Leonie Geyer 345468
Lorenz Schmidt 346956

Project proposal: Overlay Networks for peer-to-peer communication

We came up with the following propsal, a synchronised database relies on a strong interconnection between different peers in the same network. Every peer has to know with whom he can speak and every transition of the database state has to be distributed reliable in the network. For this we want to create an alternative to TCP with similiar goals (reliability and congestion), but for a peer-to-peer network. Because the integration of a new protocol into the internet is difficult, we will just use UDP as a wrapper (like QUIC). The protocol should fulfill the following properties:

- create a strong connected overlay network between different peers based on the SCAMP paper (SCAMP : Peer-to-peer lightweight membership service, for large-scale group communication)
- build a broadcast tree in the overlay network to minimise sending overhead

Schedule

28.11: connect peer with overlay network

05.12 /12.12 send packages with broadcast tree

19.12: find or program a simple UDP simulator

09.01/16.01: platform development of catascii and a more sophisticated platform (catdit)

23.01/30.01: futher ideas

UDP simulator

In order to simulate a large scale peer-to-peer it would be useful to have a simple UDP simulator. A very simple approach would just intercept send and receive syscalls and forward them to the respective peers. To handle many peers simultanously we could use epoll or a similiar approach. We assume here that we have no packet losses (at the beginning) and some random delay with gaussian distribution.

The simulator should be able to print the network graph in a meaningful manner, also it should be able to count occurences of packets at one peer (to measure the overhead). The parameters are the number of peers, the delay distribution and probably packet loss later on.

Connect peers (SCAMP)

The first part of the propsal is an overlay network, which connects all peers in a strong manner. We will use the SCAMP protocol here, which is very simple to implement. In order to build a local view of the

[1]: S CAMP : Peer-to-peer lightweight membership service for large-scale group communication

network, a new peer will send a Join message to their contact. The contact will then forward the packet to all their neighbours. These send this forward Join to their neighbours with the propabaility p = 1/(n + 1), where n is the number of neighbours (including the source of the packet). They will accept the connection with the same probability, so that we a equal probability distribution. Furthermore the contact will also forward c additional copies to random neighbours, which are forwarding/accepting them in the same way. This will lead to an average of n = (c+1)log(N) neighbours for every peer. This ensures a strongly connected network as can be seen in [1].

Send packets

Without a tree the network would have a very high overhead, because a single packet would be received by a peer multiple times. To prevent this overhead the network should know a broadcast tree. This tree defines the route packets are taking, preventing duplications.

At the beginning everyone is forwarding packets to every neighbour. If then they receive a duplicate, they know that the packet was already received at their neighbour. So they don‘t need to forward this packet in order to reach every peer in the network. The connection is then deleted from the broadcast tree. It follows that the first sending peer defines the tree structure and is the root node. It is useful to know that every peer has a local part of the global broadcast tree stored.

Use case

We have now a synchronised database and can for example use it for cat pictures. Every user is a peer in our network and can submit as many cat pictures as they want. The cat pictures get synchronised across all connected devices. Furthermore a user can upvote his favourite cat picture to make it more visible. The peer-to-peer network structure is especially useful for this because the pictures can‘t be lost as long as a single peer stays in the network.

Additional ideas

Database with approving scheme

A distributed database can be modeled with transitions between database states. A single packet in our network will contain a single transition from the local database of the peer. Every peer receiving the transition will update his database and reference to the hash of this transition in at least one following packet. Therefore each packet contains a list of approved transitions, creating a directed acyclic graph. In case a packet is lost (because of an unreliable) the database is inconsistent, because a transition is lost. This peer will then not refer to the lost transition and all his neighbours can resend a missed packet.

[1]: S CAMP : Peer-to-peer lightweight membership service for large-scale group communication

Security

This network has several problems concerning security. Firstly you could just forge a packet with of a any peer, because we don‘t use signatures. A simple approach would be using a signature scheme to sign any packet traveling in the network. Secondly the transmission is not encrypted, allowing anyone to read everything. (not really critical here because its open anyway) Lastly we don‘t know whether a peer is executing the join protocol correctly, he could just pretend to be not only a single peer, but many peers and take over the network. For this we could add signed join packets and allow any peer to observe these signed join packets.

Unallowed/allowed transitions

For a more sophisticated usecase it is compulsory to only allow certain transitions on the database, given a specific state. For example we could restrict every user to upvote a picture only once. We could now let every peer do a local decision whether a transition is prohibited and then kick the peer out of the network. Though this possible a attacker could split the network by publishing two contradictory transitions at the respective end of the network, splitting in half. This is a known problem in blockchain and measure should be taken to prevent this.

[1]: S CAMP : Peer-to-peer lightweight membership service for large-scale group communication