The problem we are going to address is how we effectively distribute keys that are used to access a chatroom. We want to maximize the amount of space that a user can acquire all the keys to access the room and we also want to minimize the amount of people broadcasting at any given time. Lets assume that for practical reasons we have 3 keys that need to be collected (A,B,C).

Lets say user a is broadcasting A, user b -> B and user c -> C.

n is the range in meters that any user can broadcast

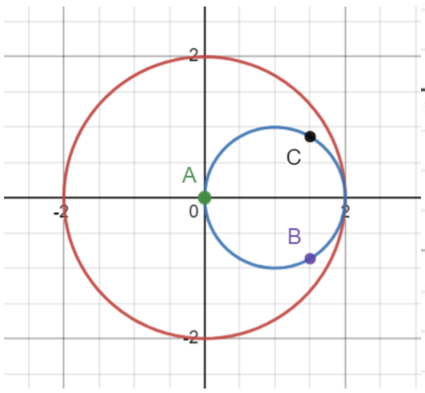
User a is located at any arbitrary point, lets just say (0,0) for simplicity

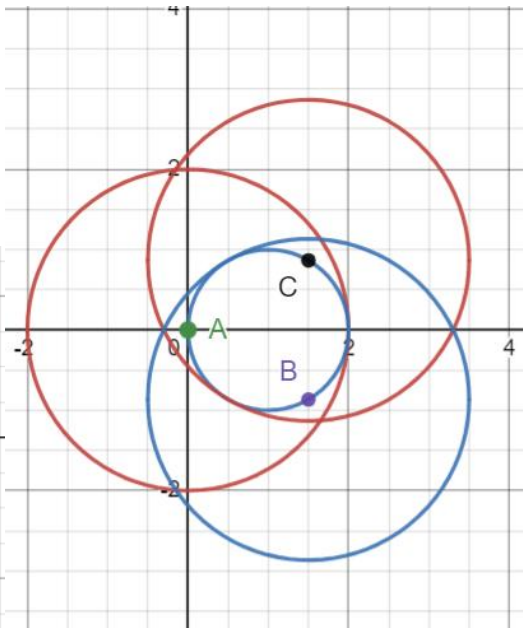
The largest circle where some user connect to a b and c is circle with ½ n radius, Where the broadcasters reside on the edge of the circle

Lets prove this

Assume the largest area we can acquire is when the points are equidistant, they lie on the points of an equilateral triangle, If some point on the triangle is a then b and c are pi/3 radians rotation away. So in the case where a is at (0,0) and it's the left most point on the triangle,

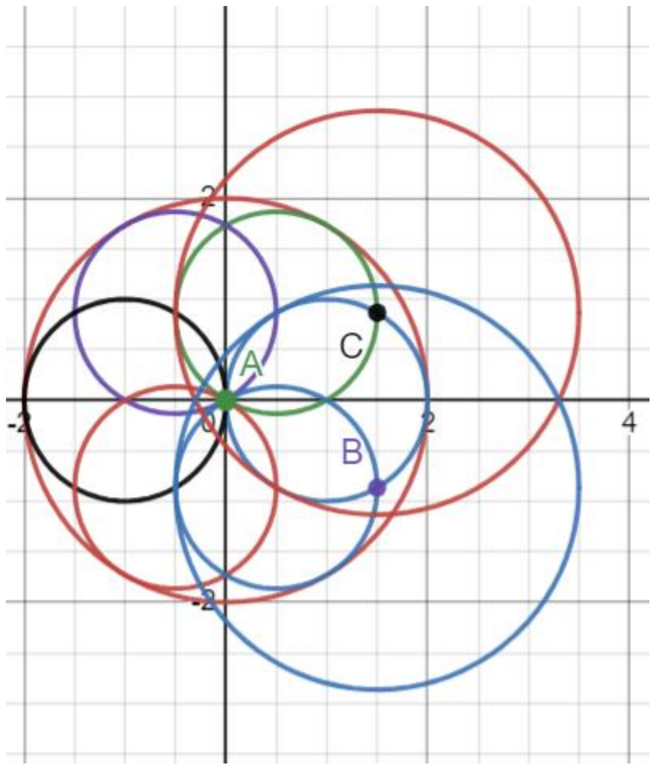
b is located at (1-cos(2pi/3), sin(5pi/3)), and c is located at (1-cos(2pi/3), sin(2pi/3))





If we draw the ranges for b and c we see a nice lineup

The inner circle is the largest circle where users can access the chatroom, now we can expand this concept and pretty much tile it as far as we need,



These points in space are calculated and the app can simply choose a user that is closest to those target points to be a broadcaster. We cant just choose any key either, if the adjacent circle next to about to be labeled is B then it broadcasts C to evenly distribute keys.

This is the optimal condition for maximum area coverage, we will probably never encounter this actually happening because it’s likely that the range will be wide enough to prevent this.