CA326 Third Year Project - Technical Specification

Squid Proxy for Safe Browsing
Completed: 20/02/23

Project Title: Squid proxy and Android app for safe browsing
Students: Aaron Crawford - 20336753
Ciaran Skelly - 20324213
Supervisor: Darragh O'Brien

Table Of Contents:

1. Introduction **2**
1.1 Overview 2
1.2 Glossary 3

2. System Architecture **3**
2.1 System Architecture Diagram 3
2.2 Android App 4
2.3 Raspberry pi 4

3. High-Level Design **5**
3.1 Context Diagram 5

4. Problems and Resolutions **6**
4.1 Time Constraints 6
4.2 Connecting Users 6
4.3 One Raspberry Pi 7

5. Future Work **7**
5.1 Whitelisting 7
5.2 Port Forwarding 8
5.3 More Platforms 8

6.1 Installation Guide **8**
6.2 Setting up Raspberry Pi 8
6.3 Installing Android App 9

References Used Throughout Project **9**
Squid 9
JSch 9
Android Application 10
Diagrams 10

1. Introduction

1.1 Overview

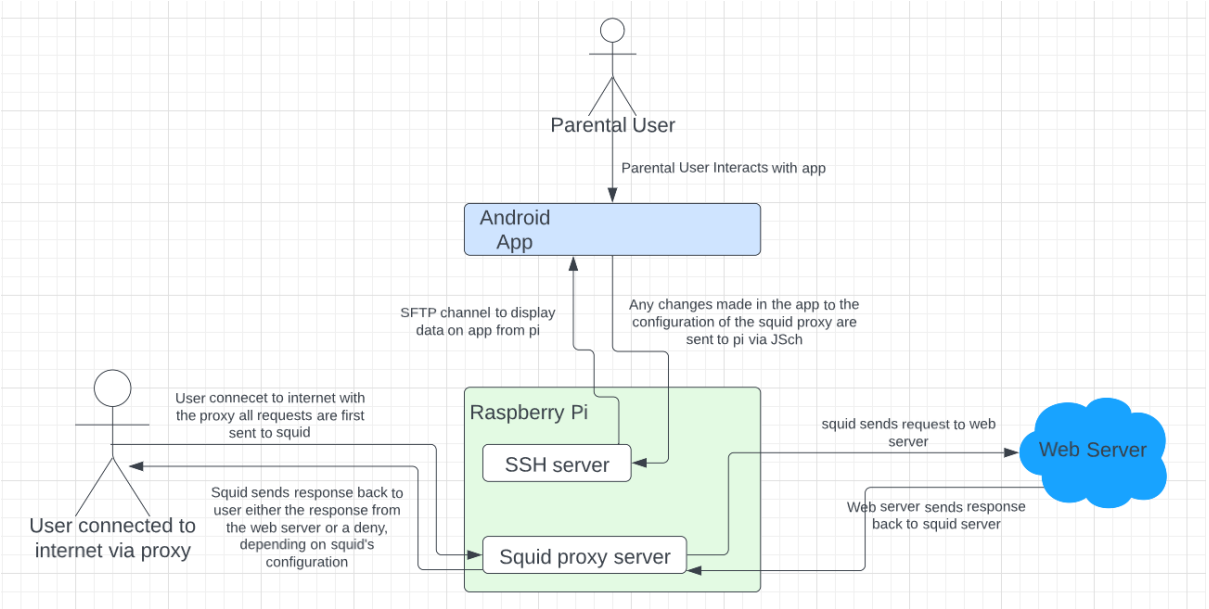
The purpose of this project was to create an app that would allow a user to control the internet access of all devices connected to the internet on the same network. We realised that while the internet is an amazing tool it can also be a dangerous place especially for young children and our aim was to ideally give parents but can be used by anyone, a tool to monitor and control their people's internet access. We would do this by having a squid proxy running on a raspberry pi in the home, the proxy would work as an intermediary between a user and a web server. This would allow us to see a user's activity such as what domains they are being visited, who is visiting them and when they are visiting them. We can then use this information to deny users access to certain websites by having squid work with a blacklist file and if a request for a domain is sent through the proxy that is contained in the blacklist file it will be denied and the user will be met with a page explaining so instead of the webpage they were looking for. We were also interested in having different rules for each user that is connected to the proxy as we knew that each home would have users of different ages that may be trusted more than others. So to achieve this each user would have their own account with their details stored on the pi, and when they opened up a for example google chrome they would be prompted to enter these details. Now each user as well as being restricted by the network wide blacklist could also be restricted by their individual blacklist. We then saw the opportunity to set time limits on individual users to restrict their internet access completely outside of specific set times so that if they were to attempt to use the internet outside of these times the proxy would deny them. All these commands can be used on the android application along with seeing all users and a users logs of what websites they have visited and how many times they have visited the website. This was done using JSch, that would allow us to use java to connect to an SSH server on the pi and execute these commands.This will allow parental users to see if inappropriate sites or certain sites are being visited too much and control them as they wish.

1.2 Glossary

Squid	Squid is a caching and forwarding HTTP web proxy
Proxy	A server that acts as an intermediary between a client requesting a resource and the server providing the resource
Raspberry Pi	A small single board computer
SSH server	A program that allows secure remote access to a computer or server over a network
JSch	A java implementation of SSH2 that allows us to connect to an SSH server
SFTP	Secure file transfer protocol allows file access, transfer and management over any reliable data stream

2. System Architecture

2.1 System Architecture Diagram



This is the system architecture for our finalised project. The raspberry pi lies at the centre of our system architecture with both parental users who are configuring the settings of the proxy and the users who are connected to the proxy server having to go through it.

2.2 Android App

The android app is what the users will interact with. It offers users a way to make changes to the configuration of the squid proxy remotely so that they never have to use the raspberry pi or go through the files of the squid. The app which is built in java uses JSch to send the commands the user would like to execute to the SSH server on the raspberry pi. These commands are then executed on the terminal of the raspberry pi and the necessary changes are made to the squids configuration. We can also open an SFTP channel using JSch which allows us to gather data that is stored on the pi and display it on the app.

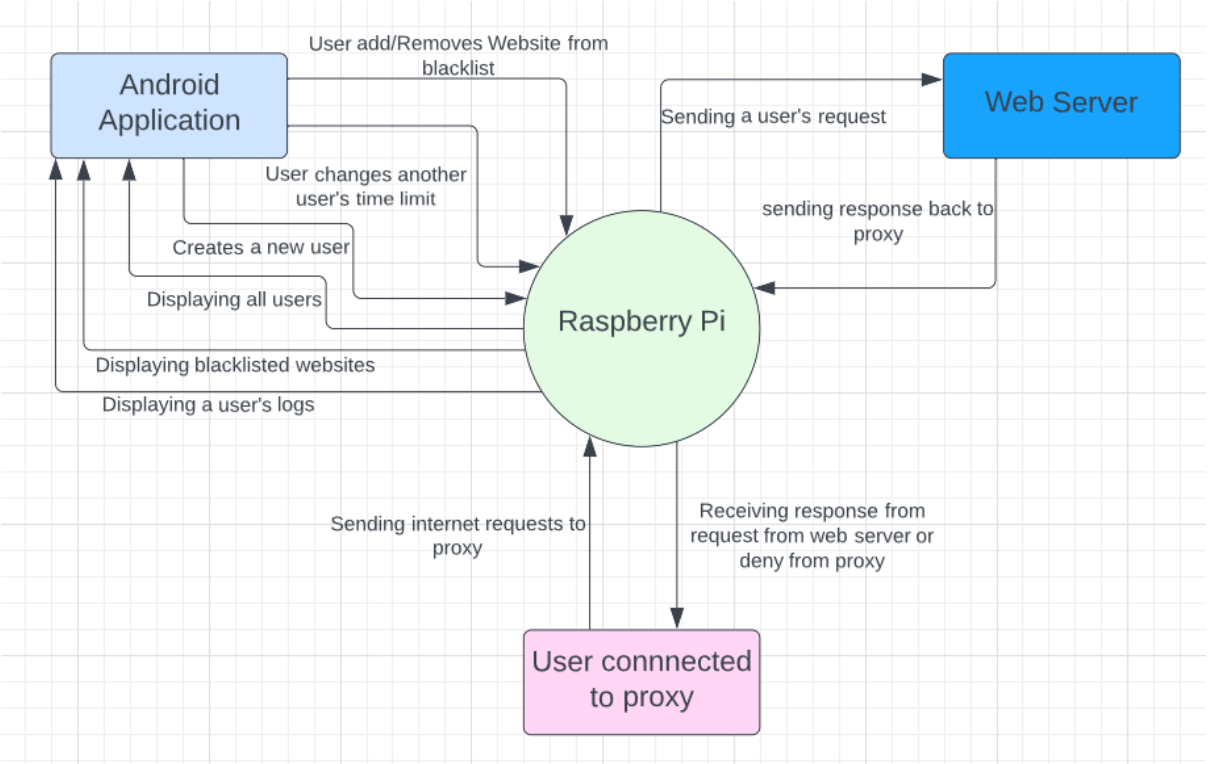
2.3 Raspberry pi

The raspberry pi is where the SSH server and squid proxy server are running. A user who is connected to the internet and using the proxy will have all their requests first sent to the squid proxy before it reaches the web server. Squid then goes through the rules in the configuration file for this user and sees if this request should be denied according to any of the rules. If the request is to be denied then the user is sent back a deny message which then displays a message telling them that they

are denied access by the proxy if they were clicking on a website that was blacklisted for example. If the request isn't denied then it is sent to the web server where the response first goes through the proxy and back to the user. This allows the access log of the squid proxy to update with the response it got from the web server. The SSH is how the user changes the configuration of the squid proxy. Changes are sent using JSch to execute commands on the terminal of the pi. It is also used to set up an SFTP connection between the app and the pi to display data such as users logs on the app.

3. High-Level Design

3.1 Context Diagram



The context diagram above shows the elements in the system and how they interact with each other. In our system the raspberry pi contains the squid proxy and SSH server which is controlled by the user that is using the android application. The android app can add and remove websites from a network wide blacklist while also changing individual user's blacklist and time restraints. It then receives from the pi a list of all users, a list of blacklisted websites and a user's log files containing the sites they have visited and how many times they have visited each site. The user that is connected to the squid proxy would

interact with the pi by sending their internet requests to the squid server for squid to then determine whether it should forward this request to the web server or to send back a deny message if the request doesn't pass one of the rules set out in the squid configuration file. The pi then interacts with the web server by forwarding on the request from a user and receiving the accompanying response.

4. Problems and Resolutions

4.1 Time Constraints

One issue we faced was time as learning all the new technologies necessary for this project took longer than expected such as squid, JSch, android studio and working with the raspberry pi itself. Therefore there were some features that when we began the project were interested in developing such as whitelisting, but couldn't as we wanted to ensure we had the capabilities to achieve everything we set out in the original proposal form before adding additional features and as learning these new technologies took longer than expected we did not have time for this

4.2 Connecting Users

Another issue we faced was connecting users to the squid proxy on the raspberry pi. At first we would let users connect based on their IP address, but this proved a problem as a device's IP address would constantly change and we would constantly have to change the squid configuration file to allow different IP addresses to connect to it. To fix this we allowed devices to connect based on their mac address as this would not change which worked while we were creating the network wide blacklisting however once we moved onto restricting individual users this also caused issues as we had no way to tell who was who. Finally to fix this we changed the squid configuration so that users had to be authenticated by a username and password before connecting. We achieved this using htpasswd, a flat-file used to store username and passwords for basic authentication. This then allowed us to restrict individual users and also search the log files for individual users activity.

4.3 One Raspberry Pi

As we only had one raspberry pi we had to figure out at the beginning of the project how we would test the code we had written such as the squids configuration and changing the squid's configuration using JSch. To get around this, Aaron who did not have the pi would use ubuntu on windows which allowed him to use the ubuntu terminal and run ubuntu command line utilities. This then allowed Aaron to run squid and the ssh server. However because these were running inside Ubuntu on his windows machine the only device that could connect to them were his own laptop unlike the raspberry pi that would allow any device on the network to connect. Aaron also had to create a maven project to develop the JSch commands necessary for the app as this would create a jar file that had the JSch dependencies needed for JSch to work included and could be run on the terminal of ubuntu. This problem lost us time as Aaron had to learn to use new technologies that wouldn't be included in the final project but were needed for testing throughout development.

5. Future Work

5.1 Whitelisting

This was an additional feature that we attempted to introduce while building this project however we came across some issues. We were able to successfully allow only a handful of domains through whitelisting however most websites these days will host their content e.g. videos, images ect on external domains that they will pull from. We could not figure out a way to find out what extra domains a website would need for them to display as they normally would without manually doing this by checking the access logs and seeing the extra domains which could never work for every website. This would be a feature that we would like to dedicate more time into trying to accomplish.

5.2 Port Forwarding

Port forwarding is a feature we would be interested in introducing in the future. We built this project with the idea that it would be used in the home and so port forwarding was not needed however being able to

connect to the proxy even while outside of the network would be an interesting development to work on.

5.3 More Platforms

We have built our product as an android application however in the future we would like to bring this application to apple products and also have a web version to increase the usability of the product.

6.1 Installation Guide

6.2 Setting up Raspberry Pi

To get the required packages for the project to work on your raspberry pi run the following command:

```
Sudo apt install squid openssh-client apache2-utils
```

Then Download the squid directory from https://gitlab.computing.dcu.ie/crawfoa4/2023-ca326-acrawford-safebrowsing and move it onto your raspberry pi to replace the directory called /etc/squid.

Now run the following commands to start the squid and ssh servers once they have been started once they will not have to be started again

```
Sudo service squid start
Sudo service ssh start
```

Run the following command to set the pi's ip as shown below

```
sudo ifconfig eth0 192.168.1.7 netmask 255.255.255.0
```

6.3 Installing Android App

To install the APK file on an android device you must first change your settings to allow Installation of unknown apps. This can be done by

opening settings navigating to apps > Special app access > Install unknown apps.

Select an app to use to install the APK file (e.g chrome or another browser). Select Allow from this source to allow APK files to be installed via that app. We recommend switching this back off after you have installed the app because this helps prevent installing unsafe or harmful apps.
On the browser that you have just allowed to install unknown apps navigate to our Application.

Click download, you may still get warnings about installing from unknown sources but proceed through them and the APK should begin to download. Once the download is complete, click open, it will ask you “do you want to install this app?”, click install. If you get another warning about unknown apps click install anyway.

References Used Throughout Project

Squid

squid : Optimising Web Delivery, http://www.squid-cache.org/.
Squid Web Cache wiki: Squid Web Cache documentation, https://wiki.squid-cache.org/.
Parvez, Husain. “How to Restrict Web Access by Time Squid Proxy Server.” *Distroid*, https://distroid.net/restrict-web-access-by-time-squid-proxy-server/.

JSch

“JSch - Java Secure Channel.” *JCraft, Inc.*, http://www.jcraft.com/jsch/.
Joksovic, Maja. “SSH Connection With Java.” *Baeldung*, 28 October 2020, https://www.baeldung.com/java-ssh-connection.

Android Application

“AsyncTask.” *Android Developers*, https://developer.android.com/reference/android/os/AsyncTask.

Diagrams

Lucid.app, https://lucid.app/.