Radio

no_independe@danwin1210.de

May 28, 2022

1 Introduction

The original intention of the internet was that every person was supposed to have their own page, therefore without the need for a social medias to do it for the person. Perhaps some features such RSS feeds could be used, but nothing more than that was necessary. The social network proposed intends to make the internet closer to the way it was back then, but also improving on certain aspects that seem to have made people stop creating their own websites. It's called radio.

Notice that the old internet paradigm is not attractive because of the volume of data that needs to be sent by some people. To back up this claim, I'll use an example. Suppose that Billy is a youtuber with 10 million followers, but wants to go back to the old way of the internet and host his entire youtube channel on a single personal computer.

Consider that Billie's computer would need to give the same video for each follower, every time they asked for. Let's say with a quality of 720p a video file has 7Mb per minute recorded (Took this estimate from a real downloaded video), therefore for every minute of a video produced by Billy, Billie's computer would need to send 70Tb in total to his followers. Not only that, he stil needs to deal with the multitude of comments that appear on his page and send to each user the set of comments that they want.

As you can see, he cannot host his youtube channel on his personal computer, due to the processing power necessary and the amount of internet spent hosting. I'll show you workarounds to both of those problems and how to combine the solutions. Those workarounds are implemented in the actual radio algorithm.

Note that Billy could be more clever and send only the torrent link to each follower, therefore removing a giant load of his page and in case his site gets shutdown, People can still share the link in their own webpages. If a link needs a 1kb of memory, then he would need a bandwith of 10G per video uploaded. Now it would be feasible for him to host a page on his computer. Therefore we have a way to deal with uploading videos to every user without causing much harm to Billie's Internet.

To deal with comments we will use multicast protocol implemented on internet protocol version 6 (IPv6), but first let me explain what's multicast. A multicast is similar to radio, where a radio station will transmit a signal a single time and anyone who listens to the frequency of the station will receive the message. The frequency will be the multicast address. Therefore by using a particular multicast address Billie's followers can send comments to each other follower without the need for Billie's computer to do any work. They all use the same address that was decided by Billie's machine.

Note that if any of his users are listening on the same multicast address, he can send the torrent links through the multicast address, Therefore only needing to send the link once, not for every follower. Not only that, but also sign up's could be done by using the multicast channel. As it was said before, those workarounds are how Radio operates: comments, posts and sign up's are sent through multicast and a user only post the link of the torrent of a file, not the file itself.

To make the explanation more simple I omitted problems like: How the users are aunthenticated (How to prove a post came from Billy or a comment came from some particular user)? What happens in the case someone is not present when a message is sent? How does a user signs up in Billie's page to begin with? Since the answers are more technical, the solution is shown, respectively, on the sections: "Safe mesage", "P2P Network and 51%attack" and "Sign up". The other sections show the algorithm with more depth and what to contribute.

2 IPV6 and multicast

The original idea of the IPV4 was to have an unique address to every computer, but since there are 4 billion addresses, in a world with 8 billion people and much more machines, it wasn't enough.

Since, it wasn't possible, it was adapted IPV4 to IPV6. The IPV6 address consists of 8 groups hexadecimal numbers, each having 4 digits. Therefore having

1

$$2^{4*8*4} = 2^{128}$$

Possible binary representations of an address. So for example, "abcd : abcd : abcd : abcd : abcd : abcd : abcd : abcd" would be a valid IPV6.

Some of the addresses are reserved for specific purposes. One of those types of address are designated to multicast channels. Multicasts channels are addresess devoted to streaming of data to multiple locations. Every person who listens on one of those addresses (hosts on that address) receives a message sent by someone on that address. The first two bytes of the address are ff, the third byte represents the lifetime of the address (0 for permanent and 1 for temporary), the fourth the scope (is it hosted only on the local network, is it global?). The multicast addresses, are all going to be with unlimited time and global, therefore all multicast addresses of the algorithm start with "ff0e".

3 Safe message

To identify the action of each user on the network, it was created the idea of safe message. A safe message, works as follows, every person who belongs to the same network, has for every user the hash of his key, and when a user sends a message, he sends together with his message the key of the old hash and a new hash. The key of the old hash certifies that this user is the the only user who has the key. The new hash will replace the old hash, and the user who sent the message will have the corresponding key to that hash.

To be more specific the first 128 characters of the safe message would be the key of the old hash, the next 128 characters correspond to the new hash, the next character the protocol (this will be explained later on), and the next sequence of characters would correspond to the username and the account where the user is sending messages.

Notice that those safe messages will be sent through multicast channel, which means if some user who follows the account is not present, he won't receive the message, nor the new hash. That's why we need peers to send the activity of the network when the user logs back. This will be explained on section 8.

4 Protocols

To interpret and send the correct response to a message sent to a node, the algorithm needs some sort identification of the type of message. The identification will be called a protocol. Here's the list of what each number mean in a request of the application

1 Sign up request. Done through multicast channel

2 Sends a signal to all the networks that the user follows, that he changed the key of its own network. Done through multicast channel

3 Send a request to peer to become neighbour and receive actions. Done through a server hosted from computer.

4 Send file to all the followers on the network. Done through multicast

5 Sends a message back to peer B that user A received the request of user B to become peers. Done through a server hosted from computer

6 Sends files to the peer who requested. Done through a server hosted from computer

7 Sends comments through multicast.

8 Sends request to receive data of all the users with their hashes and the POWs they used. Done through a server hosted from computer

9 Sends message to the network of the user to block some user. Done through multicast

5 Account

Every account will be assigned an specific multicast channel. In this channel the user can send posts to those that are listening to the channel, those will be the followers. Also the followers are allowed to comment on posts, and as duty, they need to send the files of the network when requested by a peer.

To avoid conflicting usernames, the algorithm will generate 10 random characters at the end of the username, since there can't be no commonly shared database with all the usernames.

6 Sign up

To avoid creating unlimited number of accounts, when sending a request to join a network, the user will also send a proof of work. In this case, will be a key such that when mapped to a sha512 hash, the first 3 characters will be 0 (this can later be replaced by a better proof of work). User also generates a key, and send the hash of the key to the network to perform safe messages later.

7 P2P Network and 51% attack

First and foremost consider that what I'll call an action on the network can be a comment, a sign up, a post, or anything that is sent through a multicast channel.

Secondly, The peers will be responsible for sending the actions that happened on the network during the time a particular user was off. Since any user of the network receives all actions, because they all listen to the same multicast address, we will assume that the peers all know the actions that were sent during the time he was off.

Suppose that part of the peers of a user are malicious, that means that they deliberatly decide to send wrong actions to the recently logged in user, how can we deal with that? Also suppose that most of the peers are malicious (that's why we call it a 51% attack), then how can we deal with this sort of attack?

Before showing a possible solution to the problem, it's solved the following and then I'll give some highlights: Given that people 1, 2, 3..., n have probability of sending the correct action of

2

$$p_1, p_2, ..., p_n$$

Given the set

3

$$A$$

Of people who agree on some action "a" is the correct one, what's the probability that "a" is correct? It is the probability of people who agree being correct and people who disagree being incorrect, given that those two groups of people will either be right or wrong about "a". Therefore the probability is

4

$$P_A = \frac{\prod_{x\in A} p_x \prod_{x\notin A}(1-p_x)}{\prod_{x\in A} p_x \prod_{x\notin A}(1-p_x) + \prod_{x\in A}(1-p_x) \prod_{x\notin A} p_x}$$

On the numerator the first product correspond to the probability of people who agree being correct, the second product to the probability of people who disagree being incorrect. From that it follows the formula. To simplify, define

5

$$p := \prod_{x\in A} p_x$$

6

$$p' := \prod_{x\in A}(1-p_x)$$

7

$$a := \prod_{x} p_x$$

8

$$b := \prod_{x}(1-p_x)$$

Since

9

$$\prod_{x\in A} p_x \prod_{x\notin A}(1-p_x) = \frac{\prod_{x\in A} p_x}{\prod_{x\in A}(1-p_x)} \prod_{x}(1-p_x) = \frac{p}{p'} b$$

And

10

$$\prod_{x\in A}(1-p_x)\prod_{x\notin A}p_x = \frac{\prod_{x\in A}(1-p_x)}{\prod_{x\in A}p_x}\prod_x p_x = \frac{p'}{p}a$$

Therefore

11

$$P_A = \frac{(pb/p')}{(pb/p')+(p'a/p)} = \frac{pb}{p'}\frac{pp'}{(p^2b+p'^2a)} = \frac{p^2b}{p^2b+p'^2a}$$

Notice the very important fact for later, consider for some person k. Call

12

$$\hat{p} := \prod_{x\in A-\{k\}} p_x$$

13

$$\hat{p}' := \prod_{x\in A-\{k\}} (1-p_x)$$

14

$$\hat{a} := \prod_{x;x\neq k} p_x$$

15

$$\hat{b} := \prod_{x;x\neq k} (1-p_x)$$

We have

16

$$p = \hat{p}p_k$$

17

$$p' = \hat{p}'(1-p_k)$$

18

$$a = \hat{a}p_k$$

19

$$b = \hat{b}(1-p_k)$$

Therefore

20

$$P_A = \frac{\hat{p}^2p_k^2\hat{b}(1-p_k)}{\hat{p}^2p_k^2\hat{b}(1-p_k)+\hat{p}'^2(1-p_k)^2\hat{a}p_k}$$

Consider that using formula (11), but removing the person k from the decision, we have probability of action being correct is

21

$$\hat{P}_A = \frac{\hat{p}^2 \hat{b}}{\hat{p}^2 \hat{b} + \hat{p}'^2 \hat{a}}$$

Check that

22

$$p_k = 50\% \to P_A = \frac{\frac{1}{8} \hat{p}^2 \hat{b}}{\frac{1}{8}(\hat{p}^2 \hat{b} + \hat{p}'^2 \hat{a})} = \hat{P}_A$$

23

$$p_k \to 100\% \Rightarrow P_A \to \frac{\hat{p}^2 p_k^2 \hat{b}(1-p_k)}{\hat{p}^2 p_k^2 \hat{b}(1-p_k) + \hat{p}'^2 (1-p_k)^2 \hat{a} p_k} =$$

$$= \frac{\hat{p}^2 p_k^2 \hat{b}}{\hat{p}^2 p_k^2 \hat{b} + \hat{p}'^2 (1-p_k) \hat{a} p_k} =$$

$$= \frac{\hat{p}^2 p_k^2 \hat{b}}{\hat{p}^2 p_k^2 \hat{b}} = 1$$

So while the action of person k with probability 50% of telling the truth has no influence in determining if the action is correct, a probability of 100% completely determines that the action is correct. Keep that in mind.

If we knew the probabilities, we could use the formula above to calculate the probability the action being correct, and in the case of an attack choose the one with greatest probability. Moreover, the formula above gives a linear time algorithm on the number of people (first is calculated "a" and "b", then for each action we compute "p" and "p' ").

It's going to be shown a procedure to give an estimate of that probability, such that the people that have more influence on the decision of some user are the ones who shared more files, during more time. The algorithm runs locally on every computer on the network, meaning everyone will most likely have different sets of probabilities for every individual. Every user starts with a 50% of probability and every time a user sends the correct action (determined using the greatest probability of being the correct one), the probability of the user increases according to a map that has maximum at 1, otherwise it decreases with minimum at 50%.

If the user sends the correct file and his probability is

24

$$p_u$$

Then the probability will be later

25

$$(1 + 19p_u)/20$$

Otherwise

26

$$(1 + 18p_u)/20$$

The reason why is the following. Given the map

27

$$(1 + ax_n)/b = x_{n+1}$$

The fixed point of the map will be such that

28

$$(1 + ax)/b = x \rightarrow 1 + ax = bx \rightarrow 1 = (b - a)x \rightarrow b - a = 1/x$$

For fixed point x=50% we would have b-a=2, for a fixed point at 100%, we would have b-a=1. Then it was chosen an arbitrary value for b=20.

Therefore the algorithm works as follows: "People who share the "correct" action will increase their score (probability of choosing the correct action), people who didn't will decrease. The correct file is determined by the formula 11, based on people's score. In the case of a tie, it's chosen the action that most users sent".

If a person decides to create many users at one specific moment to make the network send wrong files, it will need to deal with the problem that those users have very low influence at the beginning, meaning the attack will need to be sustained for much more time to work properly (the attacker needs to make his score high enough for enough number of users to start the attack).

8 Contributing

Those are areas that I found that are of interest to the development of the program or maybe crucial.

ISP blocking ISP's sometimes allow ipv6, but block multicast outside their network, they don't become nodes of the multicast tree. It's necessary a way to overcome the problem.

Security The peers send files directly to each others, those could be potentially infected with a virus, therefore, when receiving files, the server needs some type of scan to filter certain type of code.

DDoS Even though it's still security, it needs special attention. The algorithm needs to have some defense against DDoS attacks on server application and multicast channel

Design Needs a better design, if possible that doesn't use javascript and can run locally.

Streaming The multicast is already appropriate for streaming, therefore it would be appreciated ideas of how to add the feature, so would any help.

Crypto-Wallet Addresses This could serve as a revenue for content creators. Any ideas of how to add the feature would be much appreciated, so would any help.