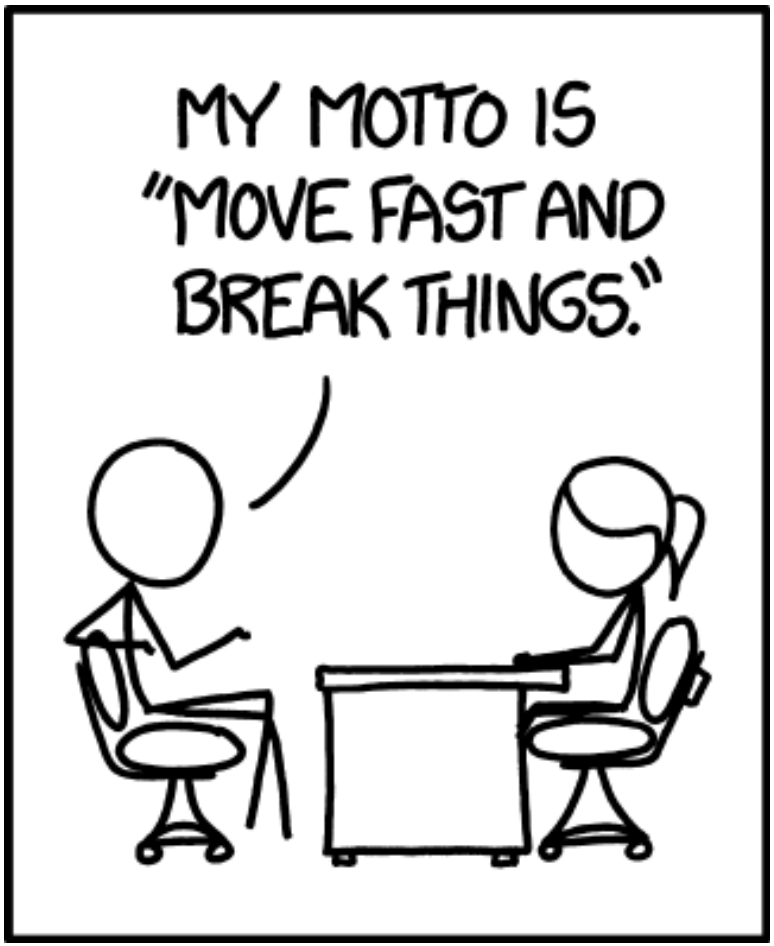
Findel: Secure Derivative Contracts for Ethereum

Alex Biryukov Dmitry Khovratovich
Sergei Tikhomirov

1st Workshop on Trusted Smart Contracts
In Association with Financial Cryptography 17
7 April 2017, Malta

... and Smart Contract Developer



Adapted from xkcd.com/1428

JOBS I'VE BEEN FIRED FROM

FEDEX DRIVER
CRANE OPERATOR
SURGEON
AIR TRAFFIC CONTROLLER
PHARMACIST
MUSEUM CURATOR
WAITER
DOG WALKER
OIL TANKER CAPTAIN
VIOLINIST
MARS ROVER DRIVER
MASSAGE THERAPIST

Contract Programming is Different

- "Move fast and break things": unacceptable!
- Real money (property, resources) at stake
- Side-effects often cause trouble (The DAO)
- We need a formally verifiable contract language

A Secure Financial DSL Helps

- Avoid misinterpretation
- Standardize templates
- Prove correctness
- Facilitate automated processing

Composable Contracts [Peyton Jones et al. 2003]

- Ten primitives to compose complex agreements
- Declarative paradigm
- Implemented as an embedded DSL in Haskell

Composable Contracts [Peyton Jones et al. 2003]

- Ten primitives to compose complex agreements
- Declarative paradigm
- Implemented as an embedded DSL in Haskell

Example: zero-coupon bond

when (at 2018-01-01) (scale (konst 100)) (one $))

Ethereum

- Turing-complete virtual machine
- Key feature: trustless execution
- Perfect match for financial agreements!

Ethereum Meets Composable Contracts

- Map declarative DSL to blockchain execution paradigm
- Retrieve and validate external data
- Ensure that execution cost is bearable

Financial **De**rivatives **L**anguage (Findel)

- Contract: agreement between *issuer* and *owner*
- Contract *description* defines rights and obligations
- Description is a tree of *primitives*
- Description and issuer are immutable
- Ownership may be transferred
- Smart contract acts as execution environment

Findel Primitives 1/2

- *Zero* – do nothing

Findel Primitives 1/2

- *Zero* – do nothing
- *One* – transfer 1 unit of currency from issuer to owner

Findel Primitives 1/2

- *Zero* – do nothing
- *One* – transfer 1 unit of currency from issuer to owner
- $Scale(k, c)$ – multiply all payments by a constant value

Findel Primitives 1/2

- *Zero* – do nothing
- *One* – transfer 1 unit of currency from issuer to owner
- $Scale(k, c)$ – multiply all payments by a constant value
- $ScaleObs(obs, c)$ – multiply all payments by an observable value (think exchange rate)

Findel Primitives 1/2

- *Zero* – do nothing
- *One* – transfer 1 unit of currency from issuer to owner
- $Scale(k, c)$ – multiply all payments by a constant value
- $ScaleObs(obs, c)$ – multiply all payments by an observable value (think exchange rate)
- $Give(c)$ – swap parties

Findel Primitives 2/2

- $And(c_1, c_2)$ – execute both sub-contracts

Findel Primitives 2/2

- $And(c_1, c_2)$ – execute both sub-contracts
- $Or(c_1, c_2)$ – owner chooses which sub-contract to execute

Findel Primitives 2/2

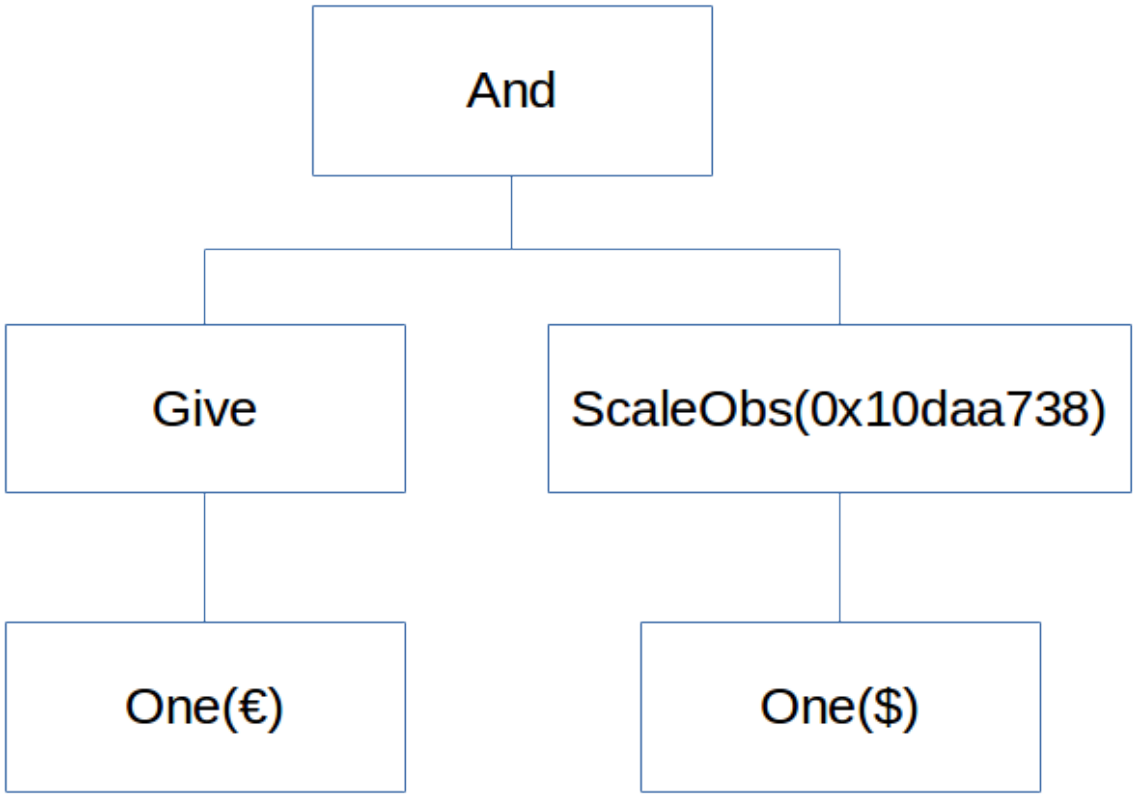
- $And(c_1, c_2)$ – execute both sub-contracts
- $Or(c_1, c_2)$ – owner chooses which sub-contract to execute
- $If(obs, c_1, c_2)$ – execute one of sub-contracts (depending on observable)

Findel Primitives 2/2

- $And(c_1, c_2)$ – execute both sub-contracts
- $Or(c_1, c_2)$ – owner chooses which sub-contract to execute
- $If(obs, c_1, c_2)$ – execute one of sub-contracts (depending on observable)
- $Timebound(t_0, t_1, c)$ – execute c, if within time bounds

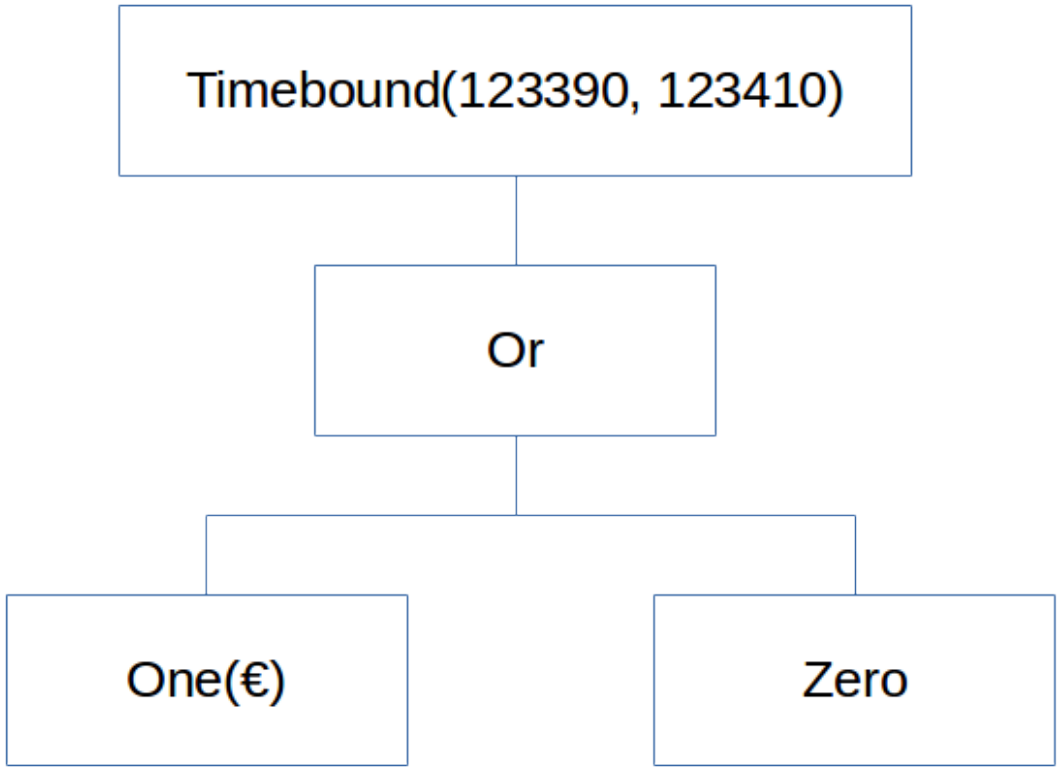
Example 1/3: Currency Exchange

$And(Give(One(EUR)), ScaleObs(exchAddr, One(USD)))$



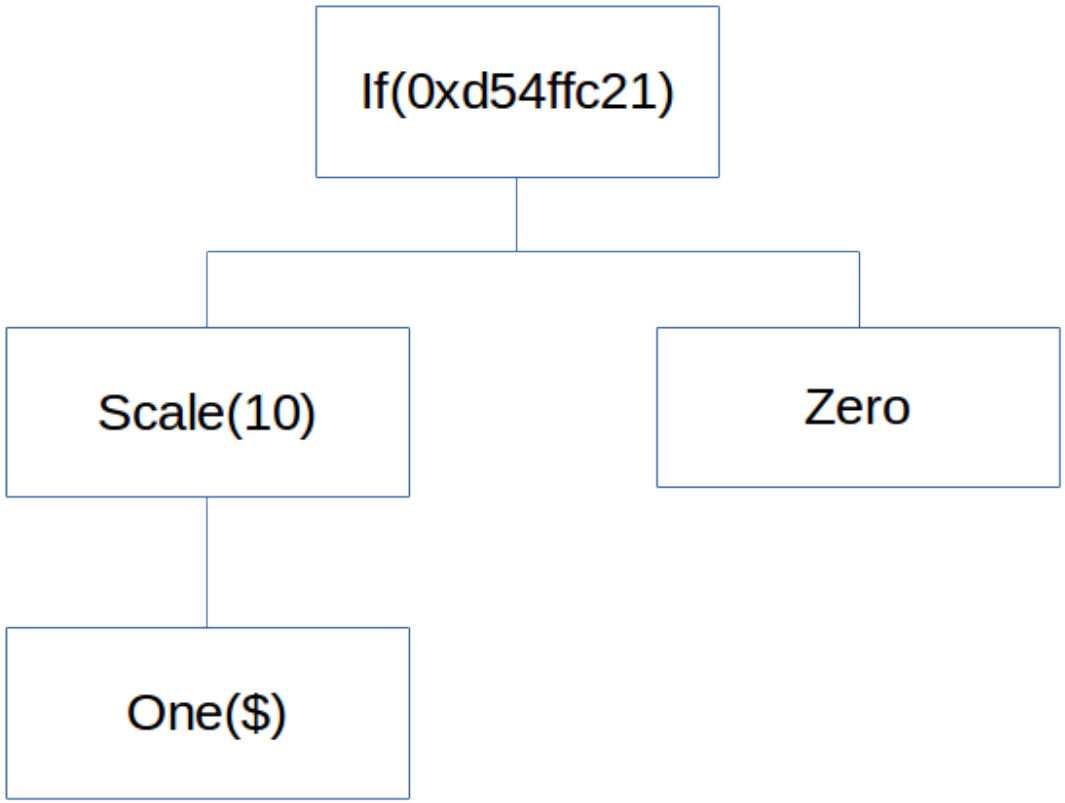
Example 2/3: European Option

$$Timebound(t_0 - \delta, t_0 + \delta, Or(One(EUR), Zero))$$



Example 3/3: Binary Option

$$If(orclAddr, Scale(10, One(USD)), Zero)$$



External Data Problem

- Contracts require external data
- Ethereum is isolated from broader Internet

External Data Problem

- Contracts require external data
- Ethereum is isolated from broader Internet
- Our solution: *gateways*

Gateways

- Smart contract pulls and stores external data with timestamp
- Optional proof of authenticity (e.g., signature under known public key)
- Findel marketplace queries gateway when needed
- Parties are responsible for updating gateways

What Is Done

- Defined a declarative DSL suited for blockchain
- Implemented a marketplace smart contract in Solidity
- Assessed cost of operation ($\sim$ \$0.1 per avg operation)[1]

[1] As of Jan 2017 at \$10 / ether

Future Work

- Enforcement: deal with users defaulting on debt
- Model balances using ERC20 Token
- Extend model to support multi-party contracts
- Look into valuation and verification

Questions?

- cryptolux.org
- @serg_tikhomirov
- sergei.tikhomirov@uni.lu