Formal Verification in Scala

Ramon Boss, Anna Doukmak

June 11, 2019

Table of Contents

Table of Contents

Stainless

- Pure Scala code
- Imperative features
- Formal specification with pre- and postcondition
- Outcomes of verification: valid, invalid, unknown

Properties to verify

- Bitcoin-S project
- No Inflation Property
- Addition with Zero Property

Table of Contents

Verifying No Inflation Property

- Validation of a transaction with *checkTransaction*
- Integration of Stainless
- Adjusting the property
- Bug in *checkTransaction*

Verifying No Inflation Property

- Validation of a transaction with *checkTransaction*
- Integration of Stainless
- Adjusting the property
- Bug in *checkTransaction*

```
1   val prevOutputTxIds = transaction.inputs.map(_.
        previousOutput.txId)
2   val noDuplicateInputs = prevOutputTxIds.distinct.
        size == prevOutputTxIds.size
```

Bugfix

6 core-test/src/test/scala/org/bitcoins/core/protocol/transaction/TransactionTest.scala

```
@@ -294,6 +294,12 @@ class TransactionTest extends FlatSpec with MustMatchers {
294 294         }
295 295       }
296 296
297     +   it must "check transaction with two out point referencing the same tx with different indexes" in {
298     +     val hex = "0200000002924942b0b7c12ece0dc8100d74a1cd29acd6cfc60698bfc3f07d83890eec20b6000000006a47304402202831
299     +     val btx = BaseTransaction.fromHex(hex)
300     +     ScriptInterpreter.checkTransaction(btx) must be(true)
301     +   }
302     +
297 303     private def findInput(
298 304         tx: Transaction,
299 305         outPoint: TransactionOutPoint): Option[(TransactionInput, Int)] = {
```

4 core/src/main/scala/org/bitcoins/core/script/interpreter/ScriptInterpreter.scala

```
@@ -779,8 +779,8 @@ sealed abstract class ScriptInterpreter {
779 779       val totalSpentByOutputs: CurrencyUnit =
780 780         outputValues.fold(CurrencyUnits.zero)(_ + _)
781 781       val allOutputsValidMoneyRange = validMoneyRange(totalSpentByOutputs)
782     -     val prevOutputTxIds = transaction.inputs.map(_.previousOutput.txId)
783     -     val noDuplicateInputs = prevOutputTxIds.distinct.size == prevOutputTxIds.size
    782 +     val prevOutputs = transaction.inputs.map(_.previousOutput)
    783 +     val noDuplicateInputs = prevOutputs.distinct.size == prevOutputs.size
784 784
785 785       val isValidScriptSigForCoinbaseTx = transaction.isCoinbase match {
786 786         case true =>
```

Verifying Addition with Zero

- Datatype Satoshis
- Method for verification

```
+(c: CurrencyUnit): CurrencyUnit
```

- Rewriting 2 classes into Pure Scala

Rewriting the code

Formal specification

```
1   override def +(c: CurrencyUnit): CurrencyUnit = {
2     require(c.satoshis == Satoshis.zero)
3     Satoshis(satoshis.underlying + c.satoshis.
          underlying)
4   } ensuring(res => res.satoshis == this.satoshis)
```

Table of Contents

Output of Stainless