White paper to describe Quantifiable trust propagation of claims using EAS

Summary:

This project is an analysis of a trust propagation algorithm that can calculate ‘confidence’, i.e. trust, of a claim and an identity based on attestations and the trust score of attesters. In other words, a trusted source of attestations can make claims of high confidence, that translates to attestees that have high trust scores themselves. This method takes into account a web of attestations, which can be made of many users and entities, each making attestations about each other. The goal of this algorithm is to provide a simple confidence metric about some claim. As an example, we discuss using attestations to quantify public confidence in the safety of smart contracts.

For this project, we use Python to simulate fake users and fake attestations in order to test the algorithm effectiveness. We include honest, dishonest, and trusted actors. Our goal is to create a report on the effectiveness of the algorithm, the design of the schemas, and optimize the code before implementing to the Ethereum Attestation Service (EAS). We also discuss how the hackathon partners, such as Gitcoin Passport, Disco, Orbis, Intuition, and others would be included in this algorithm. In short, we expect these partners to create attestations on EAS, as Gitcoin Passport already has, and play the role of ‘trusted’ actors in this algorithm. The end result of our algorithm are ‘confidence scores’ which could be posted as attestations on EAS.

Section 1. Introduction:

#Points to touch:

# 1. Calculating and propagating trust metrics in decentralized networks has been studied for many years. In the 2000s, EigenTrust was created to calculate trust of seeders in a decentralized peer-to-peer file sharing network, with the goal being decrease chances of spreading viruses from untrusted sources.

# 2. For this approach, we use this algorithm by [Gruner et al] due to its heavy reliance of attestations and compatibility with the Ethereum Attestation Service and Ethereum itself. The algorithm requires 3 main components. An attester, a claim, and the reciever (attestee). In general, the calculations described in section 2 allow for the attester to propagate its trust score, to the claim, and then to the reciever. However, in the decentralized network, others can also make attestations to support the attester, or a specific claim, or the reciever directly. In this system, honest actors should be able to achieve high trust scores, true claims should have high confidence scores. In general, this approach helps quantify the wisdom of the masses, but it does not achieve the granularity and proveability of zero-knowledge proof based systems.

# 3. To avoid conflict of interest, our service would not be involved in the business of making any type of ‘claims’. We would not create attestations to the human-ness of someone, or the safety of a smart contract. We would only be involved in running the algorithm and continuously making attestations about the confidence score attached to each claim, attester, and attestee.

# 4. The algorithm is fairly straightforward and is not meant to be proprietary. While our service may be the first one to impliment it and start making attestations on EAS, another provider can also collect the public data to run the algorithm and make their own attestations. People may even make confidence/trust attestations on our service if they agree/disagree with us!

# 5. In addition, the algorithm may also be customized or individualized. For example, if a user would like to give additional weight to friends or other trusted sources, they may do so, and run the calculation to generate custom confidence scores for all interconnected claims. This algorithm can be continously running to dynamically calculate confidence scores as new attestations are added or removed.

Section 2. The quantifiable trust/confidence model:

# Show the math equations and figures here and describe the logic.

Section 3. Analysis of the model

# Talk about the attestation generation code and the calculate\_trust function

Section 4: How it could work in practice

One of the major requirements to bootstrap this algorithm is the initial group of trusted sources. A dishonest actor in the initial group of trusted sources could invalidate the resulting confidence calculations. For robustness, a larger group of trusted sources that cannot collude is ideal. In the case of ‘Smart Contract Safety Score’ the initial group of sources could be established auditors and educators. Auditors would be asked to make a public attestation about the safety of a smart contract. In addition, they may propagate confidence in the ability of auditing smart contracts to employees or open source contributors. This is shown in the diagram below. Educators for example can attest to the ability of students that graduate from the program. This network of trusted can then attest to the safety of various smart contracts and continue to grow the network. The result of this system would be a dynamic confidence score of smart contracts that can be easily accessed via EAS, and continuously be updated as contracts are upgraded, bugs are found, and public sentiment changes.

# Make diagram

Section 4b: Personalized confidence scores

On one hand, it is possible to have only have public attestations of a smart contract included in the algorithm. However, if someone would like to include non-public attestations, that EAS is capable of, or if they would like to give greater weight to some trusted sources that the general public disagrees on, they may be able to adjust those parameters easily, and run this algorithm themselves or in a private cloud. We may also provide this as a paid service. The end result would be a network with custom confidence scores