Obsidian
A Safer Blockchain Programming Language

What is a blockchain?
• Blockchain programming environments provide shared, global state on untrusted, distributed computing nodes.
• Global state consists of smart contracts, which include both data structures and transactions that manipulate them.
• Transactions can deploy smart contracts to the blockchain (initializing their state), or invoke code implemented in specific deployed smart contracts.
• “Code is law”: a principle that suggests that a contract’s code specifies an agreement between parties. This principle implies that contracts are immutable; bugs in them cannot be fixed after deployment.

Key-value store with state of all contracts

The nodes execute a consensus protocol by which they agree on which transactions have been processed and in what order. After consensus is reached, all nodes agree on the transactions and their ordering.

One blockchain network node

| transaction 1 | transaction 2 | transaction 3 | transaction 4 | transaction 5 | transaction 6 | transaction 7 | transaction 8 |

A 3-block blockchain

| transaction 1 | transaction 2 | transaction 3 | transaction 4 | transaction 5 | transaction 6 | transaction 7 | transaction 8 |

Blockchain facts | Consequences | Design approach

- Smart contracts can hold resources.
- Bugs can transfer resources to the wrong party or lose them entirely. A bug was recently exploited to steal over US$40M from a contract.
- Obsidian models resources with linear types, which statically restrict lifecycles, so that resources cannot be accidentally lost.
- Smart contracts are immutable once deployed, but this means that bugs are not fixable.
- Support developer-authored specifications and use verification tools to prove that the code satisfies the specifications. Use a strong, static type system to detect as many bugs as possible at compile time.

Modifying a contract's code could change an agreement that parties have settled on.

Proposed application domains, such as finance and medical records, cannot tolerate serious bugs.

Correct software is paramount.

Blockchain applications commonly implement state machines. Behavior and available transactions depend on the current state.

Obsidian is a typestate-oriented language, representing state in types and statically preventing some invalid invocations. For example, a Bond that has been bought is in the Bought state and cannot be bought again.

Status and Upcoming Work

- An initial compiler translates Obsidian code to Java (for execution on a mock blockchain) and to Dafny (for verification).
- The compiler includes a typechecker based on a draft typesystem.
- We plan to port Obsidian to Hyperledger Fabric and complete the compiler.
- Our evaluation will ask users to write programs in Obsidian and either Hyperledger Composer or Solidity (existing blockchain development tools) and compare task completion times and rates and kinds of bugs.
- We are iteratively evaluating portions of the language design in the context of Java so that we can isolate those parts of the design for study.

Annotated example code

```
interface account {
  transaction pay (money m) {
  }
}

contract Bond {
  account seller;
  Bond(account s) {
    seller = s;
  }
  transaction badTransaction() {
    buyer.pay(m);
  }
}

contract ErroneousClient {
  transaction badTransaction() {
    Bond.Offered b = new Bond(...);
    b.buy(...);
    buyer.pay(m);
  }
}
```

Michael Coblenz mcoblenz@cs.cmu.edu
Eliezer Kanal ekanal@cert.org
Obsidian: A Safer Blockchain Programming Language