Repository of Open Educational Resources for Laboratory Support in Engineering and Natural Science-RELAB

Project Intellectual Output 6

WEB LAB TUTORIAL OF TECHNICAL DESIGN AND IMPLEMENTATION

“Enable conversion of off-line Lab setups to remotely accessible WebLabs – PART1/3”

version 0.1, Nov 19th 2021

Cognipix

"This project has been funded with support from the European Commission. This publication reflects solely the views of the author, and the Commission cannot be held responsible for any use that may be made of the information contained therein."

INTRODUCTION

It is frequently the case that Laboratory experiments are developed first in ad-hoc manner using available equipment and familiar computer resources, for on-the-site Lab work. While not ideal, this approach is very common.

But global pandemics, as well as growing interest in distant learning, are pushing requirements for Lab setups even further. Our goal in this part of RELAB project is to offer incremental approach: helping teachers leverage previous efforts in creation of Lab setups, but still make them remotely accessible as long as safety standards are met.

Under the above assumption, this document won't cover dedicated WebLabs, created from ground-up having in mind remote accessibility. These WebLabs should take into account this requirement from the very beginning, select appropriate platform/framework/APIs at several levels: Lab setup server, Proxy Server and the Client (Student's workstation). There are many benefits, e.g. well controlled environment allowing productive turnaround, also increased security aspects. Also, proposed solution is meant to be used by a teacher or small group of teachers without extensive IT support – this is the primary driver of our architectural decisions (since many different paths are possible).

Incremental approach will include three elements based on standard, open-source, off-the-shelf technologies. Still, some elements like Lab Setup registration, Student registration, real-time resource allocation, video streaming proxying, and admin ssh access via browser would require additional development effort.

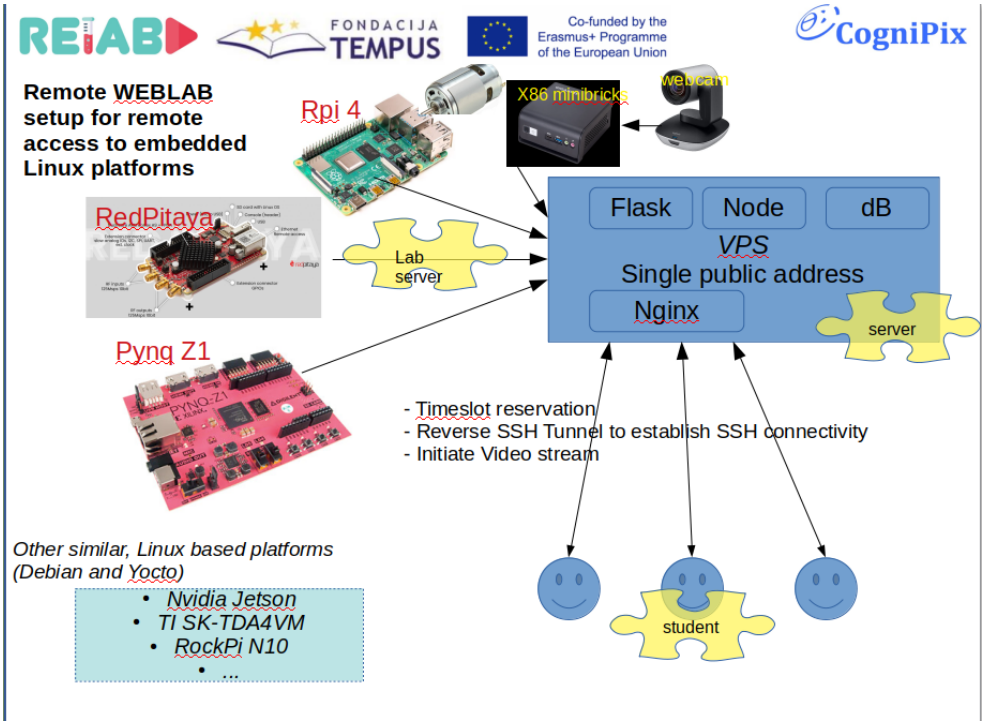
This intelectual output will be provided under terms of liberal licenses (final selection of MIT,BSD, or GPL licenses will be discussed with TEMPUS office). Necessary scripts, software packages and documentation will be hosted on GITHUB and relab.kg.ac.rs, to enable easy dissemination.

ARCHITECTURE

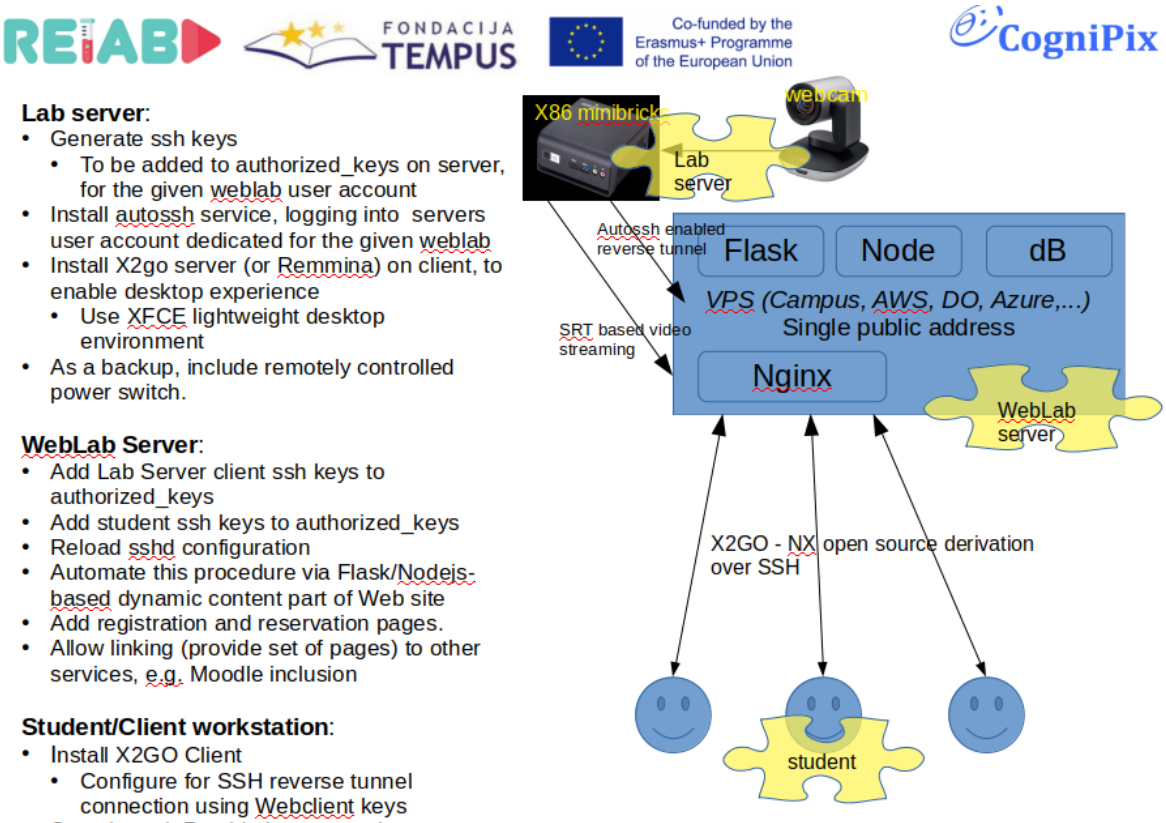
Proposed architecture includes 4 elements.

1. **Lab Setup** configuration with all the necessary equipment attached, as required for on-the-side student lab tasks, e.g. attached Arduino, motors, sensors, camera, etc. This is typically already available or beyond scope of this document.
2. **Lab Server** which is part of the Lab setup. We assume this is Ubuntu/Debian based lightweight computer (x86-based or ARM-based like RPi). We will provide detailed explanation about its configuration and necessary software add-ons. Important assumption here is that Lab Server does not have (IPv4 or IPv6) public address. We assume this setup is behind NAT, in most typical Lab LAN environment. In addition we won't discuss VPN option as it requires more complex IT setups, which may also create other security concerns.
3. **Weblab Public Server** is somewhat arbitrary term, but this is typically VPS (Virtual Private Server) available as Cloud asset, provided by University Campus IT service, or rented from Public Cloud Provider (AWS, DigitalOcean, Azure, Google, or some local cloud provider). It can be used for multiple purposes, like Web server, Chat/Blog Server, File Server, Streaming server, etc. We will limit the scope of discussion in this document to additional (very few) services/configuration modifications required for the proposed solution. This is also typically Linux-based server.
    - In follow-up step we will provide software solution for web-based registration of new Lab Setup resources and new Students. Resource (time slot) allocation for the given student and given setup will be provided. Necessary logging/journalling of activities for monitoring student work will be included and review mechanism provided. Finally carefully tuned (chroot/jailed ssh) security solution is needed and is also subject of this solution.
    - Proxy service of Video streaming will be required, and needs to include bandwidth policing.
4. **Client/Student workstation** needs very few software modifications. It can be either Windows or Linux-based. Only one or two additional software packages that can be easily installed (<5mins) are required.
    - As a stretch goal, we will try to provide browser based solution, that should enable students to access remote Lab setup without any new software installed, apart from signing up and providing supplied credentials.

LAB SETUPS WITH LINUX LAB SERVERS (RPI4, Pynq Z1, RedPitaya, MiniBrics+Arduino, etc)



THREE ELEMENTS: Lab Server, WebLab Server and Student/Client workstation



Lab server:

- Generate ssh keys
  - To be added to authorized_keys on server, for the given weblab user account
- Install autossh service, logging into servers user account dedicated for the given weblab
- Install X2go server (or Remmina) on client, to enable desktop experience
  - Use XFCE lightweight desktop environment
- As a backup, include remotely controlled power switch.

WebLab Server:

- Add Lab Server client ssh keys to authorized_keys
- Add student ssh keys to authorized_keys
- Reload sshd configuration
- Automate this procedure via Flask/Nodejs-based dynamic content part of Web site
- Add registration and reservation pages.
- Allow linking (provide set of pages) to other services, e.g. Moodle inclusion

Student/Client workstation:

- Install X2GO Client
  - Configure for SSH reverse tunnel connection using Webclient keys
- Stretch goal: Provide browser only access

Lab server configuration

Through the rest of this document we will assume lab setup which includes Linux-based thin server (e.g. x86 MiniBricks, or RPI4). Debian based distro Ubuntu will be used as example. Below instructions are provided to depict overall process – exact commands are provided in the appendix.

1. First step would be to configure, connect and verify initial Lab setup that is used by students coming to the lab.
2. Second step is generation of ssh keys using: ssh-keygen rsa
   - This step creates id_rsa private keys, and id_rsa.pub public keys
3. These keys need to be transfered to the WebLab Public Server and included in authorized_keys of weblab1 user (on WebLab Public Server)

WebLab public server configuration

1. Now, we should try to ssh to the WebLab Public Server, using:
   - ssh weblab1@relab.kg.ac.rs
     - We should be able to login without password prompt (so called passwordless ssh access)

1. Then, to make this available in opposite direction, i.e. to login to Lab server from WebLab Public Server, we need to configure and start autossh.service
2. We should verify previous step, by logging into Lab Server, from WebLab Public Server, using reverse ssh tunnel.

Client / Student workstation configuration

1. On student workstation, we need to generate keys (like in 2nd step) and add them to weblab1 authorized users.
2. To enable Remote Desktop experience, we will use X2GO software package, by installing the server side on Lab Server, and X2GO client side on Student workstation. X2GO server side is available for Linux only, whereas X2GO client is available for both Windows and Linux OS-es.