Project Design

Pype

A Decentralised Video Calling Application

Prepared by

Jerry Raju
Raeesul Asad
Shamnas TV
Soorej J. Pothoor

March 22, 2017

Contents

1 Introduction **4**
- 1.1 Purpose . . . 4
- 1.2 Project Scope . . . 4
- 1.3 Applications . . . 4
- 1.4 Similar applications and inspiration . . . 5

2 Overall Description **6**
- 2.1 Product Perspective . . . 6
  - 2.1.1 User Interface . . . 6
  - 2.1.2 Hardware Interfaces . . . 6
  - 2.1.3 Software Interface . . . 6
  - 2.1.4 Operating Environment . . . 6
- 2.2 Design and Implementation Challenges . . . 6

3 Specific Requirements **7**
- 3.1 External Interface Requirements . . . 7
- 3.2 Functional Requirements . . . 7
  - 3.2.1 Add contact . . . 7
  - 3.2.2 Make call . . . 7
  - 3.2.3 Answer call . . . 7
  - 3.2.4 End call . . . 7
  - 3.2.5 Generate new keys . . . 7
- 3.3 Software technologies used . . . 7

4 Design **8**
- 4.1 Data structures . . . 8
  - 4.1.1 Address Book . . . 8
  - 4.1.2 Call list . . . 8
  - 4.1.3 Peerlist . . . 9
- 4.2 Network . . . 9
- 4.3 Interfaces . . . 10
  - 4.3.1 Seedserver . . . 10
  - 4.3.2 Peer . . . 10
- 4.4 Architecture . . . 10
  - 4.4.1 Overview . . . 10
  - 4.4.2 Control Flow . . . 11
  - 4.4.3 Class diagram . . . 17

5 Security 18

6 References 19

1 Introduction

1.1 Purpose

In an age where communication is of paramount importance, it is considerably worrying that such networks are controlled by private companies. New information shows proof of and unprecedented amount of spying on personal communications. It is in this context that the idea of a completely decentralised mode of video conferencing supported by a peer-to-peer network, whose infrastructure is supported entirely by its users, and is secured using public key cryptography, is essential.

Pype is a decentralised video calling platform supported by a peer to peer network much like a torrent network. Node machines run a software that connects the node to the network. Pype uses asymmetric key cryptographic techniques to provide anonymity and security. A decentralised network is resilient and harder to take down making it more secure than a centralised client server based model.

1.2 Project Scope

In the wake of recent NSA leaks and global terrorism, user privacy is quickly disappearing. Pype is a platform that enables its users to communicate with each other privately and anonymously. The decentralised nature of the network protects users from Government and/or malicious interference and spying. It is also resilient to malicious attacks to the network.

1.3 Applications

- Communication system for military on hostile soil or over hostile and untrustworthy networks.
- For journalists and activists under regimes that don't grant free speech
- For businessmen or politicians that require a private channel for communicating confidential information.
- For whistle-blowers, since the network cannot be censored.

1.4 Similar applications and inspiration

- Google Duo: A proprietary client-server based video conferencing platform indexed using mobile telephone numbers.
- BlackBerry Messenger: A secure text messenger that used BBM pin for indexing users.
- Bitcoin : A virtual and untraceable currency system using peer to peer networks, and public key cryptography
- BitTorrent : A file sharing platform built over a peer to peer network
- Tor Network : A secure network that uses onion routing for untraceable internet access.

2 Overall Description

2.1 Product Perspective

2.1.1 User Interface

A minimal graphical interface with add contact button, call button and answer button when an incoming call occurs. A disconnect button during call. The UI is expected to be run on QT supporting machines.

2.1.2 Hardware Interfaces

A video camera, a mic for audio recording, a screen for displaying video, a speaker for audio, a network card for internet connection.

2.1.3 Software Interface

Asymmetric key cryptography libraries, audio/video encoding/decoding libraries, sha256, tcp/udp libraries,

2.1.4 Operating Environment

The first version is expected to run on Linux machines. It may be ported to android, ios and Windows. Hardware requirements include a platform with a video camera, a microphone and a network connection.

2.2 Design and Implementation Challenges

- Accessing a node behind a NAT poses a technical challenge for a truly decentralised network.
- Sending video and audio over an unreliable network requires specific encoding techniques.
- A mechanism for synchronisation of address book and call list updates.

3 Specific Requirements

3.1 External Interface Requirements

The external requirements involve a seed server that enables the peer to connect to the Pype network. The IP address of the seed server is precoded into the peer software.

3.2 Functional Requirements

3.2.1 Add contact

Enables the user to add a contact to the software by adding the public key of the callee.

3.2.2 Make call

Enables the user to call someone on their contact list and start video calling.

3.2.3 Answer call

Allows the user to answer call received.

3.2.4 End call

Allows the user to disconnect a call.

3.2.5 Generate new keys

Allows the user to generate new keys, and hence a new address.

3.3 Software technologies used

- Peer to peer networks
- Asymmetric key cryptography
- Streamable h264 for av encoding
- Cryptographic signatures.

4 Design

4.1 Data structures

Implementation of Pype requires a few tailored data structures. They are

- Address Book (AddrBook)
- Peer list (peer_list)

4.1.1 Address Book

Address book is a distributed document stored on each peer in the network. It is a map contains encrypted contact information indexed using a hash of individual public keys used to uniquely identify a person. It is periodically updated.

hash_addr	cypherText

hash_addr (Hash address) is a 256 bit sha256 hash of a public key generated by the corresponding peer. Each peer has a pair of public key and private key. cypherText is a plain-text encrypted using the private key corresponding to the public key used to make hash_addr. The cypherText acts as a signature for the peer. The plain-text contains contact information and a meta data field as shown in the following format.

meta_data
net_addr
hash_addr

addr_book[hash_addr] returns corresponding cypherText or null if hash_addr does not exist. cypherText can be decrypted using the corresponding pub_key. The address book is maintained and synchronised by all peers.

4.1.2 Peerlist

Peerlist is a list of ip addresses to which a peer is connected to. The peerlist, found on each peer is the backbone of the Pype network. peerlist is independently maintained by each peer.

peer1 net_addr	control_flags
peer2 net_addr	control_flags
...	...
peerN net_addr	control_flags

4.2 Network

Peer-to-peer (P2P) computing or networking is a distributed application architecture that partitions tasks or work loads between peers. Peers are equally privileged, equipotent participants in the application. They are said to form a peer-to-peer network of nodes.Peers make a portion of their resources, such as processing power, disk storage or network bandwidth, directly available to other network participants, without the need for central coordination by servers or stable hosts.Peers are both suppliers and consumers of resources, in contrast to the traditional client-server model in which the consumption and supply of resources is divided.

However, because of the widespread use of NAT architecture, our network cannot be truly decentralised. The network uses a seed server to bypass NAT restrictions and act as the entry point to the network.

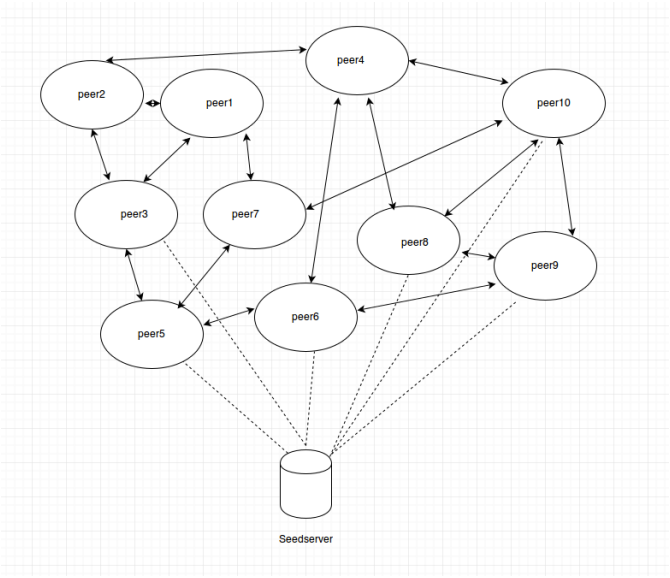


Figure 4.1: A sample network

4.3 Interfaces

4.3.1 Seedserver

The seedserver provides the following function calls

- *peerlist* get_peerlist()
- *bool* connect2peer(string peer_ip_addr)

4.3.2 Peer

The peer software provides the following function calls for accessing nodes.

- *bool* newPeerRequest(string peer_ip_addr)
- *addr_book* getAddressBook()
- *call_list* getCallList()
- *bool* connect2call()

4.4 Architecture

4.4.1 Overview

The architecture includes a seed server and a collection of peers. A peer runs a software that enables it to connect and communicate with the network. Since the NAT prevents nodes from forming direct connections, Pype uses a seed server. A new peer connects to the seed server and requests a peer list. The seed server provides a list of peers and stores the details of the requesting peer. The peer then asks the server to let it connect to a few random nodes and form it's own peer list, and the seed server allows the connection. The peer maintains a connection with the seed server by occasionally polling and checking for connection requests, which allows the seed server to communicate with the peer behind a NAT, since the address of the seed server remains registered on the NAT. To connect to a node on the network, the peer requests a connection to the seed server and sends packets to the node's ip address. These packets are ignored by the NAT on the router closest to the destination node. The seed server sends a message to the node which contains the ip address of the requesting peer. This message gets through since its source ip address is already registered in the NAT of the destination node. The node tries to connect to the requesting peer by sending a message to its ip address. Since, the ip of the node is already in the NAT of the requesting peer, the message gets through. When the node sends a message to the source peer, the ip address of the source peer gets registered on the NAT of the destination node, enabling the source peer to communicate with the destination node and vice versa. Once connected to a few nodes, the peer is a part of the Pype network. The peer also maintains a connection with the seed server to enable new peers to connect to it.

After establishing its connections to the network, the peer asks for the address book and call lists. The address book is a data structure that holds a hash address and a cypherText field. The Call list is a list that contains a callee hash address and a caller hash address field. The address book and call list are updated at regular refresh intervals.

Once updated, the peer scans through the call list to see if its hash address is listed in the callee list. If found, the peer establishes a UDP connection with the caller.

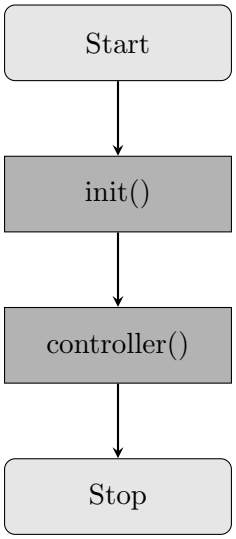
If the peer wants to make a call, it updates the call request on the call list and tries to make a UDP connection with the callee.

The address book contains two fields, and acts like a map. The first field is called the hash address. Each peer has a hash address. To create a hash address, the peer generates a pair of asymmetric keys. The sha256 hash of the public key is the hash address of the node. The cypherText field contains a plain-text that is encrypted using the peer's private key, and it contains it's ip address and some meta-data.

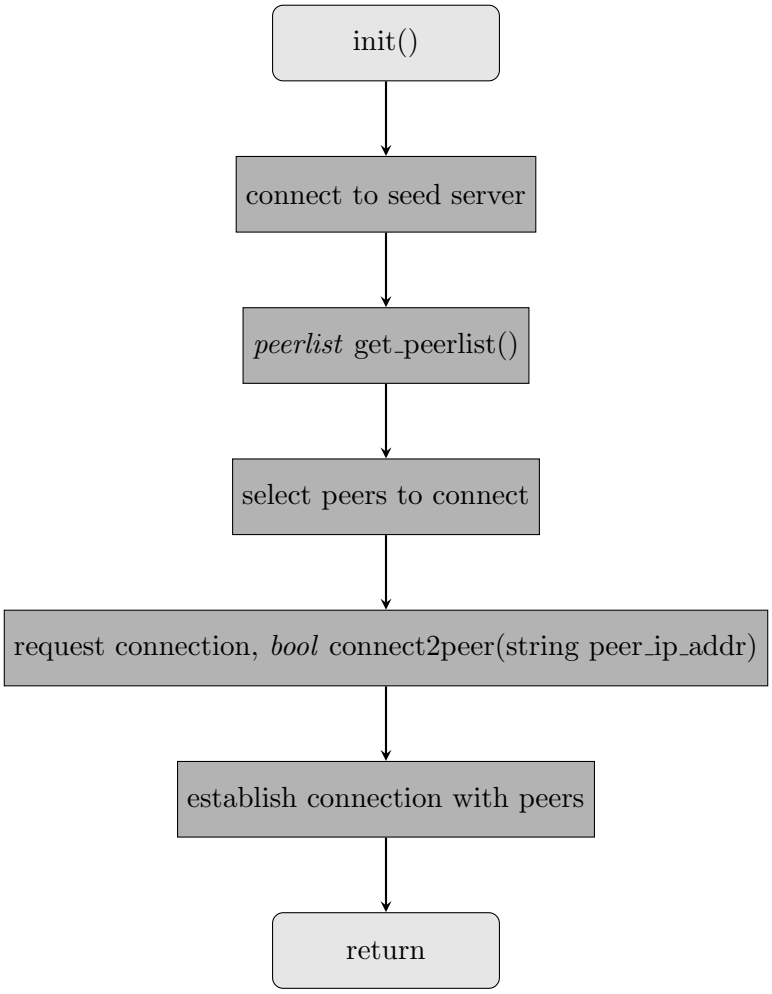
The network is anonymous because nobody can infer which ip contacts which other ip, unless they hold the public key of both peers. The network is also secure, because it uses encryption and defends itself from third party spying. Since the network is peer to peer that lacks a central server for supporting it, the network is resilient and protected from ddos and other attacks.

4.4.2 Control Flow

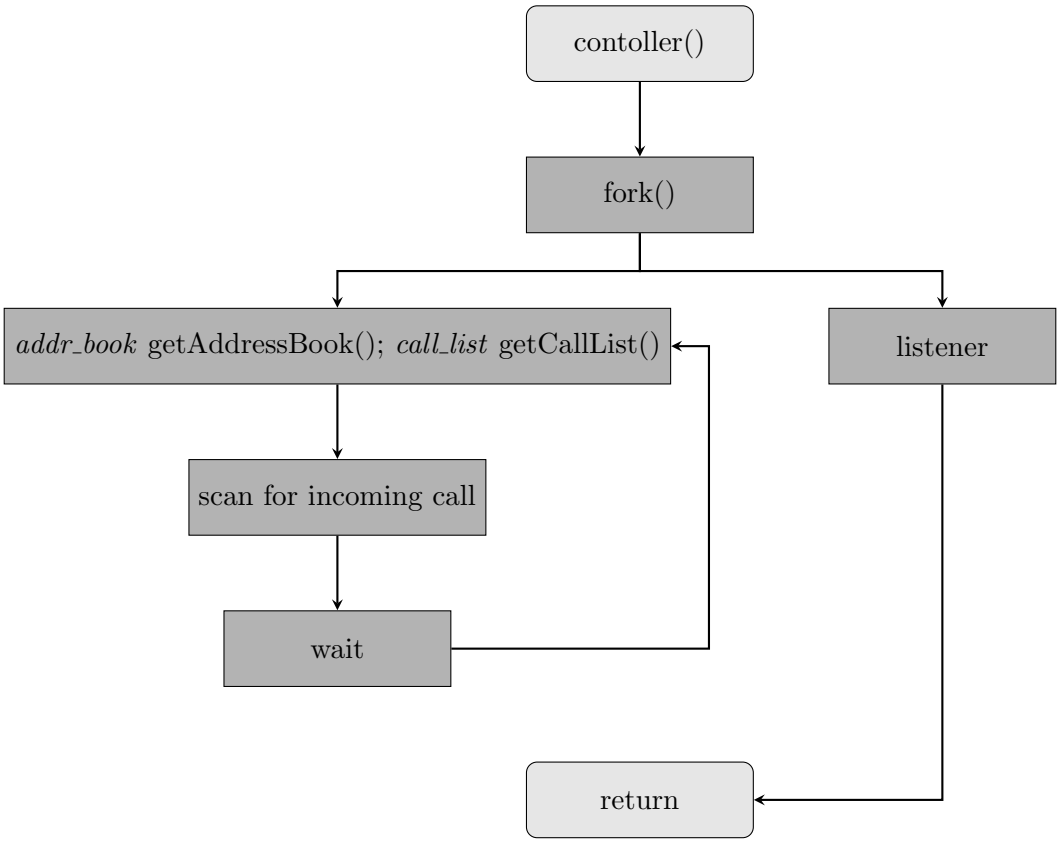
Peer process



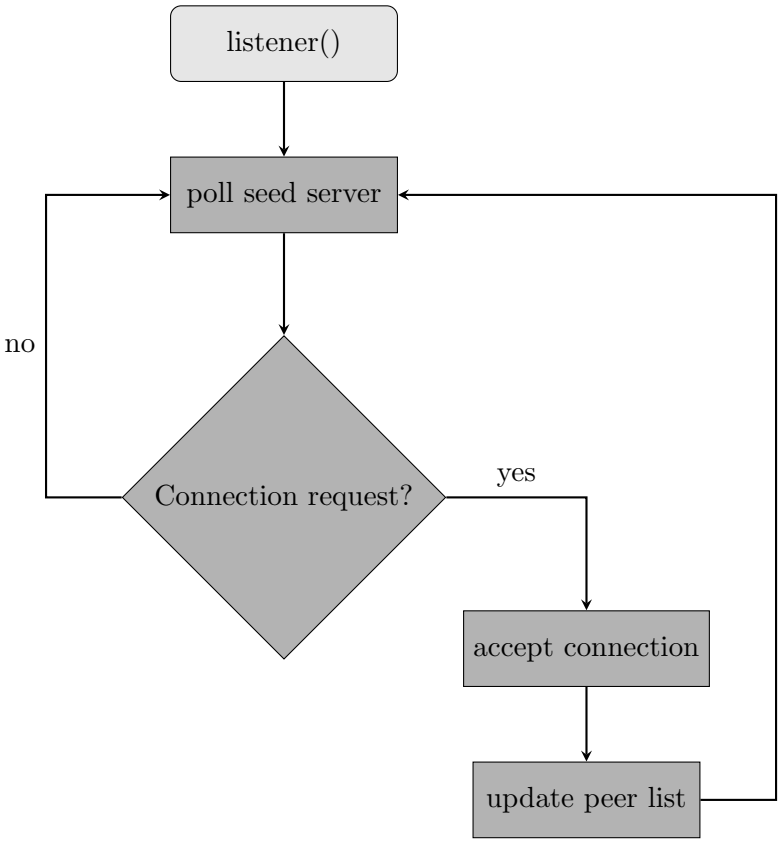
Initialisation



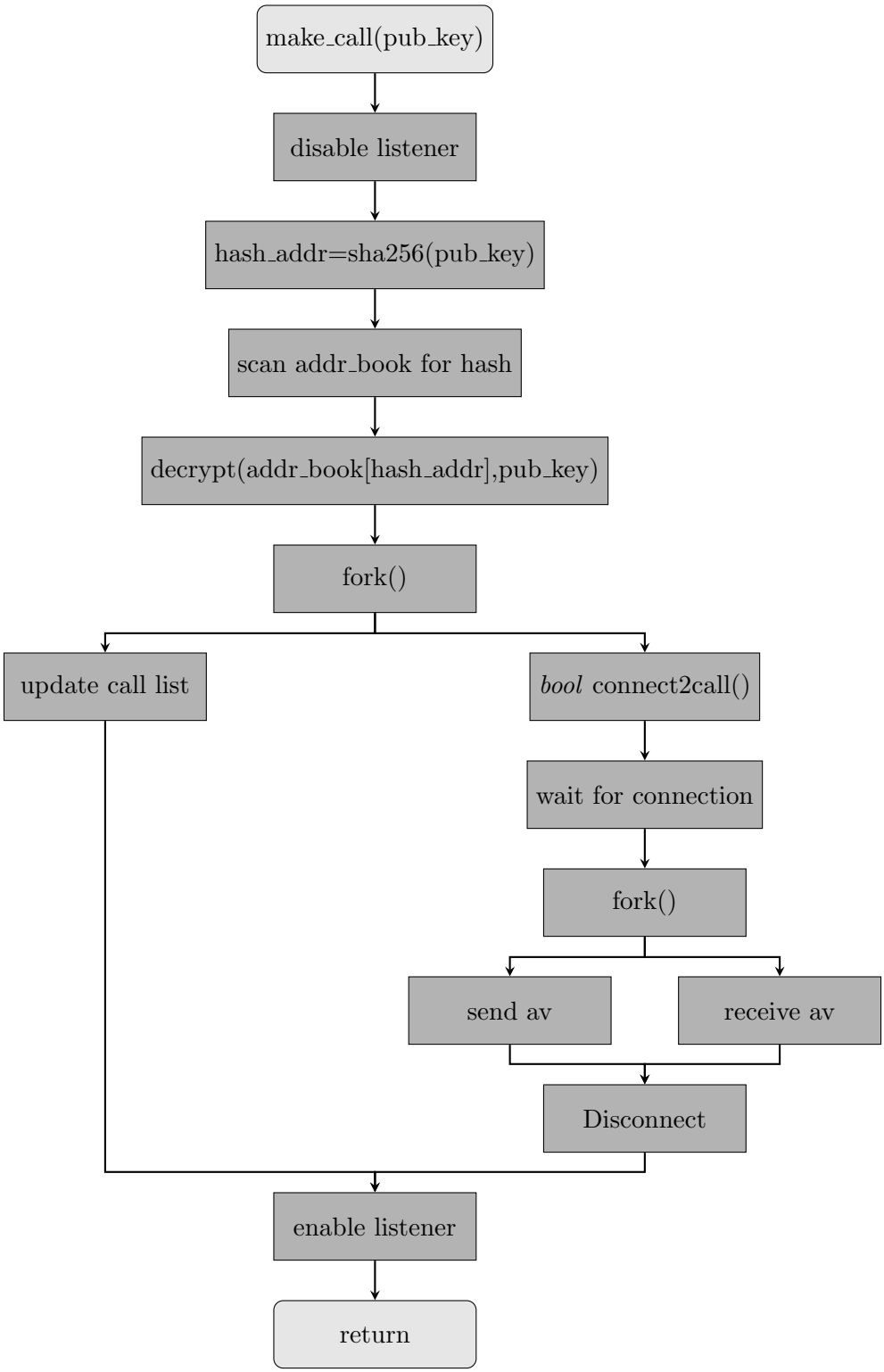
Controller


contoller()
fork()
addr_book getAddressBook(); *call_list* getCallList()
listener
scan for incoming call
wait
return

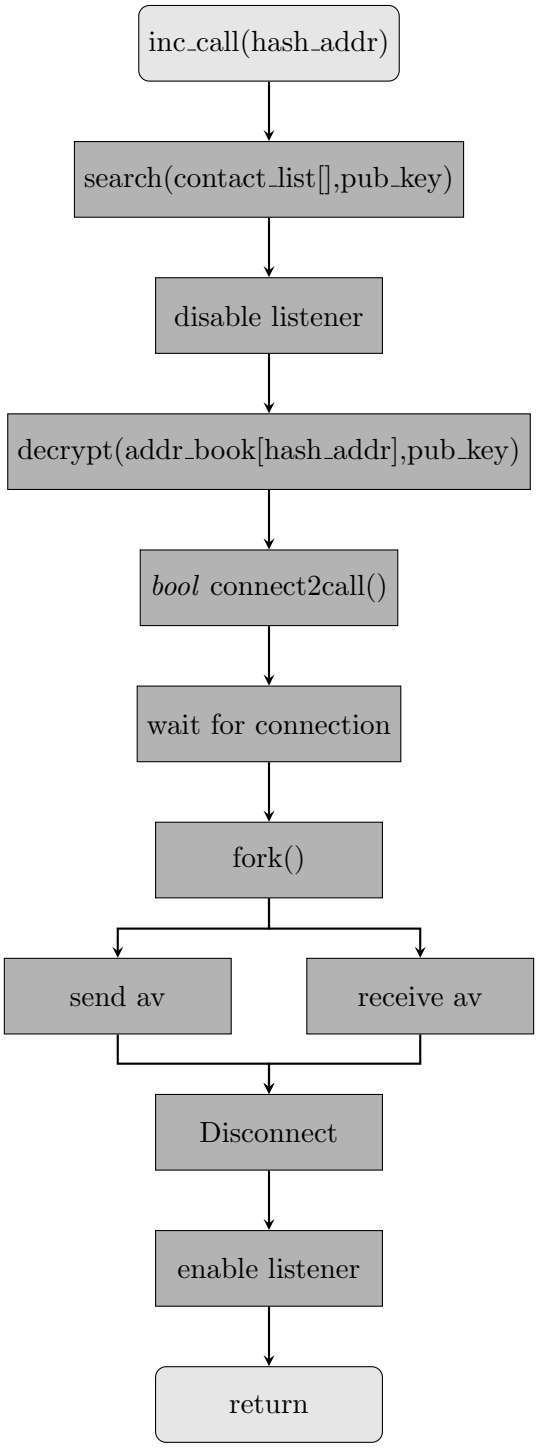
Listener


listener()
poll seed server
no
Connection request?
yes
accept connection
update peer list

Make call


make_call(pub_key)
disable listener
hash_addr=sha256(pub_key)
scan addr_book for hash
decrypt(addr_book[hash_addr],pub_key)
fork()
update call list
bool connect2call()
wait for connection
fork()
send av
receive av
Disconnect
enable listener
return

Incoming call



4.4.3 Class diagram

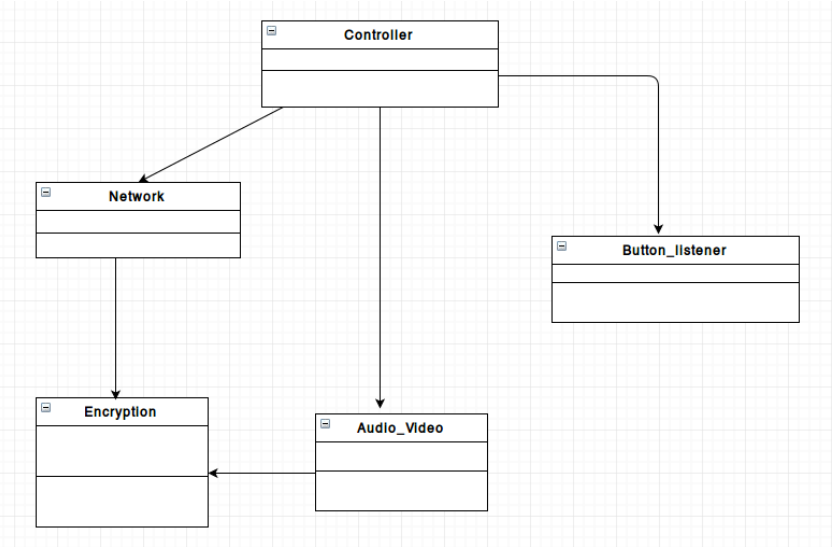


Figure 4.2: Class diagram

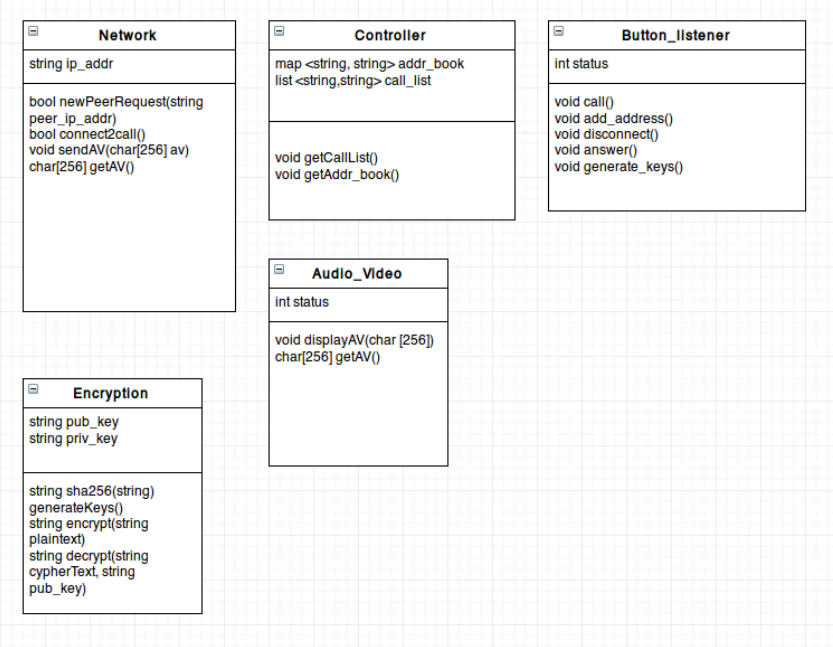


Figure 4.3: Detailed class diagram

5 Security

Each user publishes a hash of his public key along with a signature. Since a hash cannot be reversed and sha256 is long enough to beat brute force attacks. Users can communicate with each other, only if both parties have access to each other's public key. This secures the communication from eavesdropping. A user cannot perform a man in the middle attack due to usage of cryptographic signatures are nearly impossible to forge. The network cannot be compromised, since compromising the network requires taking down every node in the network, which is unfeasible. Taking down a seed server is a possible attack, but the seed server can be replaced easily and quickly, if necessary. Changing seed server is cheap compared to changing a server in a client server model. Users can, however infer communication between two nodes if the user has public keys associated with both nodes, however, for increased privacy, users may generate separate keys for separate conversations.

6 References

1. Peer-to-Peer Communication Across Network Address Translators by Bryan Ford *Massachusetts Institute of Technology baford@mit.edu*, Pyda Srisuresh *Caymas Systems, Inc. srisuresh@yahoo.com*, and Dan Kegel *dank@kegel.com* http://www.brynosaurus.com/pub/net/p2pnat

2. SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies by *Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A. Kroll, Edward W. Felten*