Opportunistic Network for Proof of Location

Abstract:

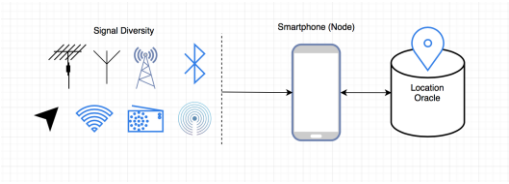
There's an old joke in American English that the three most important components of real estate are location, location, location. The new decentralization revolution, built on the internet revolution, brings this concept to money transfers, supply chain and just about any transaction. New models for building massively scalable applications are emerging. Bitcoin led the way with the first implementation of a block chain distributed ledger powered by probabilistic consensus protocols, and Ethereum led the way with implementation of executable functions known as “smart contracts.” The revolution is powered, under the hood, by new innovative implementations of cryptography, game theory and computer internet working protocols.

This entire virtual transformation has made location both more essential and yet more difficult to assess. Most business activities have an explicit or implied location, virtual or not. Think about it: whether you are buying a pack of gum or a multi-national corporation, the location of the assets, currencies, and participants is a critical factor. Yet this location is poorly implemented in the decentralized revolution. The reason is the difficulty of determining the location of an asset, coin or person when it is a virtual series of ones and zeroes.

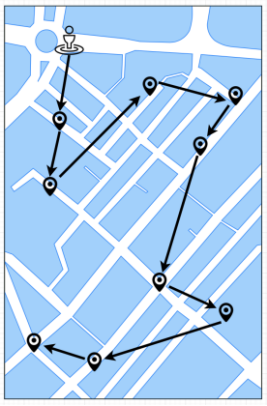
Platin.io brings location to the world of decentralization, supplying this missing piece. When Proof of Location (PoL) is provided via the Platin blockchain, a transaction fee is charged in “Platin” (PTN) that also serves as the “gas” for fueling proof of location requests. This enables for example to transfer virtual money to a certain location.

Goals:

1. Build Location Oracle Data using commercial databases of diverse signals
    a. Develop Smartphone Application (Android or IOS) that will know to:
        - Set a unique user DNS name (based for example on google account)
        - Detect Wi-Fi, GSM/LTE and Bluetooth signal strength
        - For each detected Wi-Fi signal – get its SSID, BSSID as well as calculate its distance from the AP
        - For each Bluetooth signal – detect protocol type:
            - If iBeacon – get its UUID, Major, Minor
            - If Eddystone –get the core Eddystone frames: -UID, -TLM, -URL
    b. The application should be able to save the above details for each signal with its corresponding GPS coordinates (latitude + longitude).
    c. For each Wi-Fi and GSM/LTE signal – add the GPS coordinates of its AP/Tower by downloading updated data from wigle.net, opencellid.org, or other online database. The former offers API access at api.wigle.net.
    d. Save the all the above oracle data to an external database in the cloud.



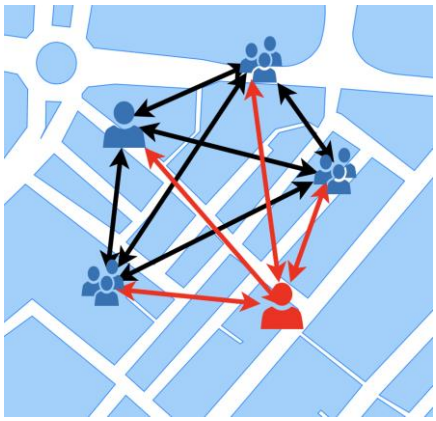
2. Collect Node Behavior Over Time by Assessing the behavior over time in a given node
    a. Move with your smartphone and the above application around your neighborhood and update the database with above data + time stamp in a predefined polling time



3. Implement Fraud Detection by Comparing a node's actual to expected behavior and assessing the difference
   - Run a server application in the cloud on the above data base (from step1 +2) a scoring algorithm for the user as signal histories accumulate and update the user's score at time intervals of your choice.
   - Assign high scores to fluid movement over time and penalize disjointed / incoherent movement, time lapses, "teleportation" and other suspicious signal and geo changes.
   - Consider the influence of BTE Beacons (trusted and cryptographically signed signals) on the trust rank and node score – for example elevator or printer

Commented [IA1]: Need more clarifications here what needs to be done actually

4. Verify Collected Data with Nearby Peers by Access proximity and signal patterns that are only available to opportunistically communicating peers
   - Add to the application option to the user to claim he is in a specific GPS location
   - Once claimed:
     - The server application should verify its claimed location vs the central data base (from step 1 +2)
     - If approved – server application should return a unique claim-ID
     - The user application will broadcast a BlueTooth Low-energy (BTL) Eddystone varying Tx power signals to peer nodes in the vicinity with: Its unique User name , Claim-ID, transmitted Tx power and its claimed GPS location.
     - Each neighboring listening node that receives the broadcasted signal will:
       - Estimate the distance and give a confidence level to the claimed GPS location.
       - Verify Claim ID with the server
       - Distributed option: Open TCP connections to all previous trusted users in local data base and send the resulting confidence level of the claiming user. Develop a protocol in which all neighbors will decide on the resulting score.
       - Centralized option: Send to server the confidence level of the claiming user. Develop an algorithm that will return resulting score.
       - TBD.. Analyze the effect of a node's trust rank (B2) on peer attestation and consensus.
       - TBD… If the requesting node is within range trigger a challenge (such as a human inaudible hypersonic sound) that can be heard and verified by one or more neighboring nodes.
       - TBD.. Update the requesting node's trust score based on the confidence level and trust level of the witness nodes.
       - TBD.. Describe any potential attack surfaces and methods by which your verification procedure can be spoofed.
       - TBD… Create a simulation demonstrating the ratio of honest nodes to dishonest nodes and the threshold at which colluding dishonest nodes break secure verification by honest nodes.



Requirements:

Internet Networking Course, Data Base Course, Java, Android / IOS development, Machine Learning

Guided by:

Dr. Lionel Wolberger, PhD. from Platin.io

Prof. Dr. Srdjan Capkun (ETH Zurich)