Cerberus
Towards Zero-Knowledge Advertising

Omnia Protocol

Josh Fourie, josh@omniaprotocol.com

December, 2018

Abstract

This paper explores the prospect of a basic advertising scheme nested within applications secured by Omnia Protocol through aligning zero-knowledge Succinct ARguments of Knowledge (zk-SNARKs), decentralised transaction networks and Secret Handshakes.

1 Background

The Omnia Protocol minimises the imperative for private-sector firms to extract and then secure individual data by substituting the trusted transfer of *data* between firms and individuals for the transfer of *verifiable computations* amongst untrusted parties. We achieve this through deploying basic computations locally to individual devices tasked with iterating over the data-sets accessible through smart-phones and then producing a zk-SNARKs proof affirming the correctness of a shareable and non-identifying output with only a negligible risk of failure (0.4%).

Currently, the value of data is famously extracted and realised in advertising schemes that construct a digital profile from data collected whilst users engage with a service provided, in most cases, as a free product simultaneously offering compensation for the data. A sufficiently sophisticated scheme might enable a client to prove they have viewed or engaged with an advertisement targeted at their group or demographic and thereby nullify the requirement for intermediate firms to handle and secure client data as well as expand the available data for improved advertising. The Omnia Protocol: Cerberus project intends to represent a simple step towards that goal.

The paper will consider a basic technical model wrappable within existing pilot proposals for *Employee* and *Student* Wellness that establishes and then develops the core engine as well as outlines future work and challenges for the Cerberus project. It assumes a general familiarity with zk-SNARKs and decentralised networks.

Requirements. Cerberus should achieve the following amongst the Client ($\mathcal{P}$), the Advertiser ($\mathcal{V}$), and an Adversary ($\mathcal{A}$):

- $\mathcal{P}$ cannot falsely substantiate a claim to to maliciously view an advertisement, or generate a false proof of engagement.
- $\mathcal{P}$ can guarantee remuneration for every advertisement served from $\mathcal{V}$.
- $\mathcal{A}$ cannot identify individual data-inputs or leverage any other attack vectors to harm or otherwise exploit either $\mathcal{P}$ or $\mathcal{V}$.
- Compromising Cerberus should not endanger individual data or incur substantial costs to either $\mathcal{P}$ or $\mathcal{V}$.

Note that these should be guaranteed to a negligible probability for *security* and *verifiability.*

2 System Outline

The client ($\mathcal{P}$) executes a computation determining their *candidacy* for engaging with a particular subset of advertisements that are made available on either a public repository or through a direct communication scheme hosted by the advertiser ($\mathcal{V}$) . $\mathcal{P}$ exchanges a *proof of candidacy* (PoC) for a package of relevant advertisements (the AdPack) that are then loaded onto their personal device, where a second layer of the proving scheme constructs a *proof of engagement* (PoE) that serves as a request for remuneration exchangeable with either $\mathcal{V}$ or an intermediary vendor.

Cerberus thereby consists of four core stages encompassed within two layers of communication between $\mathcal{P}$ and $\mathcal{V}$:

1. Proof of Candidacy and the exchange of the AdPack.
2. Proof of Engagement and remuneration.

Note that $\mathcal{P}$ takes an active role that $\mathcal{V}$ is charged with reactively reciprocating.

Acronyms. The reader should use this part as a quick-reference for the acronyms employed in the remainder of the paper:

· $\mathcal{P}$: The prover role assumed by the *client*.

· $\mathcal{V}$: The verifier role assumed by the *advertiser*.

· $\mathcal{A}$: The adversary attempting to break the system as neither a prover or verifier.

· PoC: Proof of Candidacy.

· PoE: Proof of Engagement.

· AdPack: A package of advertisements.

· Orb: Any computation verifiable with zk-SNARKs.

· zk-SNARKs: zero-knowledge Succinct ARguments of Knowledge.

· qap: Quadratic Arithmetic Program proposed in 2013 by Gennaro et al.

3 Proving Candidacy for the Advertisement

The *candidacy* component should prevent any malicious $\mathcal{P}$ from obtaining an AdPack whose requirements they do not meet within a negligible probability as a byway of ensuring that advertisements are served to the target demographic specified by $\mathcal{V}$. There are two alternate schemes capable of providing such a functionality that may be implemented based, in the first case, on extending the existing zk-SNARKs scheme to include the PoC computation, and in the second, on constructing a secret handshake scheme recently published in the academic literature.

3.1 A zk-SNARKs Implementation

A zk-SNARKs proofing scheme following the construction provided by Jens Groth in 2016 (*'On the Size of Pairing-based Non-interactive Arguments'*) enables $\mathcal{P}$ to prove the correctness of a statement for some relation $\mathcal{R}$ with a negligible probability of a malicious party convincing $\mathcal{V}$ of the correctness of such a statement for arbitrary computations. There are a collection of existing tools of varying production readiness that would enable Omnia Protocol to implement the zk-SNARKs scheme.

Existing Libraries. Republic Protocol[1] and Z-Cash Hackworks[2] have both provided open-source libraries realising zk-SNARKs in `rust-lang`, but note that both are currently either non-production ready or still R&D. We have successfully implemented a bare-bones test-API for generating a verifiable *Health-Rating* from a computation iterating over data collected on smart-phones using Republic's `zkSNARK-rs` library and have not explored Hackwork's `Bellman`. The 8_Bit Comparator and Keccak-hashing features introduced in `zkSNARK-rs` are currently facing independent issues with the former returning only a 77% correctness on verification and the latter hanging on the generation of a *quadratic arithmetic program* from a dummy representation of the circuit - Cerberus relies on both these features for candidacy.[3] [4]

3.1.1 Zero-Knowledge 8-Bit Comparator

The PoC requires that $\mathcal{P}$ proves membership of a group manifesting a set of characteristics (C_{char}) defined by $\mathcal{V}$ as being within a desired *range*, denoted as $\mathcal{P} \in C_{char}$. The comparator is capable of proving that $\gamma > \alpha$ or $\gamma < \beta$ packaged within `zkSNARK-rs` as a `test_program`, which provides the fundamental component for proving that γ belongs to a range such that $\alpha < \gamma < \beta$. The library provides a circuit for constructing the qap required for a single 8-bit comparator parsed in the `.zk` program, but this does not extend to a range, which inherently requires at least two-comparators. Moreover, membership of a target group would, in most instances, require $\mathcal{P}$ to prove that they possess characteristics nested within multiple distinct ranges. Currently, the comparator implementation raises the following in the context of the Cerberus instantiation:

- The comparator must be extended to prove that $\mathcal{P}$ knows *many smaller* weights, such that when computations determining characteristics from the individual data-set are executed and collated, $\mathcal{P} \in C_{char}$.

[1] https://github.com/republicprotocol/zksnark-rs/tree/develop

[2] https://github.com/zkcrypto/bellman

[3] Note that this is at the time of writing and may not be relevant or up-to-date at the time of reading.

[4] The `zkSNARK-rs` comparator was tested with a loop recursively proving and verifying the same inputs.