



Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts

Yashar Dehkan Asl

Chapter I

Introduction

Decentralized Cryptocurrencies:

Such as Bitcoin and altcoins are getting more popular

Blockchain Technology:

These emerging cryptocurrency systems build atop a novel blockchain technology where miners run distributed consensus whose security is ensured

Trusted Clock:

A trusted clock is needed that increment whenever a new block is mined, and it is crucial for attaining financial fairness protocol.

Lack of Privacy:

The present form of Blockchain can be trusted for correctness and availability but not privacy

Hawk Overview

Hawk: A framework for building privacy-preserving smart contracts.

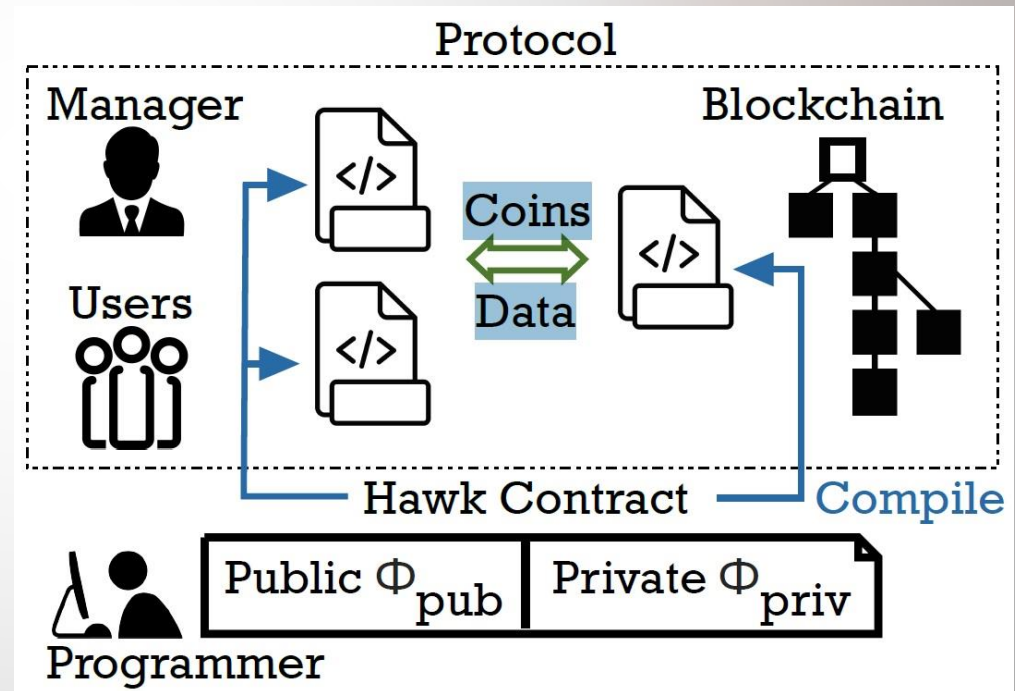
It compiles the program to a cryptographic protocol between the blockchain and the user.

Private portion:

Performs computation to determine the payout distribution amongst the parties

Public Portion:

Does not touch private data or money



Cont.

Security guarantees

- **On-chain privacy**
transactional privacy is provided against the public
- **Contractual security**
contractual security protects parties in the same contractual agreement from each other

Minimally trusted manager

The execution of Hawk contracts are facilitated by a special party called the manager

Example: Sealed Auction

Example program: A sealed auction where the highest bidder wins, but pays the second highest price

Contractual security requirements: Hawk will compile this to cryptographic protocol, and also guarantees the following requirements for parties

- **Input independent privacy**
- **Posterior privacy**
- **Financial fairness**
- **Security against a dishonest manager**

Cont.

Aborting and timeouts

Three timeouts are declared where $T1 < T2 < T3$:

T1 : The Hawk contract stops collecting bids after T1.

T2 : All users should have opened their bids to the manager within T2.

T3 : If the manager aborts, users can reclaim their private bids after time T3.

Contributions

Formal models for decentralized smart contracts

Universal Composability (UC) is presented for blockchain model of cryptography. It rely on a notion called *wrappers*. Wrappers handle a set of common details such as timers, pseudonyms, global ledgers in a centralized place such that they need not be repeated in every protocol.

New cryptography suite

A new cryptography suite that binds private transactions with programmable logic. Our protocol suite contains three essential primitives *freeze*, *compute*, and *finalize*.

Chapter II

The Blockchain Model of Cryptography

The Blockchain Model

Blockchain refers to a decentralized set of miners who run a secure consensus protocol to agree upon the global state. The blockchain not only maintains a global ledger that stores the balance for every pseudonym, but also executes user-defined programs.

- **Time**
- **Public state**
- **Message delivery**
- **Pseudonyms**
- **Correctness and availability**

Cont.

Programs, wrappers and functionalities

The users, the manager and the programs are referred to the ideal program, the blockchain and the user/manager program respectively.

- **The ideal wrapper**
- **The blockchain wrapper**
- **The protocol wrapper**

** wrappers implement a set of common features needed by every smart contract application. In this way, we need not repeat this notation for every blockchain application.*

Conventions for Writing Programs

Timer activation points

Implement a clock that advances in rounds

Delay processing in ideal program

Delayed processing can be implemented simply by storing state and invoking the delayed program instructions on the next Timer click. upon the next timer click, the delayed instructions are executed first.

Chapter III

Cryptography Abstraction

Overview: Hawk realizes the following specifications:

Private ledger and currency transfer

Hawk relies on the existence of a private ledger that supports private currency transfers.

Hawk-specific primitives

We define Hawk-specific primitives including freeze, compute, and finalize that are essential for enabling transactional privacy and programmability simultaneously.

Private Cash Specification

At a high-level, private ledger and currency transfer is needed.

Mint: The mint operation allows a user to transfer money from the public ledger to the private pool.

Pour: The pour operation allows a user to spend money in its private bank privately.

Privacy: It does not learn which coin in the private pool is being spent nor the name of the spender. Therefore, the spent coins are anonymous with respect to the private pool.

Additional subtleties:

Whenever an honest party pours, it first checks if an appropriate coin exists in its local wallet, and if so it immediately removes the coin from the wallet.

Hawk Specification

We describe the specifications of new Hawk primitives, including freeze, compute, and finalize.

Freeze. In freeze, a party removes one coin from the private coins pool, and freeze it in the blockchain by adding it to FrozenCoins

Compute. When a party calls compute, its private input and the value of its frozen coin are disclosed to the manager

Finalize. In finalize, the manager submits a public input. now computes the outcome of all parties' inputs and frozen coin values, and redistributes the FrozenCoins based on the outcome.

Cont.

Interaction with public contract

The public contract pub typically serves the following purposes:

- **Check the well-formedness of the manager's input**
- **Redistribute public deposits**

Security and privacy requirements

Timing and aborts

Chapter IV

Cryptographic Protocols

Our protocols are broken down into two parts:

1. The private cash part that implements direct money transfers between users
2. The Hawk-specific part that binds transactional privacy with programmable logic.

Warmup: Private Cash and Money Transfer

During a pour operation, the spender chooses two coins to spend. The pour operation pays amounts to two output pseudonyms.

Existence of coins being spent

No double spending

Money conservation

Technical subtleties

Binding Privacy and Programmable Logic

Freeze: does not spend directly to a user, but commits the money as well as an accompanying private input to a smart contract.

Compute: computation takes place off-chain to compute the payout distribution and a proof of correctness

Finalize: Blockchainhawk verifies the proof and redistributes the frozen money accordingly

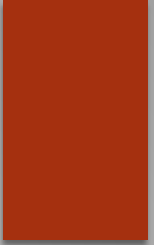
Extensions and Discussions

Refunding frozen coins to users

Instantiating the manager with trusted hardware


Pouring anonymously to long-lived pseudonyms

Open enrollment of pseudonyms



Chapter V

ADOPTING SNARKS IN UC
PROTOCOLS AND PRACTICAL
OPTIMIZATIONS



Using SNARKs in UC Protocols

Succinct Non-interactive ARguments of Knowledge (SNARK) provide succinct proofs for general computation tasks.

SNARK's security is too weak to be directly employed in UC protocols

cannot be used by the UC simulator

Practical Considerations

To achieve efficiency, optimized circuit is designed in two ways:

1. Using cryptographic primitives
2. Building customized circuit generators

Cont.

Optimizations for finalize

Two key observations allow us to greatly improve the performance of the proof generation during finalize

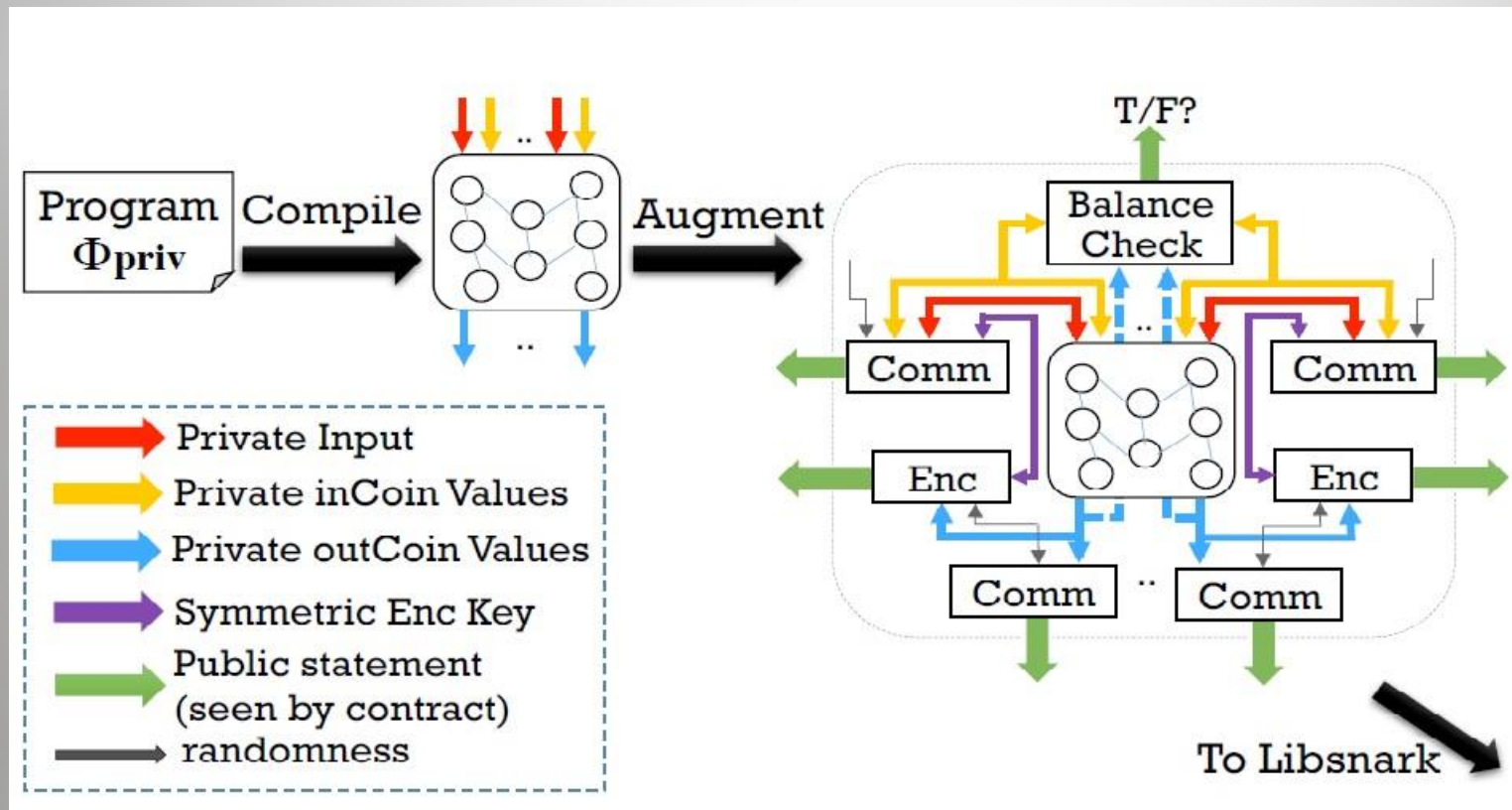
Optimization 1: *Minimize SSE-secure NIZKs*

Optimization 2: *Minimize public-key encryption in SNARKs.*

Remarks about the common reference string

Chapter VI

IMPLEMENTATION AND EVALUATION



Compiler Implementation

compiler consists of several steps,

Preprocessing

Circuit Augmentation

Cryptographic Protocol

Additional Examples

Crowdfunding

Rock Paper Scissors

“Swap” Financial Instrument