

Hardware Random Number Generator (HRNG)



Author

PC Adeel Tahir

PC Rafid Azhar

GC Hamza Hafeez

ASC Mustafa Abubakar

Supervisor

Brig. Muhammad Tayyab Ali, PhD

Submitted to the faculty of the Department of NUST Electrical Engineering,
Military College of Signals, National University of Sciences and Technology,
in partial fulfillment for the requirements of B.E Degree in Electrical Engineering
(July), 2020

CERTIFICATE OF CORRECTIONS & APPROVAL

It is certified that the work contained in this thesis titled "Hardware Random Number Generator (HRNG)" carried out by P.C. Adeel Tahir, PC Rafid Azhar, G.C. Hamza Hafeez and ASC Mustafa Abubakar under the supervision of Brig. Muhammad Tayyab Ali, PhD, for partial fulfillment of Degree of Bachelor of Electrical Engineering, in Military College of Signals, National University of Sciences and Technology, Islamabad during the academic year 2019-2020, is correct and approved. The material that has been used from other sources has been properly acknowledged/referred.

Approved by

Supervisor

Brig. Muhammad Tayyab Ali, PhD

Date: July 2020

DECLARATION

No portion of work presented in this thesis has been submitted in support of another award or qualification in either this institute or anywhere else.

Plagiarism Certificate (Turnitin Report)

This thesis has been checked for Plagiarism. Turnitin report endorsed by Supervisor is attached.

Signature of Student

Adeel Tahir

Signature of Student

Rafid Azhar

Signature of Student

Hamza Hafeez

Signature of Student

Mustafa Abubakar

Signature of Supervisor

Brig. Muhammad Tayyab Ali, PhD

Acknowledgments

I am thankful to my Creator Allah Subhana-Watala to have guided me throughout this work at every step and for every new thought which You set up in my mind to improve it. Indeed, I could have done nothing without Your priceless help and guidance. Whosoever helped me throughout the course of my thesis, whether my parents or any other individual was Your will, so indeed none be worthy of praise but You.

I am profusely thankful to my beloved parents, who raised me when I was not capable of walking and continued to support me throughout every department of my life.

I would like to express special thanks to my Supervisor, Brig. Muhammad Tayyab Ali, PhD for his help throughout my thesis.

I would like to pay special thanks to Engineer Muhammad Sajid and Capt Rida Samee for their tremendous support and cooperation. Each time I got stuck in something, they came up with the solution. Without their help, I wouldn't have been able to complete my thesis. I appreciate their patience and guidance throughout the whole thesis.

I would also like to thank Brig. Muhammad Tayyab Ali, PhD, Amal Haider for being on my thesis guidance and evaluation committee and express my special thanks to Engineer Muhammad Sajid for his help. I am also thankful to family and friends for their support and cooperation. Finally, I would like to express my gratitude to all the individuals who have rendered valuable assistance to my study.

*Dedicated to my exceptional parents and adored siblings whose
tremendous support and cooperation led me to this wonderful
accomplishment.*

ABSTRACT

Have you ever thought how devices produce random numbers? Majority of the random numbers used in our lives are generated from software-based generators which are deterministic in nature. These software-based generators are called Pseudo Random Number Generators (PRNGs). These are suitable for everyday use but when it comes to sensitive scenarios, like casino game machines or for cryptographic applications, then this method is not secure and is prone to receptiveness.

The security of any information security device relies heavily and primarily on the Random Number Generator (RNG). The majority of commercially available RNGs have backdoors implanted in them. There is a need to devise a True RNG, based on some unpredictable physical phenomenon, coupled with a P.C., allowing arbitrary users to employ RNG in their equipment/systems, thereby eliminating the threats of RNG-based backdoors.

If we look at random numbers, they are used in our everyday life in various applications, however, true random number generators are slow and complex for many of the applications, but if we look at the pseudorandom number generators (PRNGs), they are secure enough for less sensitive day to day applications. Maximum number of random number generators are software based, however due to the rise of need of security, a high request is on the development of Hardware Random Number Generators (HRNGs), which are also known as True Random Number Generators (TRNGs).

HRNGs/TRNGs generate streams of random numbers which are unpredictable and non-repeatable. In recent times, the need for true random number generators has drastically increased and has become a striking research area. Therefore, researchers have started exploring this area in order to design and develop hardware random number generators to meet cryptographic requirements.

Key Words: *Backdoor, Security, Random Numbers.*

TABLE OF CONTENTS

ABSTRACT.....	viii
LIST OF FIGURES	xi
LIST OF ABBREVIATIONS	1
CHAPTER 1: INTRODUCTION.....	4
1.1 Project Outline.....	4
1.2 Problem Declaration.....	5
1.3 History of randomness	5
1.3.1 Tables of random sampling numbers.....	7
1.3.2 Electronic advancement.....	8
1.3.3 Computational Algorithms	9
1.3.4 Middle Square Weyl Sequence.....	10
1.4 Objectives.....	11
1.5 Applications	12
1.5.1 Games	12
1.5.2 Network security.....	12
1.5.3 Cellular Network operators	12
1.5.4 Military	13
1.5.5 Modern Politics.....	13
CHAPTER 2: TECHNOLOGICAL REQUIREMENTS OF THE PROJECT	15

2.1	Software	15
2.2	Libraries Used in Project.....	26
2.2.1	NumPy	26
2.2.2	Python Hashlib	26
2.2.3	Sockets.....	27
2.3	Hardware	27
2.3.1	Catching static of TV using the camera.....	27
2.3.2	Raspberry pi 4.....	30
CHAPTER 3: PROPOSED METHOD AND RESULTS.....		34
3.1	Proposed Method.....	34
3.2	Results obtained	38
CHAPTER 4: CONCLUSIONS AND REFERENCES		45
4.1	Conclusions	45
4.2	References	46

LIST OF FIGURES

FIGURE 1: OLD DICE	6
FIGURE 2: ROMAN COINS	6
FIGURE 3: TABLE OF RANDOM NUMBERS	8
FIGURE 4: NOOBS DOWNLOADING	16
FIGURE 5: SD CARD FORMATTER	17
FIGURE 6: SETTING UP NOOBS	18
FIGURE 7: INSTALLING NOOBS	19
FIGURE 8: COMPILATION OF VNC STEP 1	20
FIGURE 9: COMPILATION OF VNC STEP 2	21
FIGURE 10: COMPILATION OF VNC STEP 3	22
FIGURE 11: COMPILATION OF VNC STEP 4	22
FIGURE 12: COMPILATION OF VNC STEP 5	23
FIGURE 13: COMPILATION OF VNC STEP 6	23
FIGURE 14: COMPILATION OF VNC STEP 7	24
FIGURE 15: COMPILATION OF VNC STEP 8	25
FIGURE 16: COMPILATION OF VNC STEP 9	25
FIGURE 17: STATIC SOURCE 1	29
FIGURE 18: STATIC SOURCE 2	30
FIGURE 19: RASPBERRY PI 4	31
FIGURE 20: PHYSICAL SPECIFICATIONS	32
FIGURE 21: OVERALL DATA FLOW	34

FIGURE 22: NOISE SOURCES FOR A STATIC	35
FIGURE 23: OPERATIONS INSIDE RASPBERRY PI.....	36
FIGURE 24: ORIGINAL IMAGE	38
FIGURE 25: GRAYSCALE IMAGE.....	39
FIGURE 26: BINARY IMAGE	40
FIGURE 28: DECIMAL OUTPUT.....	41
FIGURE 29: HEXADECIMAL OUTPUT.....	42

LIST OF ABBREVIATIONS

RNG	Random Number Generator
TCP	Transport Control Protocol
RGB	Red Green Blue
ERNIE	Electronic Random Number Indicator Equipment
BSD	Berkeley Software Distribution
RTC	Real-time clock
VNC	Virtual Network Computing
ACK	Acknowledge
SYN	Synchronization
ADC	Analog to Digital Converter
HD	High Definition
POE	Power Over Ethernet
GPIO	General Purpose Input Output
DSI	Display Serial Interface

MIPI	Mobile Industry Processor Interface
CSI	Camera Serial Interface
I/O	Input Output

Chapter 1: Introduction

1.1 Project Outline

1.2 Problem Declaration

1.3 History

1.4 Objectives

1.5 Applications

CHAPTER 1: INTRODUCTION

1.1 Project Outline

Random Number generators play a vital part in our world with different applications, the most important being cryptography, computer simulation, network communication, statistical mapping, and many more. For developing good random numbers, all random number generators require some source that has lots of random noise. This is the very fact that we need a random entropy foundation, it is one big task for developing truly random number producers, this is since random entropy physical sources can be intricate to work with and expensive for the person. Finding a random entropy foundation that is widely available and is easy to use would be the best solution for this problem. We also need the random number generator to be easy to use and work in such a way many people could get the generated random numbers at one time in an easy to use way.

We came up with a unique source for generating random numbers. We used a video source for generating random values. The video source selected was the static of a television. The static of the television is a form of noise shown on the television screens, this phenomenon of black and white occurs when the receiver of the antenna receives no outside broadcast. The random pattern shown in the TV set is an arbitrary array of points. This noise is due to electrical clatter of signals in the atmosphere, thermal noise, and the Additive white Gaussian noise selected by the antenna probe. All these noise entropy foundations are separated into many types of noises that combine to form one entropy noise source for the static noise shown on the television. The most important of these entropy noise sources is the cosmic background radiation in the sky as well as some radio signal from radio set devices. The thermal entropy noise added to the overall noise is the noise of the inner electronics of the television. The biggest contributor from these inner electronics are the small electronic transistors and components present in the device.

Live static video of the TV set is then detected by a camera. This camera then sends a video to a raspberry pi. We have provided the raspberry pi with a self-made python program. The python program utilizes the OpenCV library, NumPy, and hashes to generate the random output numbers. The raspberry pi feeds the program with the noise video from the camera. The program reads the video frame by frame. Each RGB frame is first converted into a greyscale frame, which is then converted into a binary frame. The binary frame is then further read pixel to pixel, each 0 and 1 from the binary image is recorded and stored in the memory. This binary value from all the frames of the video is combined and sent into a SHA-256 hash function. The hash function then utilizes this value and stores it in the device.

A TCP based linking is then launched between the raspberry pi server along with their clients. Any authorized client can create a secure TCP connection with the raspberry pi and ask for random numbers. The choices given to the client are how many frames does it want to use to generate

random numbers and in which format does it require the random numbers in either binary or hex. Upon selecting a choice, the values are then sent from the raspberry pi to the client in a secure manner.

1.2 Problem Declaration

Basically, a true random number producer is a type of technology which creates symbols or numbers that are difficult to determine, merely based on accidental guess, but the problem with these type of generators is that the numbers predicted are not completely random, and there is a chance of repetition, so these are classified as not completely random number generators or pseudorandom number makers which create numbers which seem random to the standard human judgment but are actually very deterministic in reality. This is the problem statement that my colleagues and I have set out to solve and create random figures of digits which will be truly random and are not deterministic all around. Before we go into details, we will tell you a little history about random number generators, now there are many stages or randomness which contribute to the development of many methods that can be used to make truly random digits and the proof of this randomness data can be seen on this earth centuries and centuries ago during the old age of mankind. Some of these methods of randomness are very well known to us such as the example of two rolling dice, the coin flipping, the shuffling of playing cards are all good examples of randomness. Other than these old fashion techniques, there are several computational methods that we can use to produce random numbers, and they have been used throughout history, but all these methods have failed to produce true randomness. Although some are successful, they are tested for randomness on the intent to measure the unpredictable results, so they can be used in cryptography.

1.3 History of randomness

As discussed in the problem statement, some very early random number generators were the dice the cards and the coin-flipping. Around 6000 years ago some coins were revealed in Iran and China, they are an example of how people way back then used randomness in their day to day lives. Of course, they had an imagination that these were not truly random but were the signs of existence of God and that this was his conclusions about them. Examples of such devices used by them have been demonstrated in the figure 1 on the next page.



Figure 1: Old Dice

Around 3000 years ago, during the time of the Roman Empire, coins were very popular in their period. They would decide, what a person's fate would be life or death. An example of those coins used in that time period is shown in the figure 2 below [1].



Figure 2: Roman Coins

If we look at the history of random number machines there are very few of them which uses static or image as a source of their entropy. There is a very famous example in history in the year 1997 a scientist named Robert introduced a new method design to create random digits. What he did was that he combined both the true random digits which was created using a hash function and a software algorithm based pseudorandom number generator. During the year 2008 some Chinese scientists proposed a new method for creating truly random numbers from additive white Gaussian noise (AWGN) with images which were taken by a computer camera. During the year 2011 some new Chinese scientist propose another method for generating random value digits, their idea was that they would scan the hand of a person and since the lines of the hand of each person is unique then that would create true randomness of numbers. But not much work has been done in this field for its further advancement.

1.3.1 Tables of random sampling numbers

It was very inconvenient for statisticians and mathematicians to take random numbers by throwing simple dice following a suggestion which was made by Karl Pearson, Tippett in 1927, they published a table of 42,700 arbitrary numbers which were taken from an old survey description about the town's population.

In the year 1927 a scientist named Fisher published a book which had a random collection of digits which were picked from a book of logs of numbers. By the year of 1938 a list of random digits was derived from the yellow pages of a phone book directory. Then they used various trials to determine the sporadic performance of random digits. In the year 1939 a scientist by the name of Mr. Kendal made an electronic device that was able to yield random digits on a disk which would be grouped into about ten segments. These segments were moved in a circular manner with the help of gears, at a rate of 2.5 turns per 60 seconds, lighting was projected at random time each at two seconds, this projected sector of the segment was logged by people, they with this devices help made a table of 100,100 random digits. This table is described on the next page as figure 3.

APPENDIX
Random Sampling Numbers Produced by the Machine

<i>1st Thousand</i>							
23157	54859	01837	25993	76249	70886	95230	36744
05545	55043	10537	43508	90611	83744	10962	21343
14871	60350	32404	36223	50051	00322	11543	80834
38976	74951	94051	75853	78805	90194	32428	71695
97312	61718	99755	30870	94251	25841	54882	10513
11742	69381	44339	30872	32797	33118	22647	06850
43361	28859	11016	45623	93009	00499	43640	74036
93806	20478	38268	04491	55751	18932	58475	52571
49540	13181	08429	84187	69538	29661	77738	09527
36768	72633	37948	21569	41959	68670	45274	83880
07092	52392	24627	12067	06558	45344	67338	45320
43310	01081	44863	80307	52555	16148	89742	94647
61570	06360	06173	63775	63148	95123	35017	46993
31352	83799	10779	18941	31579	76448	62584	86919
57048	86526	27795	93692	90529	56546	35065	32254
09243	44200	68721	07137	30729	75756	09298	27650
97957	35018	40894	88329	52230	82521	22532	61587
93732	59570	43781	98885	56671	66826	95996	44569
72621	11225	00922	68264	35666	59434	71687	58167
61020	74418	45371	20794	95917	37866	99536	19378
97839	85474	33055	91718	45473	54144	22034	23000
89160	97192	22232	90637	35055	45489	88438	16361
25966	88220	62871	79265	02823	52862	84919	54883
81443	31719	05049	54806	74690	07567	65017	16543
11322	54931	42362	34386	08624	97687	46245	23245

Figure 3: Table of Random Numbers

1.3.2 Electronic advancement

Novel scientists doing the Monte Carlo simulations or reading lists of random digits from books became very sluggish also the storing capacity was very small, so some methods were taken to resolve these issues. The first was for our fast physical electronic device and the second, something that imitates randomness in software through a purely deterministic algorithm.

Two main approaches:

- An, swift computerized electronic device.
- A computer software-based algorithm for creating random digits.

1. Swift Computerized Electronic Device

The shuffling of cards, throwing of six spotted cubes or the tossing of bits of silver have been used for ages but with computer machines, came a huge progression in electric random hardware like the counters of the random devices and the electronic noise which is periodically sampled are faster and convenient methods, there are thousands and of articles and patents that would describe physical RNG's, they use thermal electronic clatter, or devices which are built on Atomic quantum Mechanics, Zener diodes in electronic circuits or reactivity in the air sensed by a radioactive detector.

- **ERNIE**

ERNIE is a case of a device for producing random numbers. It stands for electronic random number's indicator, equipment. It could produce 60 random numbers at a rate of one sec, hence it was utilized in the British saving bonds lottery to determine the winning numbers.

It worked on such a phenomenon that it had two glass tubes at each end which was filled with neon gas and high voltage would be applied at each end of the glass, hence current would be produced inside the tube this current would cause a change and shift in electrons which would create noise, this noise was amplified and collected to be used as random numbers nowadays ERNIE 4 is being used since 2005. It takes arbitrary bits from the electrons current noise, which is produced inside of its transistors and components [2].

1.3.3 Computational Algorithms

Most of the random numbers that are used today are mainly computer-generated random numbers and they used for pseudorandom number generators or PRNG to generate randomness there are dependent upon algorithms to do this, but these algorithms are not completely random, and eventually, the sequence repeats itself or the memory that it uses that grows out of bounding. These computer-based software algorithms do make lengthy figures with non-deterministic random characteristics, but ultimately some of the sequences is bound to repeat. These types of numbers are acceptable many times, but they fail when they are compared to the random numbers which are generated from random noise sources taken from the environment. The most famous software algorithm is the linear congruential random producer. It uses the following repetition formula stated as equation 1.

$$X_{n+1} = (aX_n + b) \bmod m \quad \text{Equation (1)}$$

In this formula, the alphabets a, b, n and m are big numerals for generating numbers and the term x_{n+1} is the following X, as the sequence of arbitrary figures. One less than Modulus m-1 is the extreme quantity of figures that this mathematical equation can create. In order to get more larger phases and improved numerical, mathematical characteristics, the reappearance equation can be elongated to mathematical matrices. In order to avoid certain nonrandom deterministic properties of random numbers, a mastered number generator is used with a collection of several sub arbitrary digit creators, which have dissimilar standards of the mathematical multiplier constant. While a master generator would select several different random numbers from its sub generators [3].

Most computing programming languages nowadays include a function or class that provides arbitrary numbers, they are planned in such ways that they give random bytes, words, or floating-point numbers, which are uniformly distributed. The common arbitrary number makers which are present in computer language programs like Python, C++, Swift, and Java are based on the famous Mersenne twister computer software algorithm, but such a method is not good enough for cryptography purposes. Such devices have very poor mathematical properties and are very deterministic, they will often show recurrence designs after only thousands of trials would be made. The frequently use of the real-time clock (RTC) of the computer as the seed value would provide enough complexity in randomness for certain tasks like playing video games but are still unsuited for very advanced randomness scenarios such as network security or cryptography applications or statistical numerical analysis.

But there are more superior random digit creators that are present on most OS systems like Linux Mac OS X, IRIX by Silicon Graphics, Solaris (Unix) by sun Microsystems, or crypt Gen random for Windows.

1.3.4 Middle Square Weyl Sequence

The flaws that were present in the previous models, like the original middle square generator, were all removed in the middle square Weyl Sequence. The advantage that we get is that the convergence to zero is prevented. It also helps in solving the repeating cycle problem. Its implementation is shown on the next page [4].

```

#include <stdint.h>

uint64_t x = 0, w = 0, s = 0xb5ad4eceda1ce2a9;

inline static uint32_t msws() {
    x *= x;
    x += (w += s);
    return x = (x>>32) | (x<<32);
}

```

1.4 Objectives

We will use hardware that would take sensory input from our environment and with that data, we will generate random numbers.

These random numbers will: -

- Be true random numbers.
- Provide the necessary security.
- Require no pre-defined algorithm which will be known to other companies or organizations.

Our project completes the following objectives, which are described below.

- To produce truly random numbers that are completely random and cannot be repeated.
- A simple and cheap solution to the problem of making true random numbers. Whose solution is an expensive and complicated one.
- A device that is easily compatible with existing computer systems and software.
- A device that is easy to use and has a simple user interface.

1.5 Applications

Believe it or not, but random numbers have countless applications in our day to day tasks and are now a daily part of our lives. Described below are many applications that vary from games, mobile phones, cryptography, security and communication.

1.5.1 Games

Nowadays many generators of random digits are being used in gambling dens all around the world they decide the trial of the outcome of every game, which is electronic in nature even the gambling fruit machines mechanical wheels appear to whirl on the LCD, but these LCD are actually spinning for the acting purposes. Their time to stop is determined by the computer software by when it was programed to stop not when the person decided to pull the lever. Manny gambling dens use this to their advantage and reprogram the machines to make the people lose their money, while making it look like fun. That is why government inspectors conduct visits and supervise the electronic machines to prevent this type [5]. Other than casinos, simple computer games such as Mario, Tetris, Temple Runner, Subway Surfer and many more also rely on random number generators to generate random simulations for the game to continue.

1.5.2 Network security

In today's technological age, many technological advancements have made life very easy for us, but as they have made life easy, it comes with it catches such as electronic security. Random number generators are used in cryptography to encrypt messages in order to hide their true meaning. Many algorithms used in cryptography use random numbers to encode their messages. This ranges from simple encoding techniques to too many complicated ones. The more complex the algorithm is, the most secure it will be, but as a drawback it will require more processing power.

1.5.3 Cellular Network operators

Cellular network operators such as Zong, Telenor and Ufone etc, often require the use of random numbers firstly due to the rise of 5G random number generator. Which is being used in the visible spectrum for secure communication on the 5th generation network this will ensure a secure network other than that we have the mobile scratch cards which have random numbers on it, they are used to put credit in our Sims. Moreover, mobile networks phone numbers must also be random, so random number generators have use of great importance in this industry.

1.5.4 Military

Since the age of fighting with guns, tanks and cannons have come to an end this new century warfare includes cyber warfare attacks, for this reason we must also be prepared in this field random number devices have an extensive array of usages in the military the most important thing is securing two-way communications, that information is not leaked to the enemy. A good example of this can be taken from World War Two when the British got into the famous cryptographic machine enigma [6] and were able to decipher their communications, as a result winning the war. If we would have a machine that would produce true random numbers, then it would be almost impossible for the enemy to decode our messages and our communication would stay intact.

Another use for the military would have security keys for important rooms that contain secret information or equipment numbers on these keys would change randomly by every hour or every day so that it would be impossible for anybody without the proper clearance to gain unauthorized access.

1.5.5 Modern Politics

Believe it or not, but modern politics in this world also makes use of random numbers in some Anglo-Saxon legal systems like the which are being used in the UK, and the United States. It consists of a process known as allotment or sortition, for the selection of jurors in a jury. Some proposals have been used to bring this process for use in the government of Iraq, and various proposals for upper houses chosen by allotment. Many scholars have studied the good potential of a random selection of personal in politics and organizations of governments all around the world. [7].

Chapter 2: Technical Requirements of the project

2.1 Software

2.2 Libraries used

2.3 Hardware

CHAPTER 2: TECHNOLOGICAL REQUIREMENTS OF THE PROJECT

2.1 Software

The programming language selected for the project is python because it is the best-used language for data manipulation, especially if the data is an image. Our project uses four main libraries that are open source for image manipulation.

The main library used for the project is Open CV. This library is responsible for recording the video from the computer camera and changing that video to a mathematical matrix arrangement. This library provides many methods for manipulating each frame of the video individually.

The next library used is NumPy, which is required by the Open CV to work properly, as the Open CV requires to deal with mathematical operations requiring matrices. This is so because each image is a set of matrices with each individual pixel on its own also being stored as matrices in the memory. So overall, this becomes big data of lots of matrices, and to deal with such big numbers, we needed the NumPy library.

Another library used is the hash lib library. This library is used for manipulating the data received in the binary form after the image manipulation is done by Open CV. It has many useful methods for creating many hashes out of a given input data.

The last library that we used is the socket library. This library has many different methods to program the sockets of the hardware the program is running on. This library is important because it helps us to form a TCP based client and server connection.

Setting up working environment

The guidance for installing the software and then setting it up in an effective manner is provided below.

Downloading Noobs

The default installer to be used for the raspberry pi is NOOBS. This installer helps us install the operating system on which the raspberry pi4 device will work on, this OS is called the Raspbian. Although we can use any other Linux based operating system, for simplicity purposes, we used the Raspbian. NOOBS offline installer is supposed to be downloaded from this link provided

above. Also, to be noteworthy that the NOOBS installer is to be downloaded on your own computer. The raspberry pi, for now, must not be used.

Only the NOOBS offline installer is to be used [8].

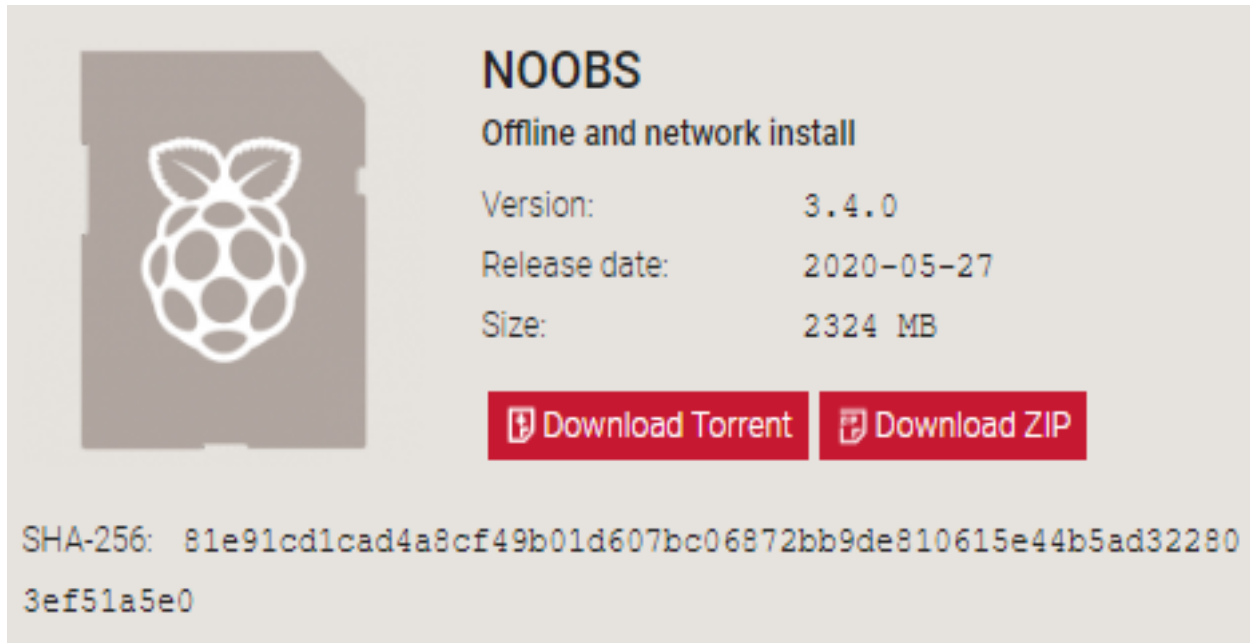


Figure 4: NOOBS Downloading

Downloading and installing the SD Card Formatter

The downloaded and installed version of the Raspbian OS on the raspberry pi 4 device, has some system files which will be transferred into an SD card. So, we need a tool first to format the SD card such that there are no errors occurring later, due to some virus OR malware present [9].

This SD card, tool formatter of the OS is vital for installing the Raspbian in Raspberry pi. The SD card tool formatter can be downloaded from the following link.

“www.sdcard.org/downloads/formatter/.com”

When our SD card tool formatter will be downloaded in our desired computer. The steps needed to install the SD card formatted are quite simple.

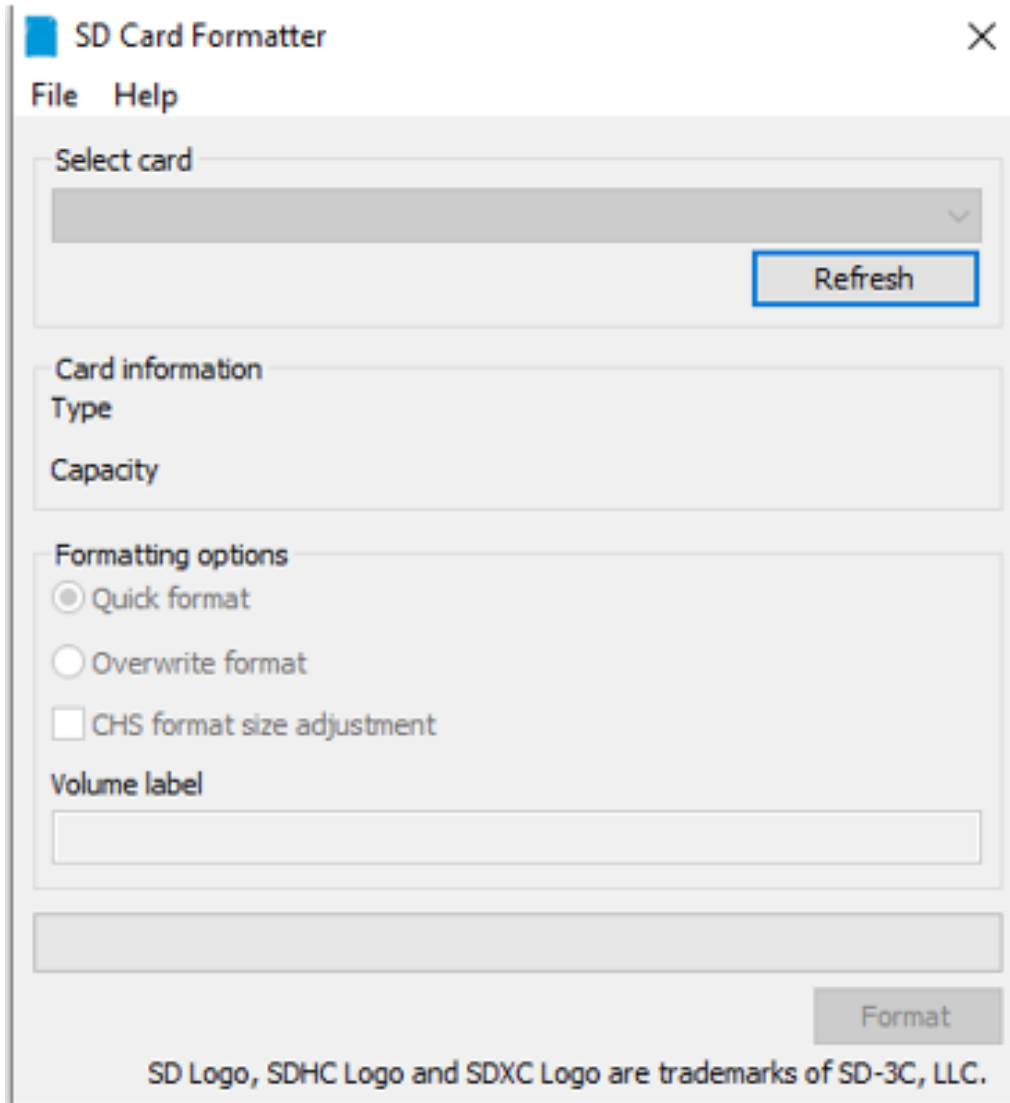


Figure 5: SD Card Formatter

Now the requirement is that you insert an SD Card into your computer, and once it shows up on the formatter, you need to select the SD card, choose the type of format of your choice, and then select format to set-up the SD card tool.

Copying NOOBS in SD Card tool

Now copy all documents in the folder relevant to the NOOBS folder and then paste them directly into the SD Card chip.

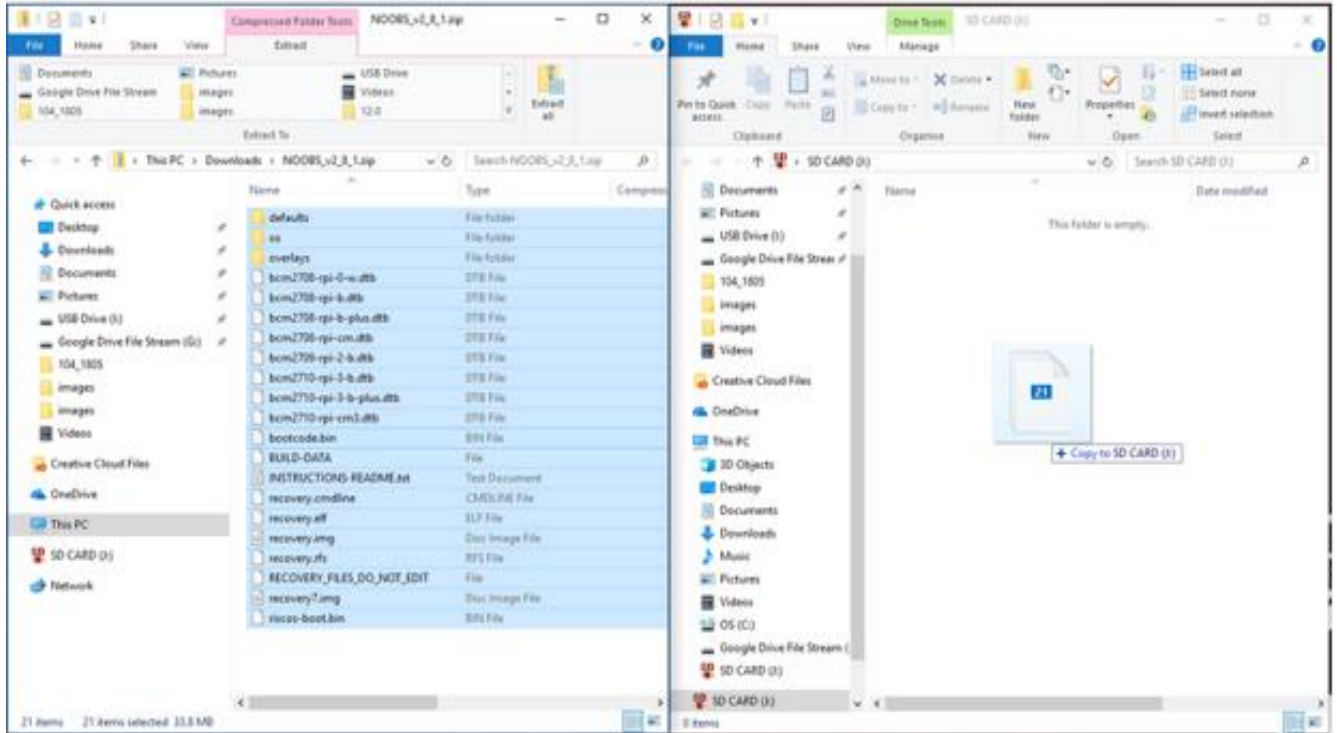


Figure 6: Setting up NOOBS

Installing Raspbian

Once the NOOBS folder has been successfully copied into the desired SD card chip using the SD Card tool formatter, we must unmount the SD Card chip from our Laptop or computer in a safe manner. Now once the SD Card chip will be unmounted, we mount the same SD card chip containing our NOOBS folder into the Raspberry pi. Once the SD card is mounted, we now connect the raspberry pi to the desired LCD screen. Power on the LCD screen, the very first time of the startup, the following window shall appear.

Shown on the next page as figure 7.

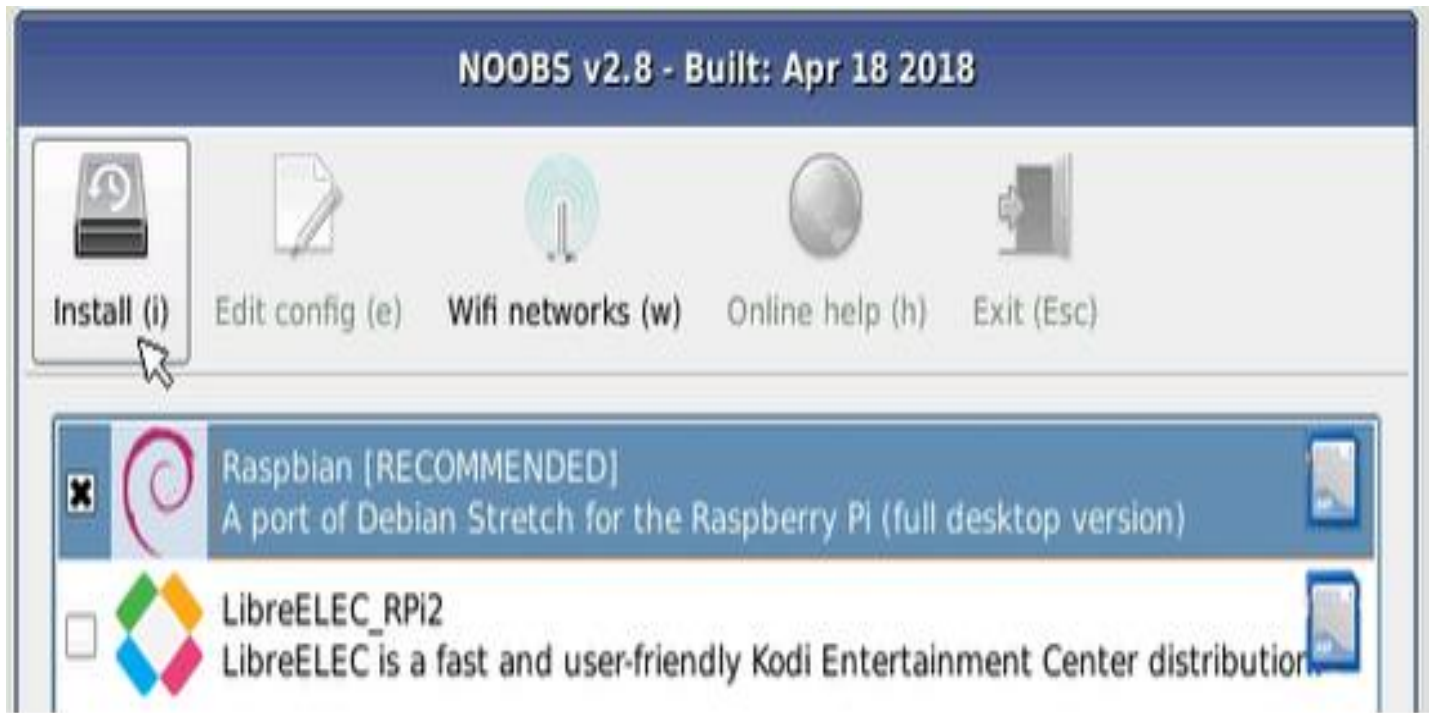


Figure 7: Installing NOOBS

In the menu displayed on your LCD screen with raspberry pi attached, select "Raspbian" as your option by checking the box and then pressing the install icon button on the upper toolbar in the menu.

After this, the setup would take some time, 30 to 45 minutes, to be exact, but this time can also vary to which type of the Raspbian OS is to be installed on the device.

After the installation of Raspbian is done on the Raspberry pi 4 device, we turn the device off by removing its power.

Now turn the raspberry pi 4 device on again.

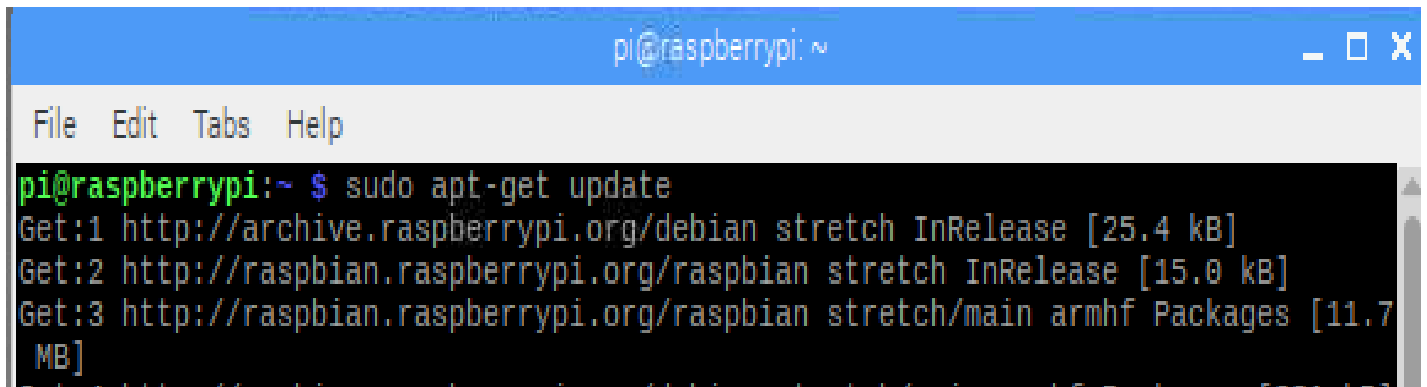
The raspberry pi would ask for credentials to log into the operating system.

The username required in the raspberry pi 4 device would be "pi," and the default password will be "raspberry."

With this, we have successfully installed the Raspbian operating system in our raspberry pi.

Setting Up VNC Server

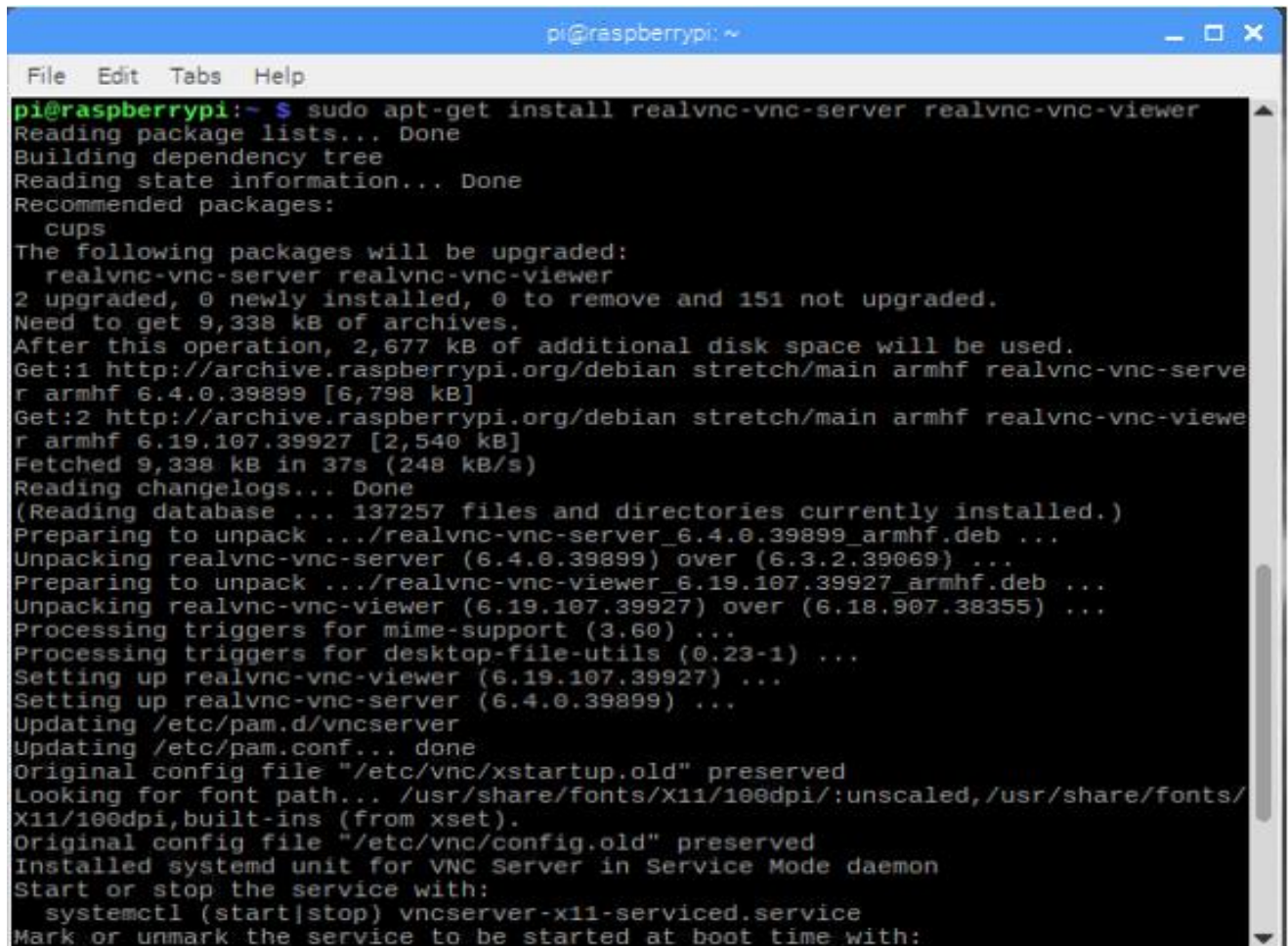
VNC, which stands for Virtual Network Computing is a very popular desktop remote sharing system. This system allows anyone to control the raspberry pi devices from anyone around the world over the use of the world wide web called the internet. Our first step now would be to configure the VNC server in such a way that we can remotely control raspberry pi over the internet. To configure the VNC viewer open the command control terminal in the raspberry pi device and type the following command “sudo apt-get update.”



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ sudo apt-get update  
Get:1 http://archive.raspberrypi.org/debian stretch InRelease [25.4 kB]  
Get:2 http://raspbian.raspberrypi.org/raspbian stretch InRelease [15.0 kB]  
Get:3 http://raspbian.raspberrypi.org/raspbian stretch/main armhf Packages [11.7  
MB]
```

Figure 8: Compilation of VNC Step 1

When we get conformation that the update is done then we type the following command:
“ sudo apt-get install RealVNC-vnc-server realvnc-vnc-viewer”.

A terminal window titled 'pi@raspberrypi: ~' with a menu bar containing 'File Edit Tabs Help'. The terminal output shows the execution of the command 'sudo apt-get install realvnc-vnc-server realvnc-vnc-viewer'. The output includes package list reading, dependency tree building, state information reading, and package recommendations (cups). It details the upgrade of two packages, the disk space requirements (9,338 kB), and the download of packages from the Raspberry Pi archive. The installation process is shown, including unpacking, processing triggers for mime-support and desktop-file-utils, and setting up the vncserver service. The terminal ends with instructions on how to start or stop the service using systemctl.

```
pi@raspberrypi:~ $ sudo apt-get install realvnc-vnc-server realvnc-vnc-viewer
Reading package lists... Done
Building dependency tree
Reading state information... Done
Recommended packages:
  cups
The following packages will be upgraded:
  realvnc-vnc-server realvnc-vnc-viewer
2 upgraded, 0 newly installed, 0 to remove and 151 not upgraded.
Need to get 9,338 kB of archives.
After this operation, 2,677 kB of additional disk space will be used.
Get:1 http://archive.raspberrypi.org/debian stretch/main armhf realvnc-vnc-server armhf 6.4.0.39899 [6,798 kB]
Get:2 http://archive.raspberrypi.org/debian stretch/main armhf realvnc-vnc-viewer armhf 6.19.107.39927 [2,540 kB]
Fetched 9,338 kB in 37s (248 kB/s)
Reading changelogs... Done
(Reading database ... 137257 files and directories currently installed.)
Preparing to unpack .../realvnc-vnc-server_6.4.0.39899_armhf.deb ...
Unpacking realvnc-vnc-server (6.4.0.39899) over (6.3.2.39069) ...
Preparing to unpack .../realvnc-vnc-viewer_6.19.107.39927_armhf.deb ...
Unpacking realvnc-vnc-viewer (6.19.107.39927) over (6.18.907.38355) ...
Processing triggers for mime-support (3.60) ...
Processing triggers for desktop-file-utils (0.23-1) ...
Setting up realvnc-vnc-viewer (6.19.107.39927) ...
Setting up realvnc-vnc-server (6.4.0.39899) ...
Updating /etc/pam.d/vncserver
Updating /etc/pam.conf... done
Original config file "/etc/vnc/xstartup.old" preserved
Looking for font path... /usr/share/fonts/X11/100dpi/:unscaled,/usr/share/fonts/X11/100dpi,built-ins (from xset).
Original config file "/etc/vnc/config.old" preserved
Installed systemd unit for VNC Server in Service Mode daemon
Start or stop the service with:
  systemctl (start|stop) vncserver-x11-serviced.service
Mark or unmark the service to be started at boot time with:
```

Figure 9: Compilation of VNC Step 2

Typing this command will install the real VNC, server on our raspberry pi device.

In the following next step, we will then enable the VNC control server. Type the current following command in order to enable VNC control server on raspberry pi device. “. sudo-raspi-config.”

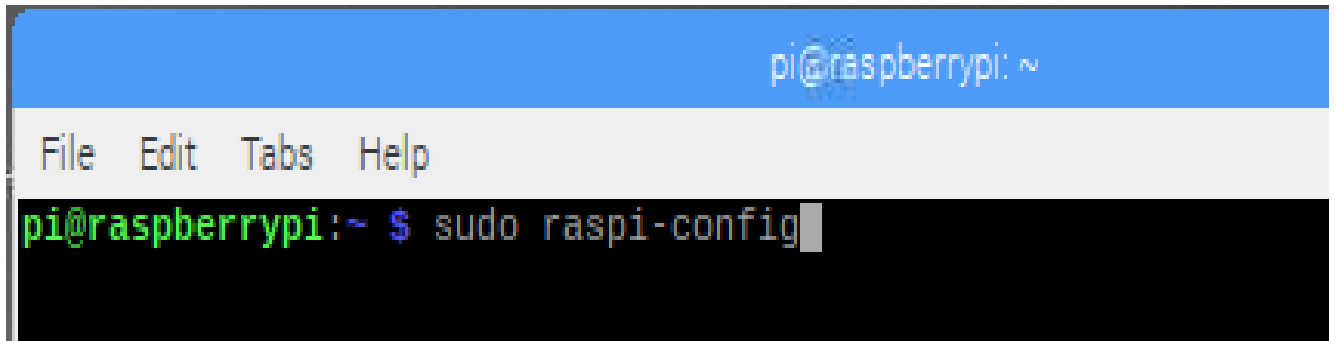


Figure 10: Compilation of VNC Step 3

After the VNC server has been enabled, press Enter, and windows shown below will appear.

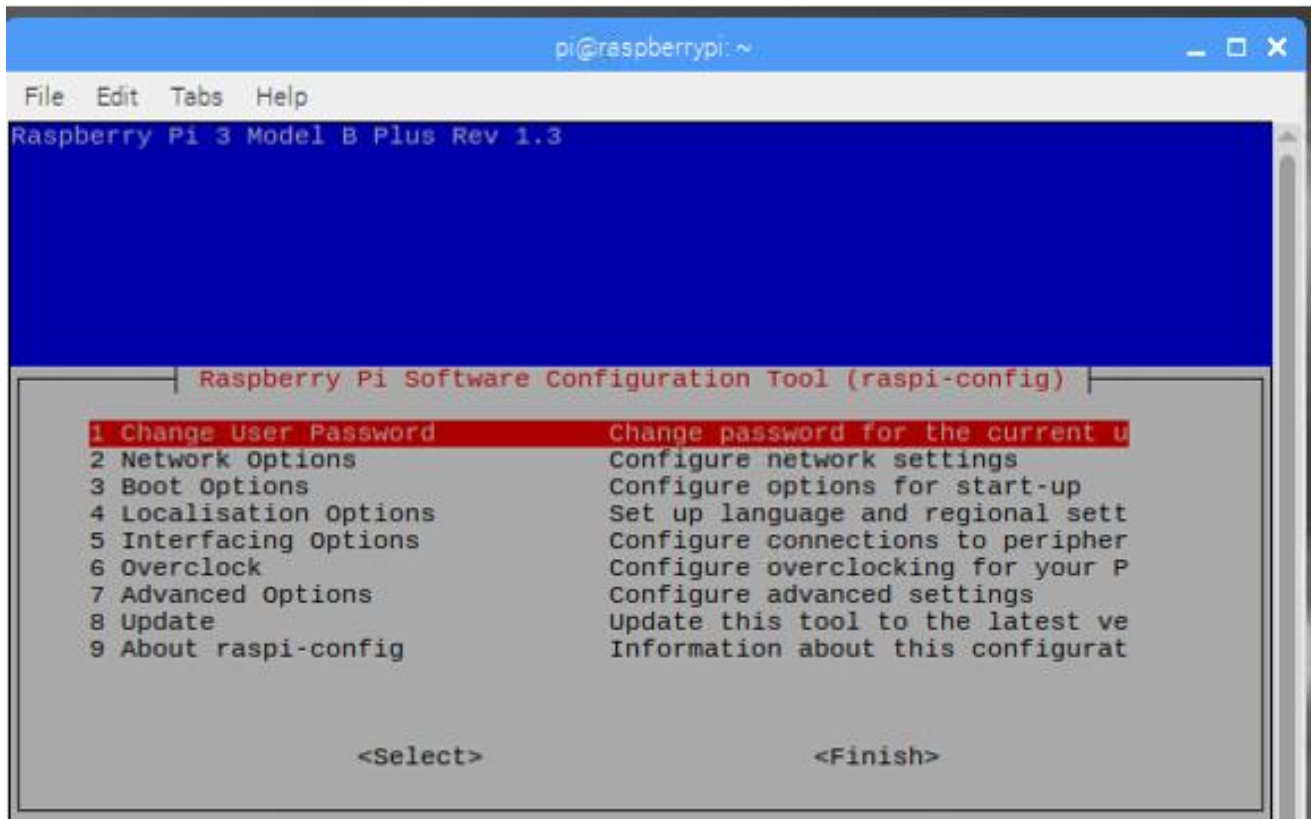


Figure 11: Compilation of VNC Step 4

Now scroll down until the "Interfacing Options" is highlighted and then press Enter

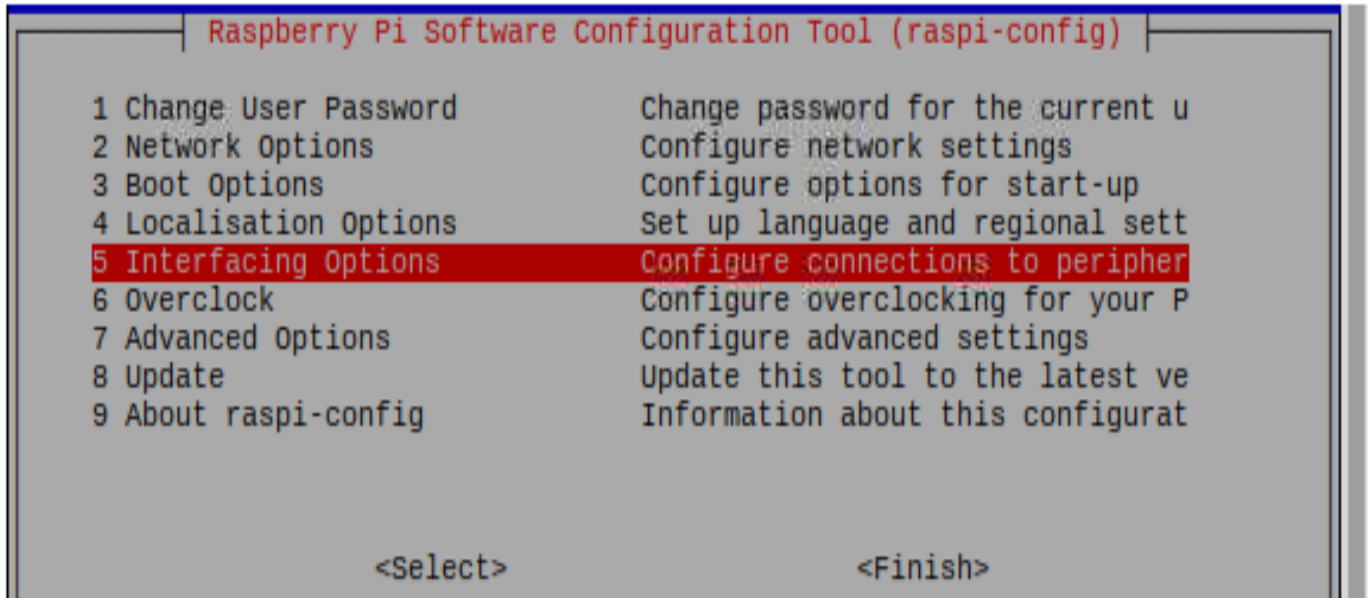


Figure 12: Compilation of VNC Step 5

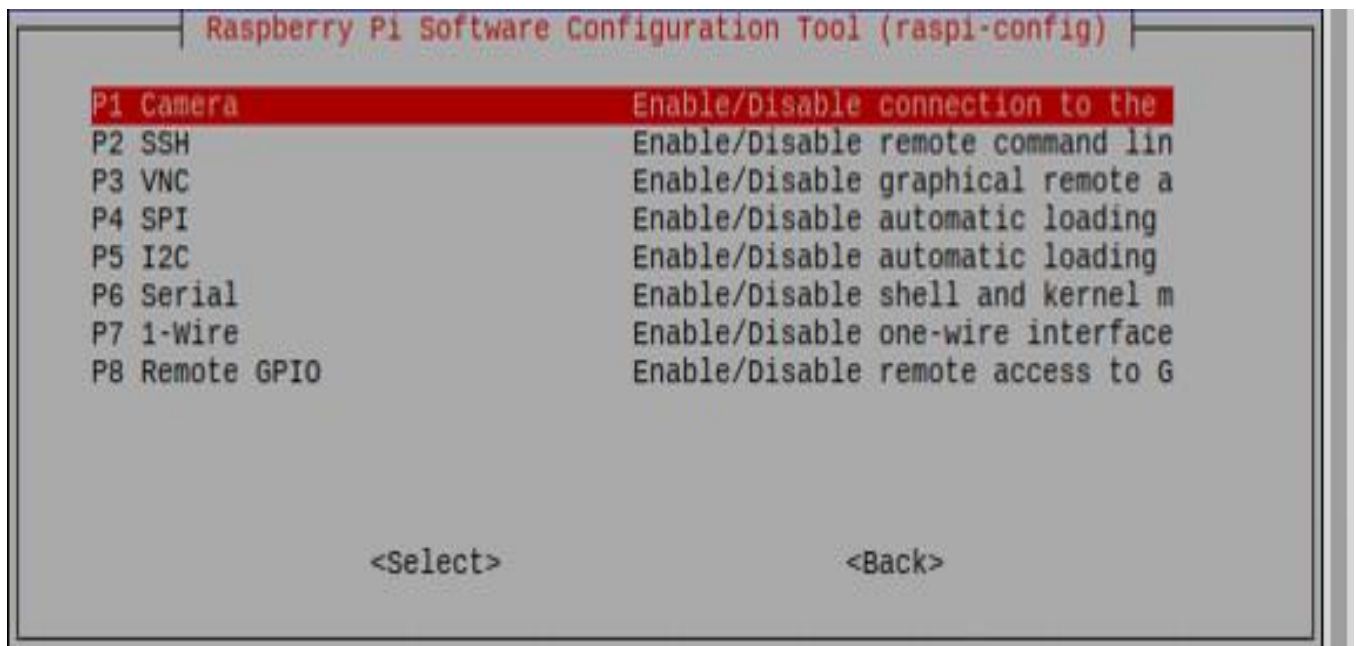


Figure 13: Compilation of VNC Step 6

Select "P3 VNC" as your option and press Enter.

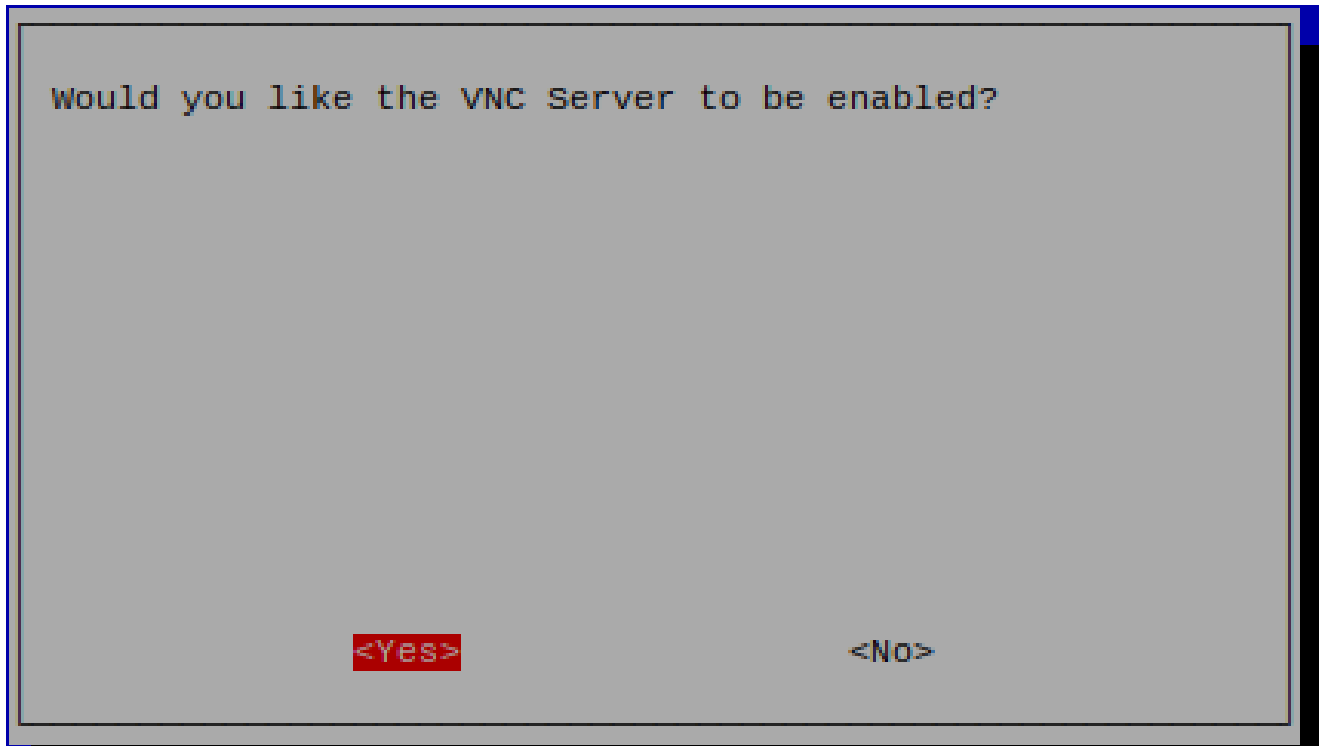


Figure 14: Compilation of VNC Step 7

Now Enter "Yes" to see if the server is enabled.

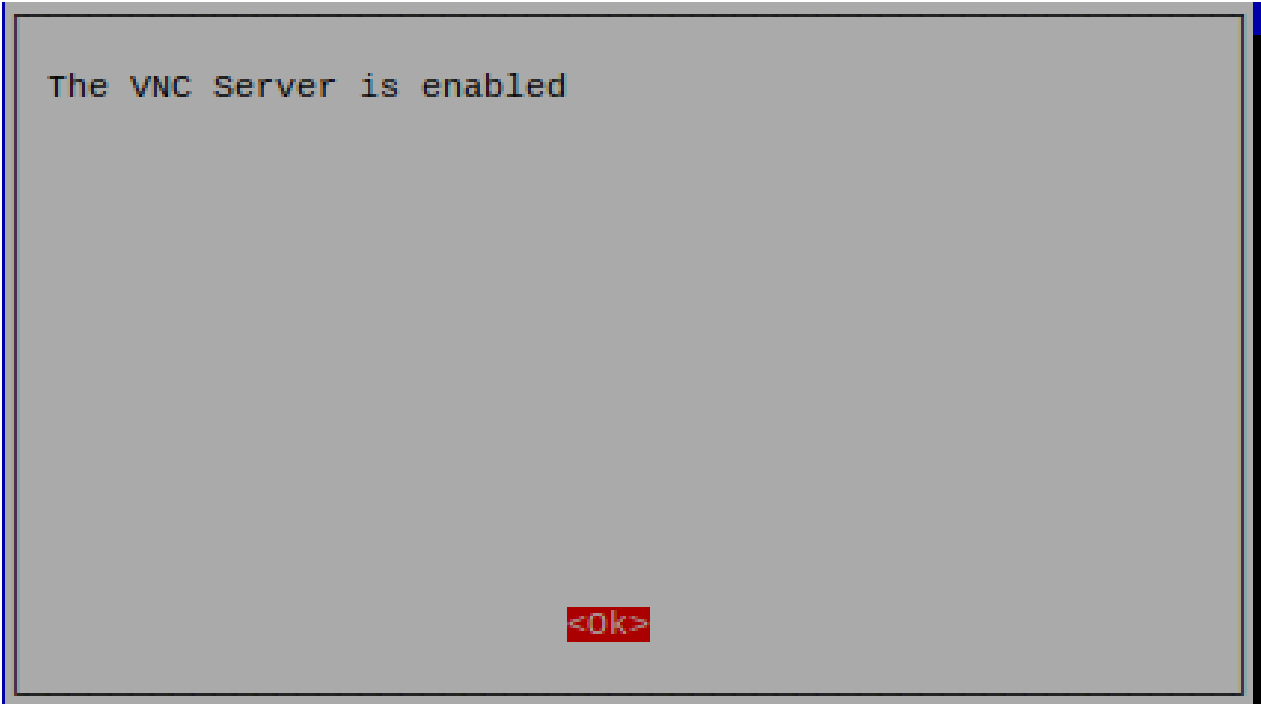


Figure 15: Compilation of VNC Step 8

Press "OK" to proceed
After pressing "OK" the initial window will appear again.

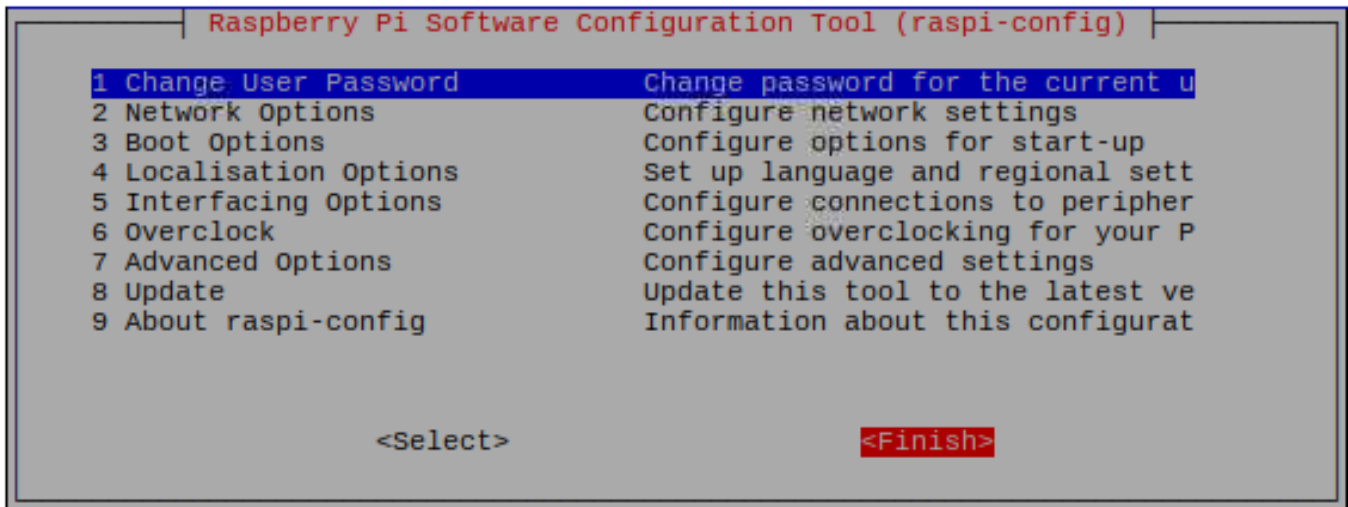


Figure 16: Compilation of VNC Step 9

Now press tab to scroll down to "Finish" and press "Enter."

The VNC Server is now completely enabled.

2.2 Libraries Used in Project

2.2.1 NumPy

NumPy is a computer programming library which is used in the python programming language, it can also be used in C language or C++ language. Its full name is numerical python. Its main purpose is to be working with programming arrays, its other function includes mathematical matrices, equations of Fourier transform, complex problems of mathematical algebra, statistical data or computer simulations etc [10].

What is the reason that we use NumPy library in python, because the list of arrays in python take a up a lot of time to process the array data. This library was thus created to solve that problem. The Object used in the language NumPy gives many supportive computer functions, this array object is known as ndarray. This makes working with NumPy very easy.

2.2.2 Python Hashlib

The hashlib library module is used for creating a secure hash message in the python programming language. This library function of python language takes the rows of unnecessary data bytes and transforms them into a securely fixed arrangement of data [11]. This function does have one major disadvantage that is it known as a one-way library function which means that if you convert a fixed arrangement of rows then you cannot go back to the original hash data, i.e. original sequence cannot be generated from hash value.

If description is not possible of the hash message to its original form, then we can say that the algorithm which we used is for more superior that the message in its original form. A significant change can occur in the messages digest value if only one byte of the original message would change.

This python hashlib library is used for security purposes in such way that the passwords are kept in encrypted form by it. Even the owner does not have the authority to access the encrypted files. Access is only given when the user enters the original password, then hash value for it is calculated, this value is then equated with the hash value already stored in the memory. Only then access is granted to the user.

2.2.3 Sockets

Sockets are used to set up two-way communication among some different processes or different types of electric devices/machines. To put it in simple works sockets can be used to talk to additional computers present on the network via a Unix descriptor. In Unix every input output task by reading a file descriptor [12]. Sockets communication between two unique processes on the same device or different devices on the same process.

Different types of stations such as Unix internet domain sockets, TCP and so on can be implemented by the socket's library. The socket library provides a general interface to manage every process. The function used to create a socket is called 'socket'. It accepts family, type, and proto arguments. To create a TCP-socket, "socket.AF_INET" or socket should be used, "AF_INET6" for family while "socket.SOCK_STREAM" for type.

2.3 Hardware

The types of hardware used are described below

2.3.1 Catching static of TV using the camera

In today's modern age of technological advancement random numbers have a vital role in it. Random numbers are used in many applications such as games, communication, network security, cryptography and mathematical sampling off numbers. Now in order to generate true random values our generator needs such a source from nature which is completely arbitrary, this is one big challenge for us because these sources of entropy are complicated to use and can be very costly as well. In order to bring true random number creators available for common usage it is vital for us that we find a viable and economic source for the generation of true random numbers. This method that we have discussed in the paper address both issues of price and feasibility. Where, what we do is that we take random data valued from image sources, to be used in the process of true random number generation. Another thing is the use of a hash function in our generator to be used as data number extracting tool. According to some research done it is proved that the hash function used will only be truly random when the input in the device comes from arbitrary random noise source, if not then the result will be deterministic random numbers and not truly random [13]. For overcoming this issue, we used some mathematical data blocking techniques that will transform a normal image data taken from a source into something that will make truly random digits of data.

What we are doing in this project is that we are taking static from a TV source that is receiving the static via an antenna. As this static is taken from a random environment, the entropy is very high, and the chance of repetitiveness is very low; this is a very good example of how to achieve high entropy.

Noise in an old dish TV set is a very arbitrary array of white and black points, arrangement of the static that is displayed on the tv screen. This happens when the antenna that is connected to the TV is not receiving any signal. The random pattern is visible as random flickers of points of black and white being displayed on the screen, or it may seem like snow. This is not actually snow or random flickers, but this is an effect of all the electronic clatter noise and the radiated electromagnetic cosmic radiations present in the sky, which is picked up by the antenna probe, this happens when no signal is being received by it. This result is usually seen in many analog television sets but is not very popular with the digital cable, which is used nowadays.

There is a lot signal of electromagnetic nature which are present in the sky that causes white and black static to be displayed on our screens. The most widely available source are the electromagnetic natural signals which are present all around made by the cosmic microwave radiation in the sky or the radio waves which are traveling, propagating from the electronically powered machines that people use.

Some devices by themselves can be a source of natural entropy which is by something that occurs in the microscopic level which is the noise made by the movement of electrons due to current flow, this is a naturally occurring phenomenon.

Some TV set do not show static rather they show a standby sign when no signal is present this is due to the reason that negative video modulation is used rather than positive video modulation but that was in the old times now majority of the people in the world do not use cable they use digital or satellite tv in their homes.

Normally, a television set is connected to an antenna. The antenna is picking-up a wide range of signals. Even if a television is tuned to a channel that appears to be nothing but white noise, there is a strong likelihood that adjacent signals are coming through. This will compromise the randomness of your signal [14].

So, all what is needed is replacing the antenna with a resistive terminator. This will provide a truly random noise (which will dominate all-transistor noise sources on the television). Finally, because we are only interested in a binary sequence of random numbers, we need to digitize the data. This means we need three things: a resistor, a high-gain amplifier, and a high-speed ADC. Fortunately, we are in luck: buy a TV tuner for our computer, rip-off the antenna, and stick-in a matching resistive load. Random numbers are, by nature, unpredictable. If the pattern is completely random, then yes, it could be used to make random array of numbers which will be non-deterministic. For example, you could take the first 8 pixels and assign a 1 to a bit if it's white or a 0 if it's black to get a random byte. Atmospheric noise tends to be the method of choice in most cases, though.

Now we know that the white, black noise on the TV is completely random, then the answer is a clear yes say choose a pixel on the screen and use each sequential n-length series of black and white spots as binary digits in a series of random n length numbers.

Since it is proof that the data which is appearing as static on the television is completely random so we can use it as a source of entropy for our random number generator, the static which is being viewed on the screen is also randomly changing every second. What we do is that we use a camera pointed at the screen on which the static is being displayed. The camera does what is supposed to do, and it captures the images. Once the images are collected, they are separated frame by frame, and all that random data is sent to Raspberry Pi for processing.

An example of these images which appear as static is shown in the figures 17 and 18.



Figure 17: Static Source 1

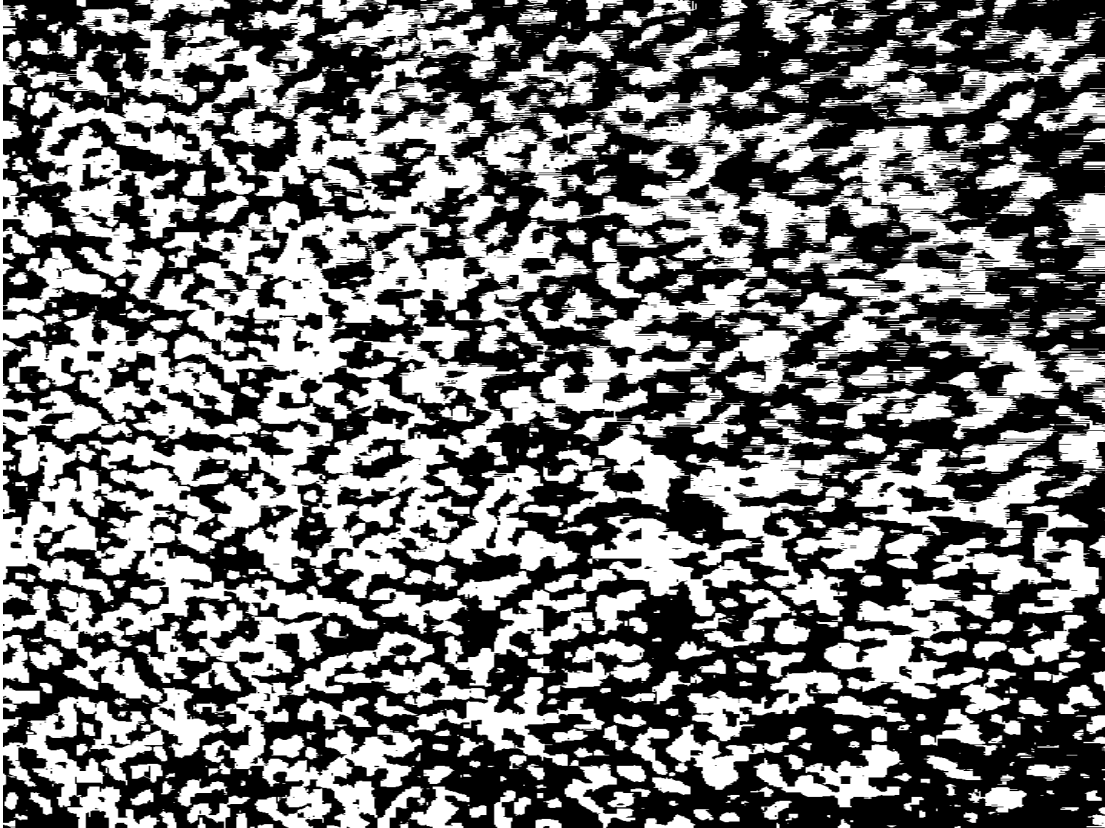


Figure 18: Static Source 2

2.3.2 Raspberry pi 4

Raspberry Pi is a small chip computer designed for the purposes of computing and teachings purposes for the developing countries. It was mainly used in robotic projects and then its applications were extended towards research project because of its small size and affordability.

The Raspberry Pi 4 devices are of most up-to-date products in Raspberry Pi companies range of computer silicon chips [15]. As compared to its old model the new up to date one gives users top end speed and memory options other than it only lacks in its power efficiency. It gives its user a very powerful desktop like presentation which can be linked to any entry level system of its time. This device gives its user a quad core processor of 64-bit architecture, it has an option of two displays hence the micro HDMI ports are used its display quality can also go up to ultra HD, it provides RAM memory size of 8 gigabytes. It also features more upgrades such as Bluetooth option, Ethernet port for connecting to internet cables as well as a WIFI option too for connecting to wireless networks. Other than all this device also includes a power over ethernet connection

(POE), it is a new networking feature which gives the device an ability to draw power via the ethernet cable which is providing the internet connection.

An example of the device is shown below.

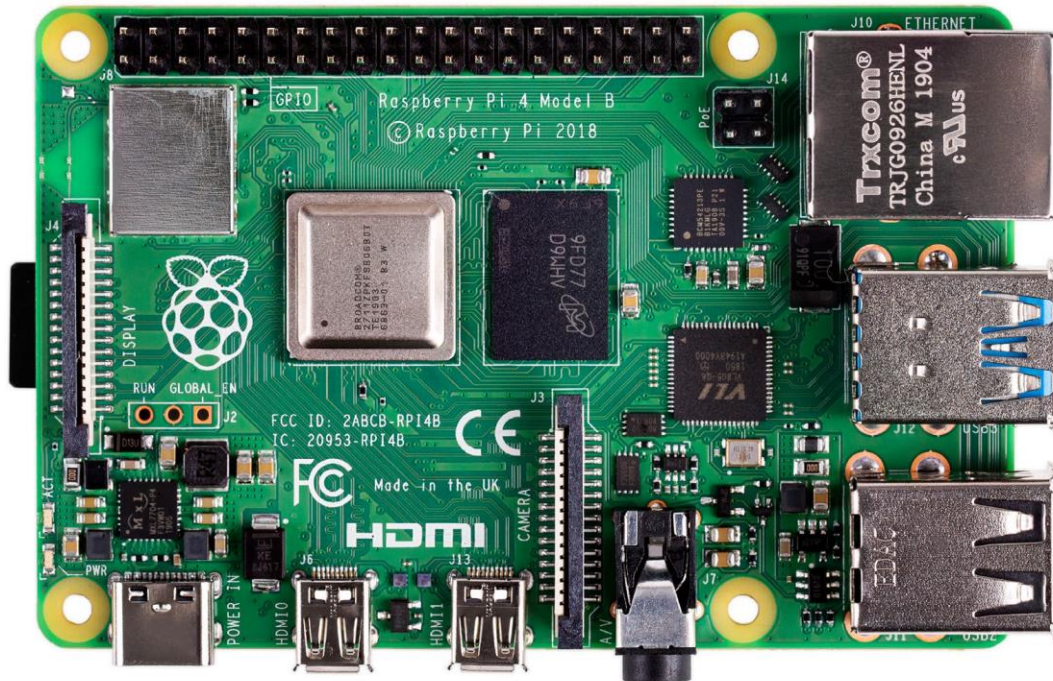


Figure 19: Raspberry Pi 4

Specifications

A detailed list of all the device specifications is shown below

- Processor: Broadcom, quad-core Cortex-A72.
- Memory: It offers memory in order of 2,4 or 8 gigabytes.
- Latest Internet and Bluetooth options.
- Ethernet port.
- USB port of 3. Speed.
- USB port of 2. Speed.
- General-purpose input and output (GPIO).
- A MIPI DSI mobile industry processor interface display serial interface port.
- A MIPI CSI mobile industry processor interface camera serial interface port.
- stereo audio and a video connection port.

Physical Specifications

The devices physical specifications are shown in the figure below

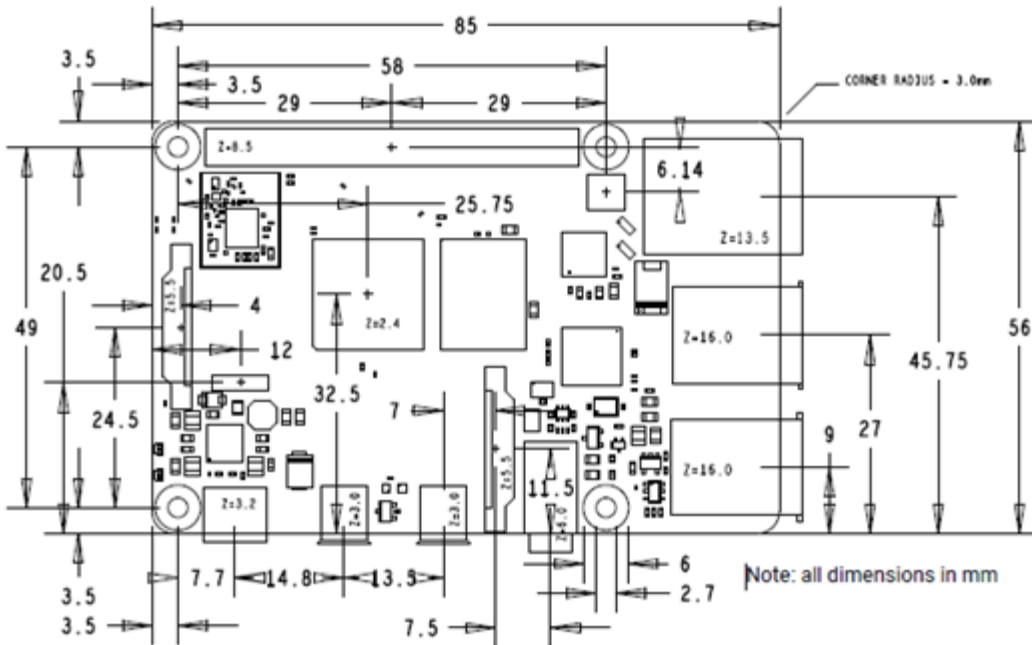


Figure 20: Physical Specifications

Software:

The Raspberry Pi company gives Raspberry Pi OS to be the certified operating OS system for all their Raspberry Pi devices. The company says that the programming language of Scratch or Python are the official certified languages of their devices. It is not limited to only the official operating systems it can also run other popular OS systems as well.

Accessories Some common accessories include the following

The Raspberry Pi accessories include:

- HAT (Hardware Attached into)
- Camera
- Infrared Camera

Chapter 3: Proposed Method & Results

3.1 Proposed Method

3.2 Results

CHAPTER 3: PROPOSED METHOD AND RESULTS

3.1 Proposed Method

To understand this project and its workings a flow chart diagram has been shown below. It describes the flow of data of our project and makes it simpler for people to understand why each component is required and what is its use.

Flow of Data in the Project

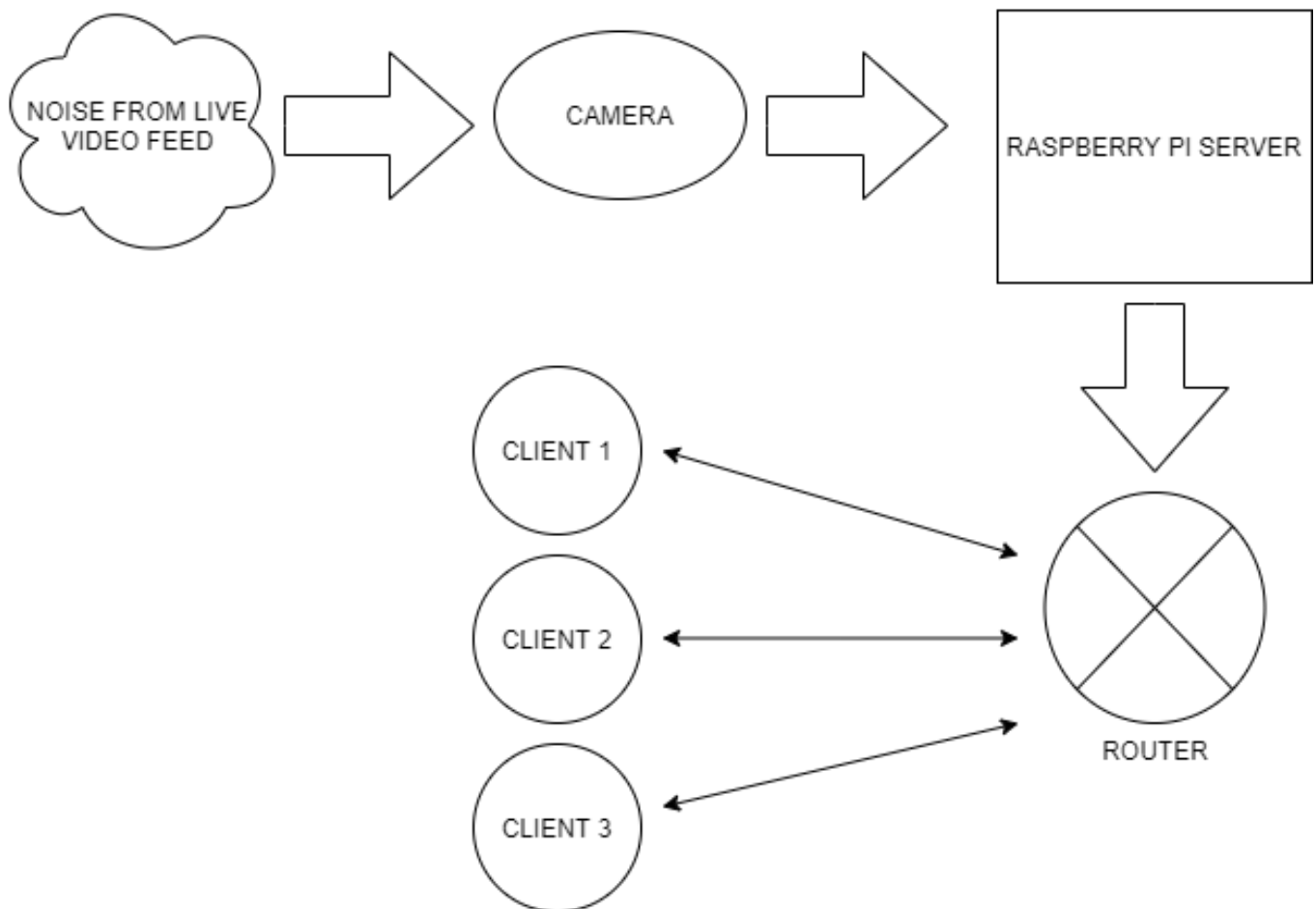


Figure 21: Overall Data Flow

The overall data flow in this project starts from the noise that is present in our surroundings in the form of cosmic radiation noise. Along with the cosmic radiation noise there is also the noise that comes from random electronic interferences from surroundings example from the electronic devices, signals from the nearby cellular tower's signals from the satellites and many more. Since there is no specific analog signal to display some meaningful video on the television so the antenna automatically picks up all this electromagnetic noise from the surroundings from the environment.

When the noise signal is picked up by the antenna it travels through the antenna in the form of an electric signal. This electric signal further faces some distortion in noise. Moreover, incoming electric signal coming from the antenna probe to the television then further faces some more distortion in noise. This distortion is due to the reason of electrons in the electronics present in the television. These electronics heat up when the television is working, hence emitting thermal noise. This thermal noise combines with the electric noise and the whole signal changes again. This phenomenon is described in the picture below.

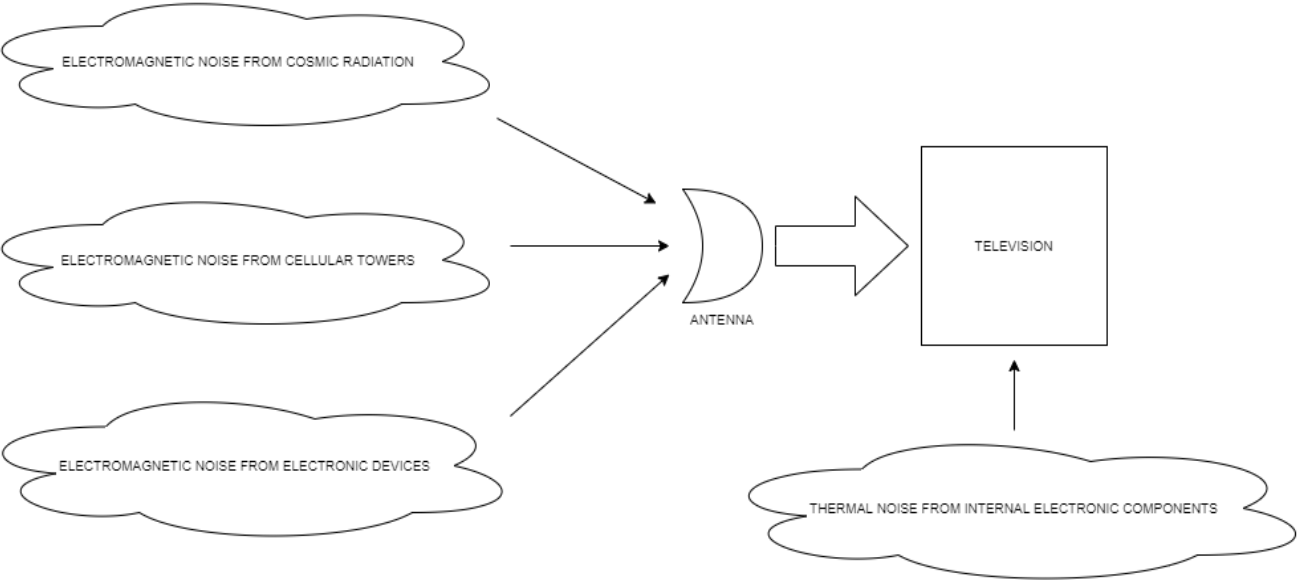


Figure 22: Noise Sources for a Static

This signal is then shown on the television screen in the form of a static. The static shown on the television screen are like a video of white and black dots. This video is the picked up by the camera that is pointed towards the television screen.

The video is then sent from the camera to the raspberry pi server. The server receives this data through the library used which is open CV. The open CV library further needs help from a python library called NumPy, which is used for making mathematical calculations. The video is read by the open CV in form of matrices of each frame of the video.

Each frame has pixels and every pixel is then further resolved into matrices depending upon the color related to that pixel.

To read the frame and get valuable data out of it we need to convert this frame into a binary frame. This process is described in the figure below.

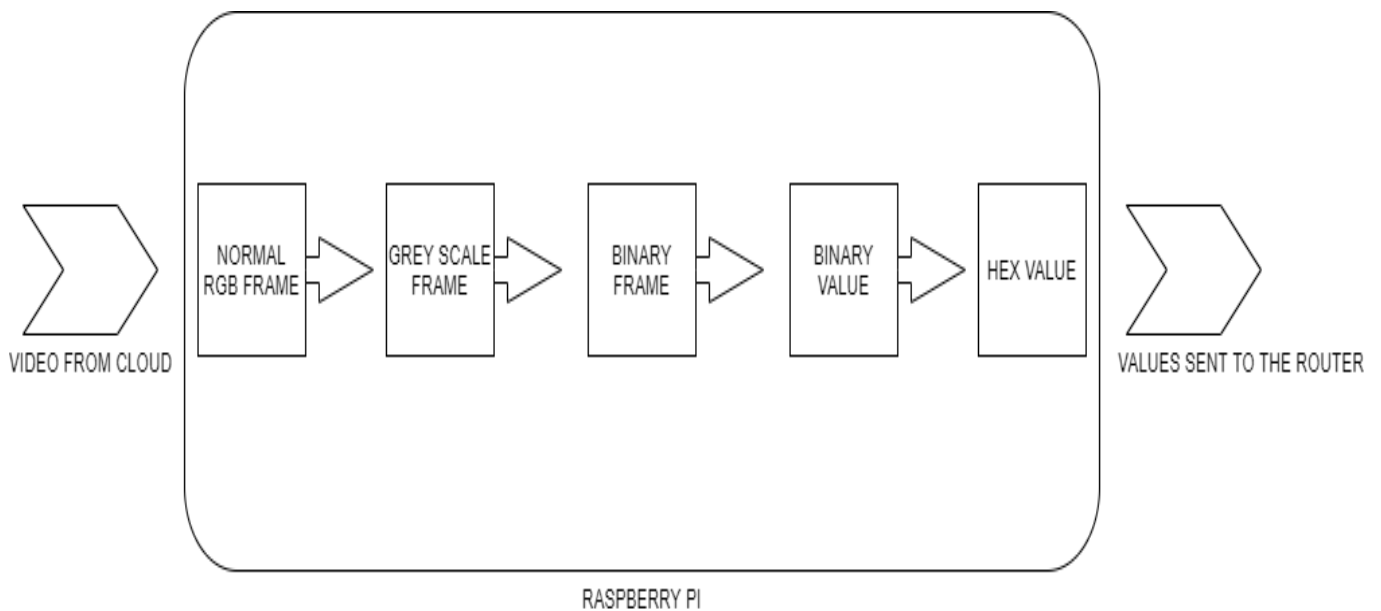


Figure 23: Operations Inside Raspberry Pi

Each frame is hence first taken as an RGB frame, the frame is then converted into a grey scale frame. This change in the image is achieved by a given method “(cv2.COLOR_BGR2GRAY)” of the open cv library. Due this method the RGB data picture is transformed to grey scale rendering of its color differences in the pixel. This grey scale picture is then transformed in a binary type

image by adjusting a threshold. The threshold we used was 127,255, hence any pixel above 127 color according to open cv is converted to white and vice versa. After thresholding the image, we have some valuable data that we can use for generating our random numbers.

We start from every frame, depending on the number of frames the client has asked to be used for generating the random numbers. For simplicity let us say two frames were asked to be used for generating random numbers. Every frame is read from top right pixel and then all the pixels are read moving in the left direction. The pixels are read in such a way that if the value of pixel is zero in binary form then a zero is registered in the memory and if there is a one in the binary image then a one is registered in the memory. After reading the whole frame a value in binary form is achieved. The next frame will then be converted into a grey scale picture that is then transformed into a binary type picture. Its values are read and stored in the memory.

All the frames required are read and their values are stored separately, the values of all the frames are concatenated and then converted into decimal. These decimal values are then converted into hex by using a library called hashlib. The hashlib library has special methods for making hashes. The hash we are going to use is the sha-256 hash. The decimal values are fed into the hashlib method and are then converted into a hash of hex value. This hash value is of 256 bits and is then sent to the router via a safe TCP connection established between router and the raspberry pi. On receiving the values from the raspberry pi, the router sends all the values to the client that asked for these values via a secure TCP connection.

3.2 Results obtained



Figure 24: Original Image

To experiment the process of generation of random numbers, a frame from the TV-static having the resolution of 640x480p was captured and analyzed.

First, the static screen having black and white dots was converted to Grayscale by using OpenCV library.

After converting the image to grayscale, it is then converted into a binary image which is clearly visualization of pure black and white colored dots.

This again is done by using OpenCV in such a way that it turns black dots into pure black and white dots into pure white by setting up a limit. The Black dots in the frame represent 0s (zeros)

and the White dots represent 1s (ones). Then the frame is read from top-right to leftwards and the frame is stored in the binary form.

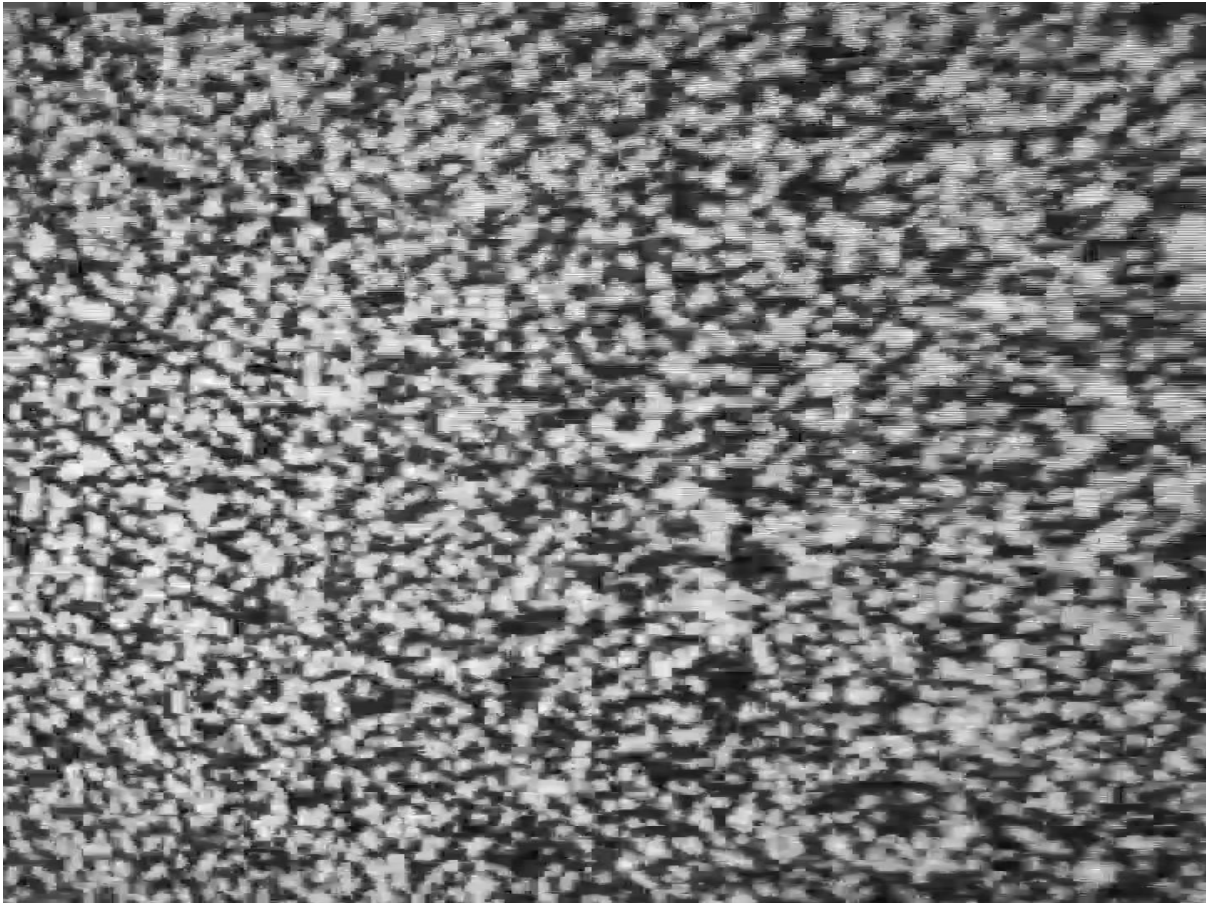


Figure 25: Grayscale Image

This was the case of representing a single frame in binary form. If the client desires of getting more than one frames; for example the client has asked for random binary of two frames then first the binary of one frame is determined by the above mentioned method and then the same process is carried on the other frame and the binary values obtained from both the frames are joined as binary of first frame followed by that of the other one.

Now if the client wants the results in decimal forms, the binary results obtained first are converted into decimal results.

If the desired form of the obtained numbers is Hexadecimal, then the decimal results are converted using hash function SHA-256 and the hexadecimal random numbers are generated.

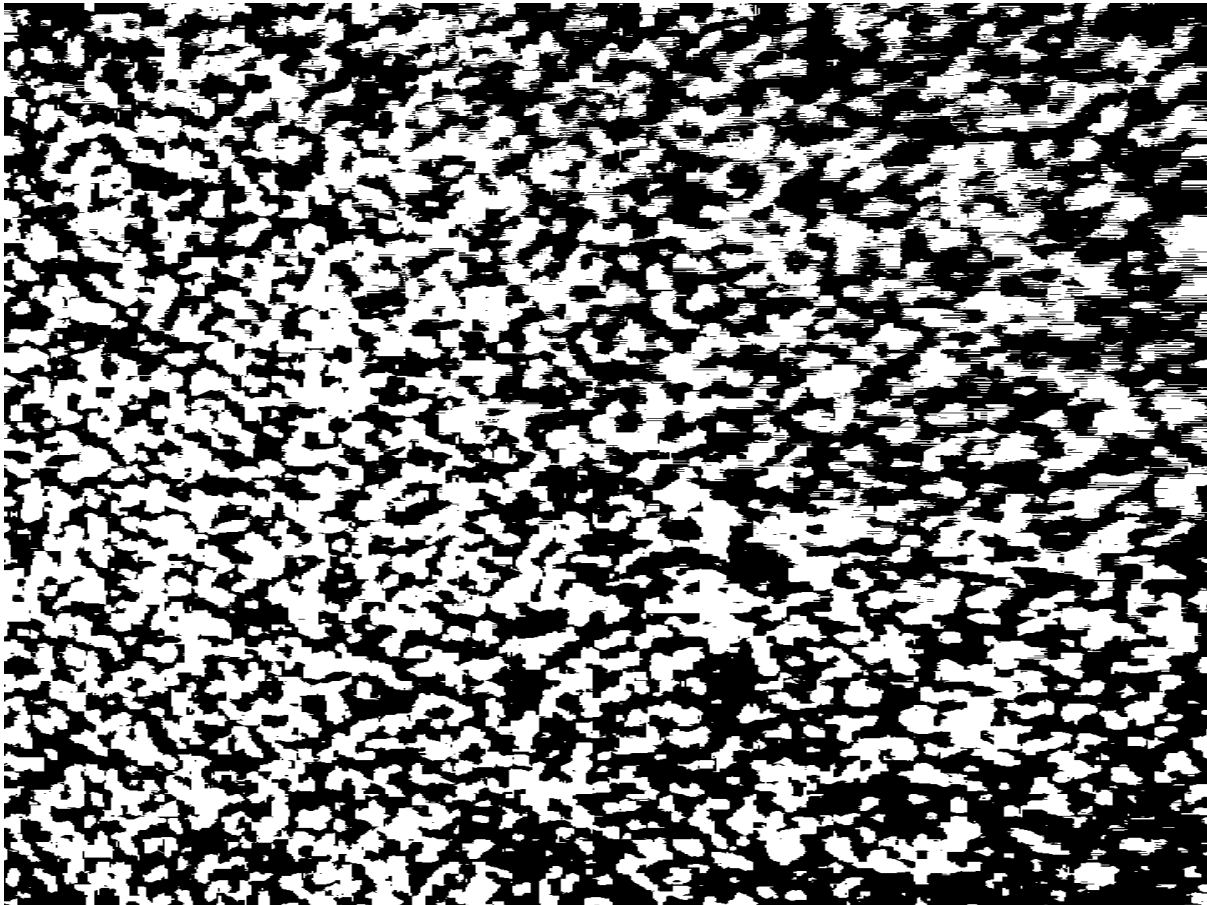


Figure 26: Binary Image

The generated output in the binary form as obtained by the single frame that is shown in the figures above is given in the next page. As one can see all the pixels from the frame are converted into 0 (zero) and 1 (ones). The decimal output if so, required by the client is given in the figure below. It shows how the values in the binary form are then converted into values in decimal format. The decimal output shown in the figure 27 below is the output for the binary random numbers.


```
613373075932821936341423679736707431324157929212470265135264723947813584095095297819688310481573336174940485852939032937943
442599005281905963775552934698132453102655275382651044656062695089060283192002565720118591825464212091517230361459190946286
586076353858541073803146335534761428504670838714930129896587633089548252789236077053061002440096003145521164744122659942455
347860984420523436958535626002559999314108908821255305984187856324496704004978043816140156479225235734170402364058725783625
773872996141265086879417816572006801632662116416916653795886360713319031906439267202097146546657698269455562997975066276525
007490848175000899894220215325027795595969223461470894290740949270614882151853760762550640573906999876122089829269919910624
112519308546089567137562689693174278694998444328807407661506264638594627894456171158286562261081223481368890376490651402369
294475944206434431409130918510752175274661281764064788084540653251679428022406544801348406204887474427985561539457952203626
595788162324820499136878323032515912193768305230238925713041379561747988748906042345209880762489915636618969620840216766778
102650695013887003823461903677083034671045075144536173372449828916015498597119803781985137082659902688985411340734116780031
202211442682327895913553793235207289853886859148760510094164276784276892557653583363414181912705174848907209931051287857924
446649215920455591541525125521833970586410994001638057770908713114567112276013680147971116707783810459249663627789199146391
973461199003702866830642573889991425682098737753398739662617249981303723009087229225023263867047147325056941288418064843869
784045763345854918383714618609491724442486609314855267325334609435884634397500200402804999014420643835130311413980121016123
655370046737945557696104331821725603345673582613182770783978531691199466521721458289782215091993507747983495124775052024053
295340933203328050505542660366841849940438418278557918678909350509052753925884655929556609447826526807899678404740457063666
482218803242680920385120637508797457855287214656315729836446752971801546055040296155755655362603117214224544785563400828034
173818582144649067892335730857951733490039674886066561296701992283116800955789988620412289443334951509341608041329650154983
402362288383218288574117782278874295594202088089225099779927178999488632870083131049165191237034429032372353953961979554874
607635409360368691794225330301543926697159392731900266421787792360881956672707501237644252907062082013044210641885631703355
977938214461451201905628913518781459373193893977065195772129804526781346402169471965739643419676176704105184348775483124910
289692065790534821133403330890146632519128939790069925458919842714094508388389746156973662586679796808892174243585694157350
335294656796528925784897726471397868301138536371835197186821646896822584949835619135207519838279979776601695452354331997167
3565134814050479253037091048554856858061617409736973984888014558737639987370601135286379803721006313708771035105113650105
```

Figure 27: Decimal Output

The hexadecimal output if so, required by the client is shown in the figure below.

This output is generated from the hashes that are generated when the above output in the form of decimal is fed into the sha-256 method of the hashlib library. All the hashes for groups of values from the decimal output are concatenated to form on complete hash value. The hexadecimal value thus shown in the figure 28 below is for the same output given above in the form of decimal output.

```
b9be277fb6ce30b6161cb432190b0141810b900424f30202a3442a0efdae9665b9be277fb6ce30b6161cb432190b0141810b900424f30202a3442a0efda
e4abc2eab4e994f099e7e20f65c8129ebc98071a6bf50e288e4e1ed534d48440cdf7bbc3bc16794db5df805e5223cdd5e78b5b4c62e7a597e6db65a357f
ac20dee4d60e238dce28e4427a0b39aaa1983c595607656f9e6713bbfb17f6aec551af4eca600368ca2bd392f205e1d804e63f79f3edc2fa259c0be240
bf9ca4673cedef00b78c7f0ccf67c68c4f9aa8f3d5cc87905dcb47ee4687df47a6fcf0100336b5399b2dfa33700f1d0e9a9425b0603e471e0fe6ceacd4e
3e69f53f6673c3272ddca33ee1753b60e9a4e2b307123de5fdb7c1fc917e54dc49f1184e0689213564e87702369e21d0f2bf37b9bc4e722e7b16262a038
44d332860549607c9fabfddcf9fc6ed324c1692573e049989e80ddc4d365f61e399dc69df85bd74680fb7b35ca44714347df54a299e3283f3ffc88637a9
1deb294277f058f615d00144da2751561508ae547c67b037ea0dba408745dec8a609d77a177cb9761ee864ae138a135224523fbef95a7a3da321e6129f
eaa79efc75d3faec060981d3edf6e55f6f7ba80a097658ec4b161ea931dca0906fb7c68251c4ebe89ae1f31775bef66dd1184abb48c9171262ef1464ce63
9a5258b1707aedad005a6374069220f1c4df48fd990fa2169590590787aab17254a4a3ae8c32ee54e8fdb951c14b7b28759098aae4357ec6ba77a17fdac
059f8b83d302455d70e3d03c502e39623d8b0b9768865569d893c5b49534347ae45df2e9ebc358b4ddbee010fd53b16d22d764e97643072720368db3f0
35cc71b7ed9c385fec5661485549ff3a1fc2e620864278e019ed8f9aa4cd20adff91c11c8d119bf2748692a3b89422a25d80194fab3173cadd9e281d908
1dd63e121fd8ac91757e95e97265c7aa7990d72175eaf0366eac0dcfffffd7c0a4792ad39569e0c3e884614e6508b276b1485c5191614fcdae926d3ab178
31842980bea555d3e93e1cab21e2eaeac40ff0492d7e871abef9ceb29615f95a71c4576336a3d56da64e5feb7ff66b927240953df5a868e3f4a03e9276ed
fa0e9ab8c752704b3766d6472a83905aef02e7d221ac043532790c0a33f08717b1ab46662fd36d376f2247e11e0ac0fa31ce04542ead201d9d8b9aee96
4546f78cf1174f67d0c30efed176786098be0b58a4f51ee40b377782a19870c66acee4af03a1795597160ed05f67ba8f645a255d7e62f958a05cf52a1d3
75501e84042d81263a51be205d4bad5a05de052e3a22a1fbc3322ed3ca6ec321efdb5632abee8b618276b2161740e50f948d9e104fea446c03d30ac07c6
1cb25897259b4a15fde1389506aa4ef76af152621e7c668c50cf5c2d250b79bfd7c226b8146ed8a8e0b55c0e436733aa162743a2ddbc733e9f470843a40
899cd4f360d9e24adbd537d6c366cb78e9390dce360bac9d2a35fa29fa6905151d89dbe5087a02ba25f801a7e035173ff907ae68613fd2443445c3c674c
eb3646ecba659be17f61222274d8bd32a97c17378db9e8742df0e534dbec80af74abd96f0e5c12aeaa0b30d4e8226dd4318c6306b27209e1e4638c88b19
53cf3c54d1b87cacd0d30553eedac52dda991fe020120a3b89271beafef58e5736991fa77bb73d67d9d09c91e55717adf8e314ae9a8b39f4aaa22c668dd
813ccced1e68782df0b8fdb5d628b4e3eec987187b11a4f2626801e4728bc315312665c40637ef8b0ec1a62b3824013b7457cbba173c2151b645f4ee26a
fbfce1e6b093469224bde3c49b5db3823ab9ec761c22b37aa5b330f9209d4ce8d1cf162f664ad742a24e6a80c3437b902311f8db6aa87de8e1f54b41b4c
42ca244b2a696a82cb28286d91cab1afe3816cf577473f7096bf5be3f21dea51d599bea7c3f8e96d621ad5fd4db349b131172da8e69c305d089c90224dc
```

Figure 28: Hexadecimal Output

TCP (Transmission Control Protocol) connection:

A connection between a TCP client and server is established in mainly three steps. First, the client which has a desire to establish a connection with the main control server, sends a SYN request to the server. The control server then replies with a acknowledgement (ACK) + the synchronized (SYN) data packets [16]. That SYN is then acknowledged back by the client and a connection is established between them.

Transmission of Random Numbers:

In the project under discussion, if the client requests for random number generation to the server, it cannot do so directly. The client asks the router for the desired format of random number using the TCP connection. The router then requests for formation of a TCP network connection with the control server. Once the network server client connection is made, the router forwards the request of the client to the server and the server sends the required data to the router. The router then again establishes connection with the client and sends it the data it asked for.

Chapter4: Conclusion & References

4.1 Conclusion

4.2 References

CHAPTER 4: CONCLUSIONS AND REFERENCES

4.1 Conclusions

So, to conclude we can say that the use of the static displayed on old analog television sets is a great way of collecting random noise. This noise is a good source of randomness because the static itself is composed of many different forms of noise mostly electromagnetic and thermal.

The main noise being Electromagnetic noise from the cosmic radiations which are as we know the most random source of noise in existence. The other sources of noise being the electromagnetic noise from the electronic devices around the television, random electromagnetic waves propagating through the television from many other different sources and then lastly the thermal noise from the electronic equipment that is present inside the television itself.

All these noises combine and give rise to a new noise that is then displayed in the form of static of the television. So using a simple television, a camera, a raspberry pi and a router we can essentially make a very cheap and easy to use hardware based random number generator.

4.2 References

- [1] "<https://www.iro.umontreal.ca/~lecuyer/myftp/slides/wsc17rng-history-talk.pdf>," [Online].
- [2] "<https://www.thisismoney.co.uk/money/saving/article-6760741/New-generation-ERNIE-machine-draws-winning-numbers-12-minutes.html>," [Online].
- [3] "'RANDOM.ORG - True Random Number Service". www.random.org. Retrieved 2016-01-14.," [Online].
- [4] "<https://mswsrng.wixsite.com/rand>," [Online].
- [5] "<https://www.quora.com/What-are-the-practical-applications-of-random-number-generators>," [Online].
- [6] M. Tyldum, Director, *The Imitation Game*. [Film]. usa: FilmNation Entertainment, 2014.
- [7] "<https://www.worldcat.org/title/political-potential-of-sortition-a-study-of-the-random-selection-of-citizens-for-public-office/oclc/213307148>," [Online].
- [8] "[https://www.raspberrypi.org/downloads/noobs/.](https://www.raspberrypi.org/downloads/noobs/)," [Online].
- [9] "[https://www.sdcard.org/downloads/formatter/.](https://www.sdcard.org/downloads/formatter/)," [Online].
- [10] "<https://numpy.org/doc/stable/user/whatisnumpy.html>," [Online].
- [11] "<https://docs.python.org/3/library/hashlib.html>," [Online].
- [12] "https://www.tutorialspoint.com/unix_sockets/what_is_socket.htm," [Online].
- [13] "https://www.nasa.gov/vision/universe/starsgalaxies/cobe_background.html," [Online].
- [14] "https://www.researchgate.net/publication/220612093_Random-bit_sequence_generation_from_image_data," [Online].

[15] "<https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/>," [Online].

[16] "<https://www.geeksforgeeks.org/tcp-connection-establishment/>," [Online].

[17] "<https://www.quora.com/What-are-the-practical-applications-of-random-number-generators>," [Online].

[18] M. Tyldum, Director, *The Imitation Game*. [Film]. usa: FilmNation Entertainment, 2014.

HRNG

ORIGINALITY REPORT

7 %	4 %	1 %	7 %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	3 %
2	www.swedishryk.edu.pk Internet Source	1 %
3	steelkiwi.com Internet Source	<1 %
4	Submitted to University of South Australia Student Paper	<1 %
5	Submitted to Nottingham Trent University Student Paper	<1 %
6	Submitted to University Tun Hussein Onn Malaysia Student Paper	<1 %
7	www.diva-portal.org Internet Source	<1 %
8	Jeremy Holleman, Brian Otis, Seth Bridges, Ania Mitros, Chris Diorio. "A 2.92μW Hardware Random Number Generator", 2006 Proceedings	<1 %

of the 32nd European Solid-State Circuits
Conference, 2006

Publication

9	Submitted to University of Portsmouth Student Paper	<1%
10	Submitted to Staffordshire University Student Paper	<1%
11	Submitted to University of Arkansas, Fayetteville Student Paper	<1%
12	Submitted to KDU College Sdn Bhd Student Paper	<1%
13	Submitted to Glyndwr University Student Paper	<1%
14	Elad Elrom. "Pro MEAN Stack Development", Springer Science and Business Media LLC, 2016 Publication	<1%
15	Submitted to Far Eastern University Student Paper	<1%
16	Submitted to University of Glasgow Student Paper	<1%
17	Submitted to University of Witwatersrand Student Paper	<1%
18	Submitted to Liverpool John Moores University Student Paper	<1%

19	Submitted to Limerick Institute of Technology Student Paper	<1 %
20	Submitted to Atilim University Student Paper	<1 %
21	Submitted to European University of Lefke Student Paper	<1 %
22	en.wikipedia.org Internet Source	<1 %
23	Submitted to University of Edinburgh Student Paper	<1 %
24	Submitted to PSB Academy (ACP eSolutions) Student Paper	<1 %

Exclude quotes Off
Exclude bibliography On

Exclude matches Off