CS 189A Vision Statement

Team Name: Logos

Team Sponsor: Veridise

Members:

Shivansh Kapoor (shivaansh@ucsb.edu)
Colter Sirlin (csirlin@ucsb.edu)
Thomas Hale (thomashale@ucsb.edu) - scribe
Christopher Chang (christopherchang@ucsb.edu)
Maya Ma (yema@ucsb.edu)

Motivation

Zk proofs, or zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge), tackle foundational challenges in the blockchain arena, particularly those concerning privacy and scalability. Inherent to blockchain's design is its transparency, where every transaction is logged on a public ledger, potentially compromising user privacy. Zk proofs offer a remedy by enabling transaction validation without disclosing the transaction's specifics, thus preserving data confidentiality. Additionally, as blockchain networks expand, the individual verification of each transaction can become a limiting factor, impeding scalability. Zk proofs address this by permitting the aggregation of multiple transactions into a single, succinct proof that can be swiftly verified. However, while zk proofs offer these advantages, they introduce a new challenge: ensuring the security and correctness of zk proof circuits. Vulnerabilities in these circuits can compromise the integrity of the entire system. Detecting and rectifying these vulnerabilities is paramount, as even a minor flaw can lead to significant breaches, undermining the trust and reliability that blockchains aim to establish.

Veridise, a leading company in the blockchain space, specializes in auditing ZK circuits and smart contracts to ensure their security and robustness by examining these circuits for potential vulnerabilities. However, a significant challenge that Veridise faces is the abundance of frameworks available for writing zk circuits. Each framework comes with its own nuances, structures, and potential pitfalls. This diversity makes it exceedingly challenging to develop security analysis tools that are both comprehensive and universally applicable. A tool optimized for one framework might not be directly transferable to another, necessitating the creation of multiple specialized tools or the constant adaptation of existing ones. This fragmentation in the zk circuit frameworks complicates the auditing process and underscores the need for more standardized or generalized approaches in the industry.

Solution

Common Intermediate Representation (IR) Development
In response to the challenges presented by the diverse zk circuit frameworks, our first solution revolves around the creation of a common Intermediate Representation (IR) for all frameworks. This web application will serve as a platform, enabling users to seamlessly translate and integrate various zk circuit frameworks. By introducing a unifying IR, we aim to bridge the disparities between different frameworks, fostering a more cohesive environment for security analysis. As a starting point, our web application will support translations for Circom and will be followed by Halo2, Artworks, Gnark, and Plonky2. This initiative not only streamlines the development process but also ensures a standardized approach to zk circuit creation and validation.

Circuit Visualization for Debugging
Complementing our IR-centric solution, we are also introducing a state-of-the-art circuit visualization tool within our SaaS web application. Recognizing the complexities and intricacies of zk circuits, this tool is designed to provide developers with a clear, interactive representation of their circuits, facilitating efficient debugging and optimization. Given the inherent requirement of an IR for effective visualization, this tool synergistically aligns with our first solution, highlighting the interconnected nature of our projects. Users can effortlessly upload their zk circuits, regardless of the originating framework, and leverage the visualization tool to gain insights, identify bottlenecks, and rectify potential vulnerabilities. This tool not only enhances the debugging process but also empowers developers with a deeper understanding of their zk circuit designs.

Goals

- Plan an intermediate representation that can capture the behavior of all frameworks.
    - Work with Veridise to determine the constructs and organization they would like to see.
    - Hopefully, looking at the commonalities between frameworks will allow us to develop an IR that is extensible to other frameworks we are not working with.
- Develop a compiler to convert ZK circuits from each framework into the intermediate representation syntax
    - Share as much of the logic between frameworks as possible
- Create visualization tool for ZK circuits represented in our IR
- Create a publicly available web app that interacts with visualization code
    - Text box to paste/type and convert within the web app
    - Stretch goal: you can hover over an expression and it will highlight the converted expression
        - Ex: https://kampfkarren.github.io/full-moon-viewer/

Non-Goals

- Decompiling machine code/byte code to our IR
- Blockchain coding
- Performing the actual contract audits

Implementation

Circom -> IR

Circom is a language by itself with its own parser we can leverage to transpile to our own IR.

Frameworks -> IR

We can build upon the frameworks' internals to generate our IR as a side effect.

Web App

We will use React to display the IR. Users will be able to paste in their code to the app and it will emit the transpiled code. We will leverage AWS lambda with API gateway to transpile the codes on the cloud. Alternatively, we can run the transpiler locally with webassembly.

Technologies

- Rust
- AWS
- React
- Circom
- Halo2
- Gnark
- Arkworks
- Plonky

Resources

Background knowledge resources provided by Veridise:
https://veridise.notion.site/veridise/UCSB-Capstone-Resources-35f31fc8c032451993b06ce975cefe11