STRUCTURING MULTI TRANSACTION CONTRACTS IN BITCOIN

Jeremy Rubin
Is Cash Bitcoin’s Killer App?

• Ethereum does smart contracts!
  – like the DAO

* Betteridge’s Law
Safe Contract Extensions for Bitcoin Contracts

• Tools for complex contracts
• Avoid internal complexity
We Can Have It All

• Transaction level invariants called Covenants can get us there
Contributions

• Extensions to Covenants
• Merkle Compressed Covenants
• Transaction Diagrams
• Multi-Phase Execution Techniques
Covenant Contracts

BACKGROUND KNOWLEDGE
The Naughty Banker

• You ask Bob’s Bank to hold a $100 deposit
  – Bob buys himself some new sneakers with your money
Covenant Contracts

• Contracts that **REQUIRE** creation of a contract of certain form
• Example: Hiring a Banker
  – They have your money but only you can withdraw!
Placeholder Notation

• high level:
  – COV(plain English invariant)

• script:
  – <plain English invariant> OP_COV

• Examples:
  – COV(Bob’s Bank only lets me withdraw)
  – <Bob’s Bank only lets me withdraw> OP_COV
Making Covenants

• Two Major Variant
  – Invariant by Execution
    • OP_COV[MES16] “Introspective”
      – Pattern matching
    • “Computational” 😞 impractical
  – Invariant by Construction
    • Recovered PubKey “Cryptographic”
      – OP_CHECKSIGFROMSTACK + OP_CAT/SUBSTR
      – SIGHASH_MASK-ing
    • MultiSig “Trustful”
      – 1 of N honesty
Grave Concerns

- **Fungibility & Privacy**
  - Forced Compliance

- **Computational Explosion**
  - Loops could make Turing Complete

- **Open Topic**: Expressive & Safe contracts without covenants?
  - Like preventing Turing Completeness...
    - \{CSS, MOV, C++ Templates...\} are Turing Complete
  - Trivial Bitcoin “covenants”
    - \text{sum}(	ext{Outputs}) \leq \text{sum}(	ext{Inputs}), \text{inputs exist}, \text{etc}...
extensions to COVENANTS
Tale of Expired Accounts

• Let’s say you have a phone number as 2FA to your bank account
  – When you change your number, you want the 2FA to change too
  – In fact, you want to not be able to change your phone before your 2FA points to the new record
Input-Join Covenant

• Two outputs forced to be consumed in one transaction
• Execution or Construction based implementations

✓ Minimal Bitcoin Extensions needed
Two Cars Problem

- You need One car at noon
  - You have a Ferrari and a Porsche
  - You want your (really good) friend to borrow one car at noon but not the one that you want
  - “Only after I have chosen should you be able to drive away”
Impossible Input Covenant

• Prove an input creation impossible
  1. Prove an input was already consumed
  2. Construct input from consumed input
  3. 1 & 2 Prove input creation impossible

• “Constructive” without extension, “Introspective” with new OpCodes
  – Consuming output exclusively made in a branch equivalent to chain introspection
  – Must be constructed ahead of time
Bad Airlines

• You’re flying from JFK to SFO with a layover in ORD
  – You go JFK → ORD
  – ORD gets snowed in
  – You’re stuck in the snow

• How can we ensure JFK→SFO next time?
Intermediate Output Covenant

- `<i> OP_IS_IUTXO requires output at index `<i>` be spent in same block`
- Bad for two-phase-commit protocols
  - Except *between* commits
- Complicates block-creation code
  - Child-Pays-For-Parent similar
Bad Airlines (Part 2)

- You book your own transfers, avoiding ORD and other cold airports
  - When you get to the airport you realize the first check-in won’t give you all your tickets, you need to go through security twice
Virtual Output Covenant

- `<s> <i>` **OP_SIG_VUTXO** requires that output at `<i>` redeemable with `<s>`
- `<i> **OP_IS_VUTXO** requires some other input script provide proof
  - allows optimizing malleating provers…
- Same goal as **OP_IS_IUTXO**
  - No multiple transactions
  - No mining complexity
  - Additional Signing complexity
- Could permit shared-stack
  - Through alt-stack?
- Safe “Turing Complete” recursion?
  - $\Delta_0$, Russell's Post Theorem Trace Witness
  - You can tweet me too…
    “Imma let you finish but VUTXO is the best $\Delta_0$ – @JeremyRubin”
application

MERKLE COVENANTS
Compressed Contracts

• Summarize useless clause in contract

• Example: Appendix A
  – Provide your Tax Payer ID here ____
  – See Appendix A if no Tax Payer ID
MAST: Merkelized Abstract Syntax Tree

• $O(\log(n))$ branch elimination compression
• Huffman Codable
• Example:

Compile

```
if (A) {T} else {F}
```
to

```
assert(H(code)==(A ? H(T) : H(F)));
eval(code);
```
Bitcoin Implementation (Proposal)

- Put all branches into a tree & run
- Example
  - if (A){if (B){C} else {D}} else {E}
  - Merkle Tree of {`assert(A&&B);C`, `assert(A&&~B);D`, `assert(~A);E`}
  - Prove branch in the tree, then run
- One Input inside one Transaction
Properties

• Atomic Execution
  – No intermediate state
• Minimal Hash “Overhead”
  – 1 Hash/Pruned Branch, 1 Parent Hash
• No need to reveal not-taken branch
Conditional Covenant

• Make an Output as follows
  – scriptpubkey:
    OP_IF
      <output 0 = A w/ 1 satoshi> OP_COV
    OP_ELSE
      <output 0 = B w/ 1 satoshi> OP_COV
    OP_ENDIF
Either Red Tx or Blue Tx

Transaction 1 (if Branch 1)

scriptsig: OP_TRUE
scriptpubkey: OP_IF
<output 0 = A w/ 1 satoshi>
OP_COV OP_ELSE <output 0
= B w/ 1 satoshi> OP_COV
OP_ENDIF

Transaction 1 (if Branch 2)

scriptsig: OP_FALSE
scriptpubkey: OP_IF
<output 0 = A w/ 1 satoshi>
OP_COV OP_ELSE <output 0
= B w/ 1 satoshi> OP_COV
OP_ENDIF
Red Tx; Either C or D?

Transaction 1 (if Branch 1)

\[
\text{scriptsig: OP_TRUE}
\]
\[
\text{scriptpubkey: OP_IF}
\]
\[
<\text{output 0 = A w/ 1 satoshi}>
\]
\[
\text{OP_COV} \ \text{OP_ELSE} \ <\text{output 0 = B w/ 1 satoshi}> \ \text{OP_COV}
\]
\[
\text{OP_ENDIF}
\]

Transaction 2 (if Branch 2)

\[
\text{scriptsig: OP_FALSE}
\]
\[
\text{scriptpubkey (== A):}
\]
\[
\text{OP_IF} \ <\text{output 0 = C w/ 1 satoshi}>
\]
\[
\text{OP_COV} \ \text{OP_ELSE} \ <\text{output 0 = D w/ 1 satoshi}>
\]
\[
\text{OP_COV} \ \text{OP_ENDIF}
\]
Properties

• Non-Atomic Execution Mode
  – Intermediate states allowed
  – Extra hash per branch

• Atomic Execution Mode
  – Using OP_IS_VUTXO
  – Minimal Hash “Overhead”
    • 1 Hash/Pruned Branch, 1 Parent Hash

• No need to reveal not-taken branch

• Signature Parallelization benefits

• Larger Max Script Size
TRANSACTION DIAGRAMS
Primitives: Transaction

input: A
script: “…”
scriptSig: “…”
output: B
Primitives: Output Covenant

input: A
script: COV(B)
scriptSig: “…”
output: B
Primitives:
Conditional Covenant

```
input: A
script:
OP_IF
    <output 0 = B>  OP_COV
OP_ELSE
    <output 0 = C>  OP_COV
OP_ENDIF
scriptSig:  OP_TRUE
output: B
```
Primitives: AND Covenants

input: F
script:
<output 0 = G > OP_COV
<output 1 = H> OP_COV
OP_ENDIF
scriptSig:
outputs: G, H
Inputs: A, B
Script: "..."
ScriptSig: "..."
Outputs: C
Primitives: Impossible Input Covenant (Constructive)

inputs: C, ~B
script: “…”
scriptSig: “…”
outputs: D

inputs: X
script: OP_IF
<~B, A> OP_ELSE
<B, ~A> OP_ENDIF
F OP_COV
scriptSig: “…”
outputs: ~B, A
Primitives: Impossible Input Covenant

inputs: C, ~B
script: “…”
scriptSig: “…”
outputs: D

inputs: X
script: OP_IF
<~B, A> OP_ELSE
<B, ~A> OP_ENDIF
F OP_COV
scriptSig: “…”
outputs: ~B, A
Primitives:
Impossible Input Covenant

inputs: C, \sim B
script: “…”
scriptSig: “…”
outputs: D

inputs: X
script: OP_IF
\langle \sim B, A \rangle OP_ELSE
\langle B, \sim A \rangle OP_ENDIF
F OP_COV
scriptSig: “…”
outputs: \sim B, A
Primitives: Virtual Output

input: A
script: <index(B)> OP_IS_IUTXO
scriptSig: “…”
output: B, C
MAST
Execution

COV(A) → COV(B)

COV(C) → COV(D)
A
SELECT
C → D

COV(E) → COV(F)
B
SELECT
E → F
Shorthand
Shorthand

COV(C) COV(G, H) COV(E) COV(F)
Shorthand

COV(C) COV(G,H) COV(E) COV(F)

C G H E F
Shorthand with Depth
techniques for

MULTI-PHASE EXECUTION
Stuck State

• A multi-transaction contract which is stuck at a certain branch, when other branches could have avoided the stuck state
• Transactions **CANNOT** be rolled back
Simply Non-Stuck

• Avoid contracts that may get stuck
  – Only use virtual/intermediate outputs

• Two-phase commits must be able to get stuck
Taken-Branch-Elimination Rollback

- [If after acceptable delay,] recreate all of a transaction’s input scripts without branch taken
- Finite (no looping)
- No New Opcode
- Drawback: Program Size
Taken-Branch-Elimination Rollback

COV(A)

SELECT

COV(A)

A

COV(B)

COV(C)

TIMEOUT;RE

SELECT

C

SELECT

A

A
Safe High Voltage Switching

• Ask an Electrical Engineer how they keep high voltage circuits with low voltage control separate – Optical Isolation!
Optical Isolated Contracts

• Use separate control flow for access control and value
• Impossible input covenants ensure fund usage with protocol
Optical Isolated Contracts

M₀ 1000 Satoshi (Fees only)

M₁ 100 Bitcoin

SELECT COV(INP(~F)) COV(F)

SELECT A COV(F)

F

B ~F

E INP(~F)

G
Optical Isolated Contracts

M0 1000 Satoshi (Fees only)

M1 100 Bitcoin

COV(INP(~F)) COV(F)

INP(~F)
Optical Isolated Contracts

- M0: 1000 Satoshi (Fees only)
- M1: 100 Bitcoin
- COV(INP(~F)), COV(F)
- A: COV(F)
- B: ~F
- E: INP(~F)
- G:
Optical Isolated Contracts

M0 1000 Satoshi (Fees only)  ->  SELECT -> COV(A)  -> COV(B)

M1 100 Bitcoin  -> SELECT -> COV(INP(~F))  -> COV(F)

A  COV(F)

~F

G

INP(~F)
The Deli Problem

• You want to buy some deli-meats and prepared foods
  – But the line is blocking the counter
  – Deli-number congestion control?
Congestion Control

• Suppose you have a time sensitive close operation
• Do a cheap “commit-close” txn
• More expensive close when excess bandwidth available
  – size(COV CLOSE) < size(CLOSE)
• Send both to miner, they can choose!
• Overall, more expensive, but faster
Congestion Control

\[\text{close Bob} \quad \text{SELECT} \quad \text{close Alice} \]

- RSMC
- Alice
- RSMC
- Bob
Congestion Control

COV(Close Bob)

COV(Close Alice)

select close Bob

select close Alice

RSMC Alice

RSMC Bob
Congestion Control

COV(Close Bob) → RSMC

COV(Close Alice) → Bob

close Bob → RSMC

close Alice → RSMC

SELECT → close Bob

SELECT → close Alice

SELECT → RSMC

SELECT → Alice

SELECT → Bob
Etc

- Inductive Execution
  - Start from the last transaction up

- Single Induction Execution
  - Run forward, except for first step
    - Run as above

- Rate Limited Rollback

- Traditional M-of-N timeouts
Quality of Service Matters

• We can’t just make protocols more resource-efficient, we need to make them work better when resources are constrained
“Secure Contracts Isolate Value”

• Give your friend the keys to your car
  – but not the garage door opener
  – because you can open the garage door from your phone

• That Bitcoin transaction propagate value makes incentives harder
Covenants Are Not Evil

• There are strong reasons to fear general-purpose covenants, but they still are worth consideration
• VUTXO-only COV is low risk
Bitcoin Must Pick Battles

• Tension between security and complexity
• Keep scripts simple predicates!
• Better higher-order inter-output interaction may be safer