732A54

Big Data Analytics

Technical Introduction

Huanyu Li

LINKÖPING UNIVERSITY

Aims

- This session is held via Zoom online in SU rooms
- You will get to practice on working with relevant tools for the BDA labs
- Key topics
  - Linux Systems
  - Shell commands
  - Git and GitLab
  - Working on Sigma

Linux Systems

LINKÖPING UNIVERSITY

Linux Systems

- Prefer using the command line rather than GUIs
  - more powerful, scriptable and available on remote servers

- Ubuntu in SU rooms
  - connect to Sigma from a computer here
  - Using Thinlinc, ssh connection, or remote-ssh in VS Code
  - We will get back to this later

Shells

- The terminal is an application, the shell is the actual command interpreter
- Command line shells
  - sh
  - bash (default on most Linux systems)
  - zsh (default on macOS)
  - cmd.exe (default on Windows)
    - You can also install Git Bash on Windows, so you can run commands the same way as on Linux/macOS systems

Useful shell commands

- `ls` – list files in the current directory
  - `ls -l` – show more detail
- `mkdir dirname` – create a new directory
- `cat file.txt` – print file contents
- `cp src dst` – copy a file
  - `cp -r src dst` – copy a directory
- `module load *`

Useful shell commands

- module load in SU rooms
  - `module load courses/732A54`
- Thinlinc and VS Code are available for 732A54
  - `tlclient &`
  - `code &`
- module load on Sigma
  - `module load VSCode/latest-bdist`
  - `code &`

Apache Spark and PySpark

Apache Spark and PySpark

- Apache Spark is written in Scala and needs the JVM to run
- APIs are available for Scala, Java, SQL, Python, R
- This course uses Python and therefore the PySpark API
- Stand-alone and cluster mode
  - `run_local*.q`
  - `run_yarn*.q`
- Spark Core API for BDA1 lab
  - https://archive.apache.org/dist/spark/docs/3.5.1/api/python/reference/pyspark.html
- Spark SQL API for BDA2 lab
  - https://archive.apache.org/dist/spark/docs/3.5.1/api/python/reference/pyspark.sql/index.html

Git

LINKÖPING UNIVERSITY

Git

- Git is a distributed version control system
  - Distributed
  - Decentralized
  - GitHub, GitLab etc.

- Git is already installed on Unix systems
- Windows: Must install it manually – https://git-scm.com/download/win

GitLab

- gitlab.liu.se
- A GitLab repository has been assigned to you and your lab partners
- Log in using your LiU ID

Git commands

- Bring a copy to your local machine
  - SSH
    - `git clone git@gitlab.liu.se:732a54-big-data-analytics/vt-2026/a0.git`
  - HTTPS
    - `git clone https://gitlab.liu.se/*`
  - Download as a zip file

Simplified workflow for using Git in this course

- Key commands you need:
  - `git pull origin main`
  - `git add {file}`
  - `git status`
  - `git commit -m ”some informative message”`
  - `git push origin main`

Simplified workflow for using Git in this course

- Merge conflicts happen and are normal!
  - You can prevent them by avoiding working on the same file at the same time
  - Regularly pull changes from the remote repository to stay updated

- If it happens: Open the conflicted files, search for the conflict, solve it
  - `<<<<<<< conflict part>>>>>>>`
- Then stage, commit and eventually push the file

Secure Shell (SSH) & Keys

Secure Shell (SSH) & Keys

- Enables a secure remote shell connection (tunnel)
- Uses a keypair consisting of a public and a private key, default location is `~/.ssh`. Unix systems have a default key pair which you can use.
  - If not, use the `ssh-keygen` to generate a key pair
- On Windows (e.g. PuTTY) you must create them on your own or use WSL
- Add the SSH public key to GitLab (Settings -> SSH Keys)
  - https://gitlab.liu.se/-/user_settings/ssh_keys

Secure Shell (SSH) & Keys

- git can use https or ssh as the underlying protocol
- ssh uses key pairs instead of username and password
- If you log into any Git system, (GitHub, GitLab) for the first time, they usually want you to upload your public key for authentication

Basic SSH connection

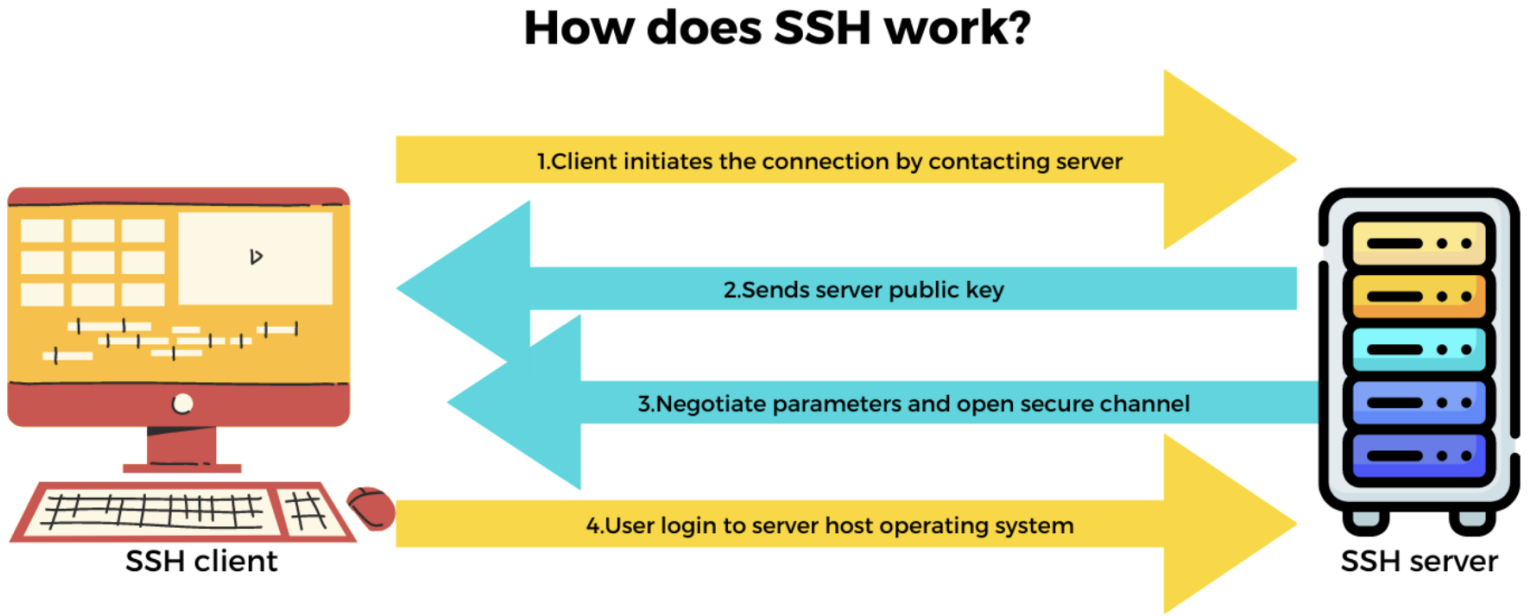


image source: https://www.cloudns.net/blog/what-is-ssh/

SSH with TOTP

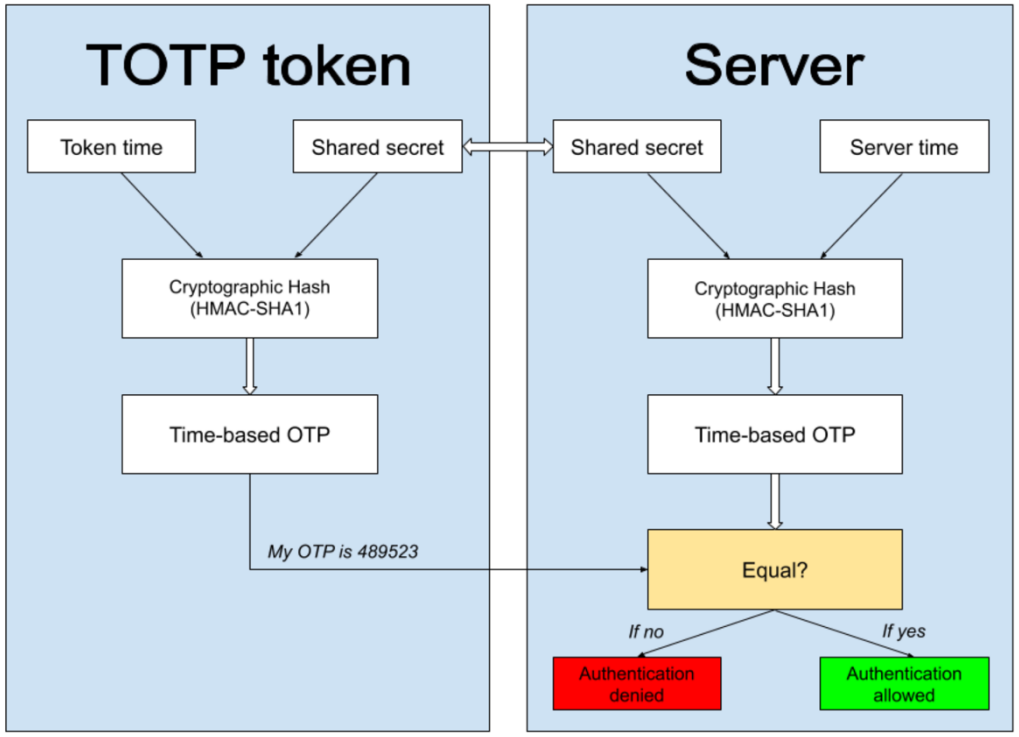
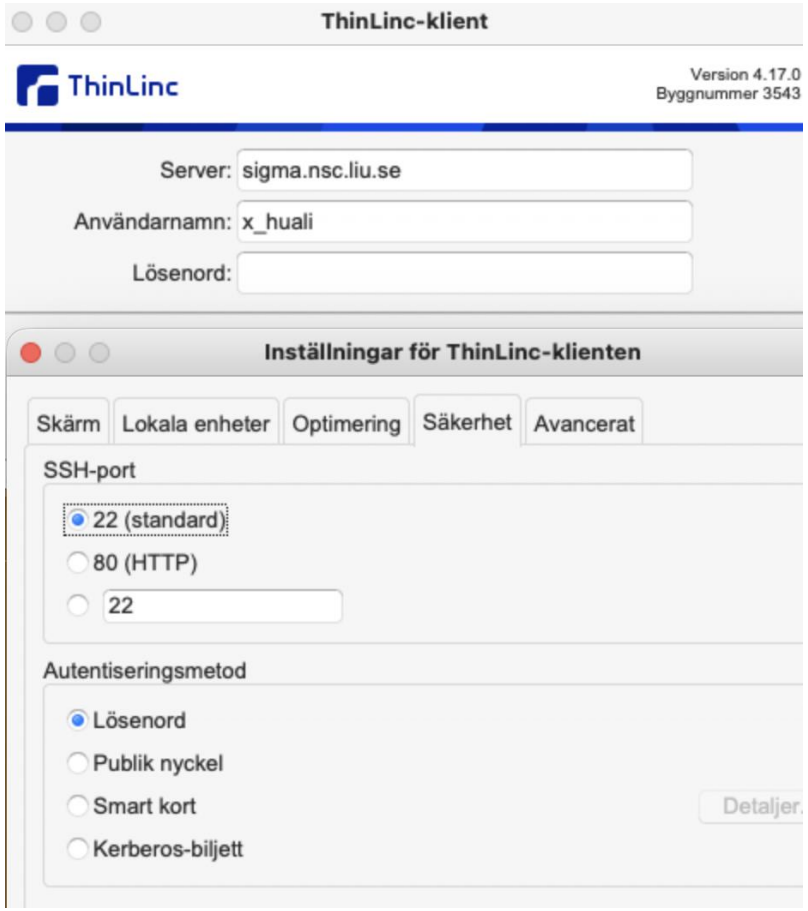


image source: https://www.protectimus.com/blog/totp-algorithm-explained/

Thinlinc is also based on SSH



Working on Sigma

LINKÖPING UNIVERSITY

Working on Sigma

- Connect to Sigma
  - https://www.ida.liu.se/~732A54/lab/bda-intro.en.shtml
- Set up Gitlab (for lab submission)
- Copy demo code and start to submit jobs

Working on Sigma

- Connect to Sigma – option 1

Option 1: Thinlinc connection

1. If you have already logged in to a computer in one of the SU rooms, you can run the following command to start the Thinlinc application.
   `$ module load courses/732A54`
   or,
   `$ module load courses/TDDE31`
   then run:
   `$ tlclient`
   If you are using your own computer, you can download **Thinlinc** directly.
2. A graphical user interface will appear. Enter the server address (`sigma.nsc.liu.se`), your account name, and password.
   You will be asked for a verification code from your authenticator app.
3. If your login is successful, you will see the graphical user interface of Sigma.
   Open a terminal window to prepare a working folder, copy the demo code, and check the files as shown in the steps below.
4. Check what exists in the default folder and create a new folder to use as your working folder for the labs.
   `[x_antli@sigma ~]$ ls`
   `[x_antli@sigma ~]$ mkdir BDA-lab`

Working on Sigma

- Connect to Sigma – option 2

Option 2: Remote-SSH connection via VS Code

You can find the instructions for connecting to the NSC server via VS Code **here**. The key steps are:

- Install the **Remote - SSH** extension in VS Code.
- Add a new SSH host: `sigma.nsc.liu.se` using your Sigma account credentials.
- Once connected, copy the demo code folder to your own folder at Sigma and edit your scripts via VS Code.

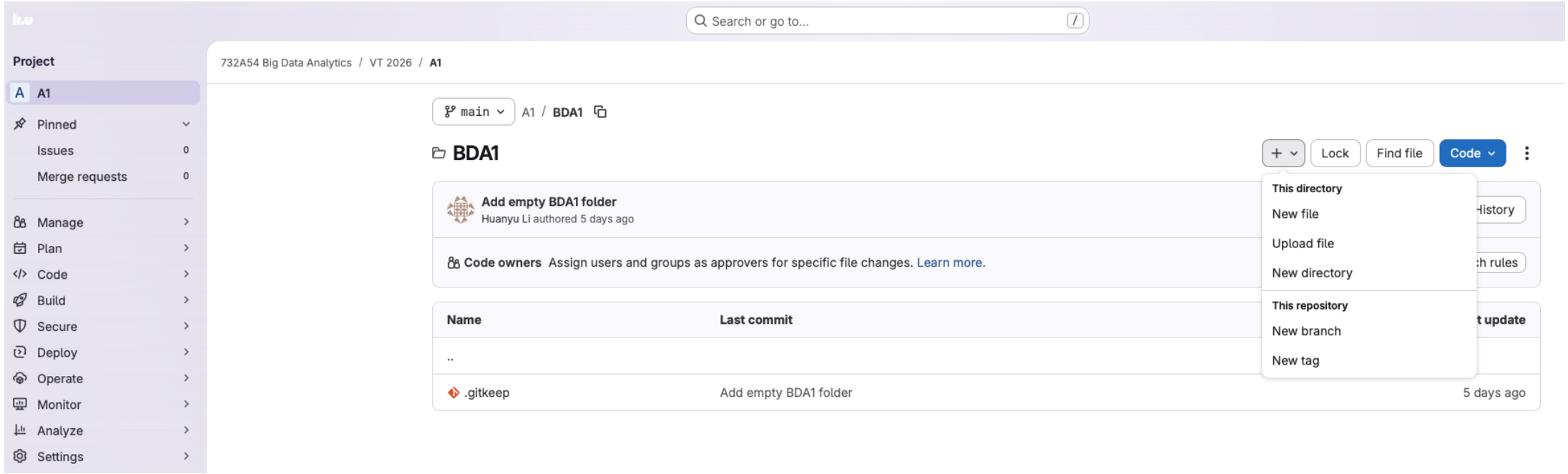
A screen recording demonstrating how to connect to Sigma using VS Code and Remote-SSH

Working on Sigma

- Option 1: use Git commands
  - Setup Gitlab, run a number of git commands
    - `git --version`
    - `git config --global user.name "LiuID"`
    - `git config --global user.email "liu-email"`
  - Generate a pair of SSH keys (private and public), public key can be given to GitLab/GitHub
    - `ssh-keygen -t ed25519 -C "liu-email"`
  - copy the public key and add it in your gitlab account, go to:
    - `cat ~/.ssh/id_ed25519.pub`
    - https://gitlab.liu.se/-/user_settings/ssh_keys
  - Test the setup and then clone the remote repository
    - `ssh -T git@gitlab.liu.se`
    - `git clone git@gitlab.liu.se:732a54-big-data-analytics/vt-2026/a0.git`

Working on Sigma

- Option 2: use the graphical interface



Working on Sigma

- Copy demo code and start to submit jobs

LINKÖPING UNIVERSITY