Lesson 2.6: OPERATING ON LINES
Lesson 2.7: THE UNDO COMMAND
```

Summary:

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                              Lesson 2 SUMMARY
  1. To delete from the cursor up to the next word type:    dw
  2. To delete from the cursor to the end of a line type:    d$
  3. To delete a whole line type:    dd
  4. To repeat a motion prepend it with a number:   2w
  5. The format for a change command is:
              operator   [number]   motion
     where:
       operator - is what to do, such as  d  for delete
       [number] - is an optional count to repeat the motion
       motion   - moves over the text to operate on, such as  w (word),
                  $ (to the end of line), etc.
  6. To move to the start of the line use a zero:  0
  7. To undo previous actions, type:           u  (lowercase u)
     To undo all the changes on a line, type:  U  (capital U)
     To undo the undo's, type:                 CTRL-R
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Vimtutor: Put Replace

```
Lesson 3.1: THE PUT COMMAND
Lesson 3.2: THE REPLACE COMMAND
Lesson 3.3: THE CHANGE OPERATOR
Lesson 3.4: MORE CHANGES USING c
```

Summary

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                              Lesson 3 SUMMARY
  1. To put back text that has just been deleted, type   p .  This puts the
     deleted text AFTER the cursor (if a line was deleted it will go on the
     line below the cursor).
  2. To replace the character under the cursor, type   r   and then the
     character you want to have there.
  3. The change operator allows you to change from the cursor to where the
     motion takes you.  eg. Type  ce  to change from the cursor to the end of
     the word,  c$  to change to the end of a line.
  4. The format for change is:
         c   [number]   motion
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Vimtutor: Search

```
Lesson 4.1: CURSOR LOCATION AND FILE STATUS
Lesson 4.2: THE SEARCH COMMAND
Lesson 4.3: MATCHING PARENTHESES SEARCH
Lesson 4.4: THE SUBSTITUTE COMMAND
```

Summary

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                              Lesson 4 SUMMARY
  1. CTRL-G  displays your location in the file and the file status.
              G  moves to the end of the file.
     number   G  moves to that line number.
            gg  moves to the first line.
  2. Typing  /  followed by a phrase searches FORWARD for the phrase.
     Typing  ?  followed by a phrase searches BACKWARD for the phrase.
     After a search type  n  to find the next occurrence in the same direction
     or  N  to search in the opposite direction.
```

```
   CTRL-O takes you back to older positions, CTRL-I to newer positions.
 3. Typing  %  while the cursor is on a (,),[,],{, or } goes to its match.
 4. To substitute new for the first old in a line type    :s/old/new
    To substitute new for all 'old's on a line type       :s/old/new/g
    To substitute phrases between two line #'s type       :#,#s/old/new/g
    To substitute all occurrences in the file type        :%s/old/new/g
    To ask for confirmation each time add 'c'             :%s/old/new/gc
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Other Text Editors

In this section we will briefly cover two other common and popular text editors available for HPCs and used in Linux Command Line Interface

---

- Nano
- Emacs

Next Steps

Nano

GNU `nano` is a small and easy to use command line text editor. Besides basic text editing, `nano` offers features like undo/redo, syntax coloring, interactive search-and-replace, auto-indentation, line numbers, word completion, file locking, backup files, and internationalization support.

- Nano is terminal based and for use in command line.
- There are add-ons which allow it to be used in a GUI
- More information for using Nano on ARCC HPCs may be found here.

Emacs

Emacs is a portable platform for creating applications with a text user interface. It serves as a fully programmable text editor. Many users use Emacs in a variety of different ways.

- Emacs may be run in a command prompt (command line environment) or in a GUI.
- It can also be used as an IDE (Integrated Development Environment).
- More information for using Emacs on ARCC HPCs may be found here.

08.01 Next Steps

- **vimtutor**: Continue…
- **Google**: Many online tutorials:
  - LinuxFoundation: Vim 101: A Beginner's Guide to Vim
  - OpenSource: Getting started with Vim: The basics
  - FreeCodeCamp: How to Use Vim – Tutorial for Beginners
  - OpenVim: Interactive Vim tutorial
- **Cheat Sheets**: (find what works for you)

Vim Text Editor Cheat Sheets

- https://vim.rtorr.com/
- https://devhints.io/vim
- https://vimsheet.com/

Nano Text Editor Cheat Sheets

- https://www.nano-editor.org/dist/latest/cheatsheet.html

- Nano Cheat Sheet
- https://www.cheatsheet.wtf/Nano/

Emacs Text Editor Cheat Sheets

- Emacs Reference Page
- Emacs Cheat Sheet

08.02 Further Trainings: UWYO LinkedIn

08.03 Summary

In this workshop we have:

- Introduced users, using the Linux command line environment, to text editors: Specifically vim.
- Demonstrated how to:
  - Open, edit, update, save and quit a text editor.
  - Use the keyboard and shortcuts to navigate around the text file.
  - Use the command line to search for terms within the text file.

Next Steps

Previous	Workshop Home
Vim Tutor	Intro to Text Editors in Linux

Use the following link to provide feedback on this training: https://forms.gle/GHwxQ9X9cPtWMnAF8 or use the QR code below.