M I N I S C R I P T

Spending Policies You Can Reason About

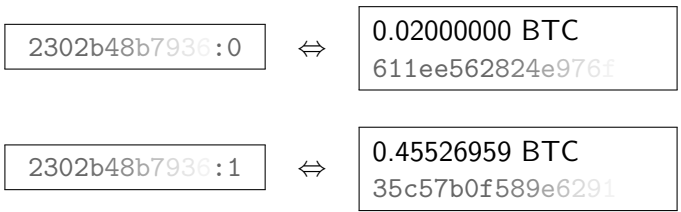
Andrew Poelstra
Director of Research, Blockstream

June 18, 2019

Bitcoin's state consists of a set of *unspent transaction outputs* (UTXOs).

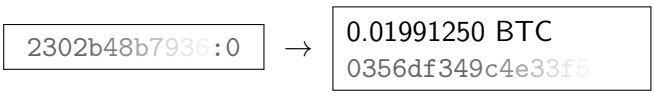
These UTXOs are labelled by an amount and witness program.

Bitcoin's state consists of a set of *unspent transaction outputs* (UTXOs).



These UTXOs are labelled by an amount and witness program.

Transactions destroy existing UTXOs and create new ones.



txid 39549b99b0f16dff9d3788b602f00df38a4b88ae647e269ecf82c51b15f1de32

To spend a UTXO, a transaction must include a witness for its witness program.


```
DUP HASH160 <hash> EQUALVERIFY CHECKSIG
```

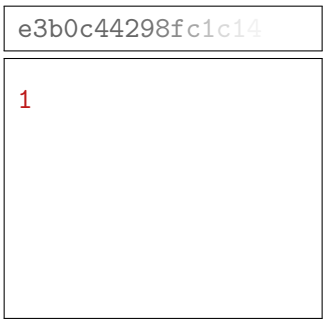
DUP HASH160 <hash> EQUALVERIFY CHECKSIG


```
DUP HASH160 <hash> EQUALVERIFY CHECKSIG
```

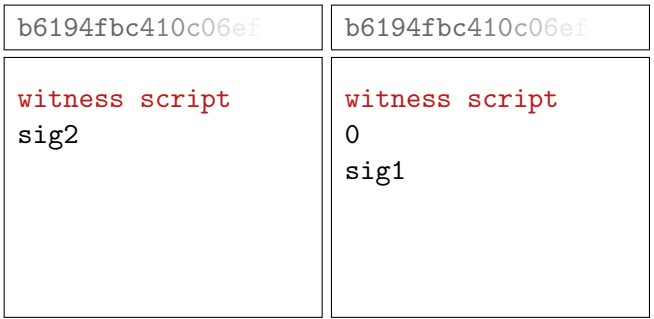
DUP HASH160 <hash> EQUALVERIFY CHECKSIG


```
DUP HASH160 <hash> EQUALVERIFY CHECKSIG
```

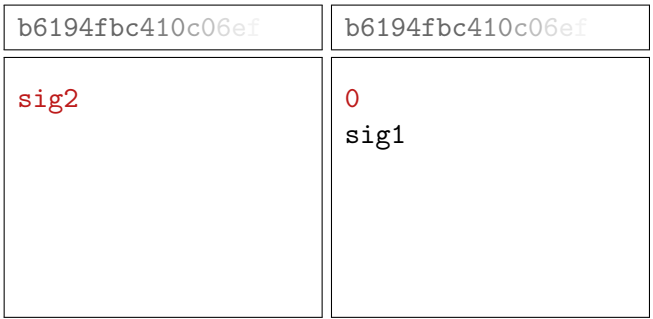

```
DUP HASH160 <hash> EQUALVERIFY CHECKSIG
```



DUP HASH160 <hash> EQUALVERIFY CHECKSIG




```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```

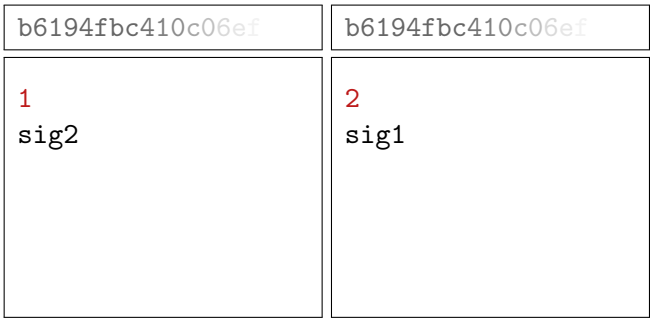



```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```

b6194fbc410c06ef	b6194fbc410c06ef
sig2 sig2	0 sig1

```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```

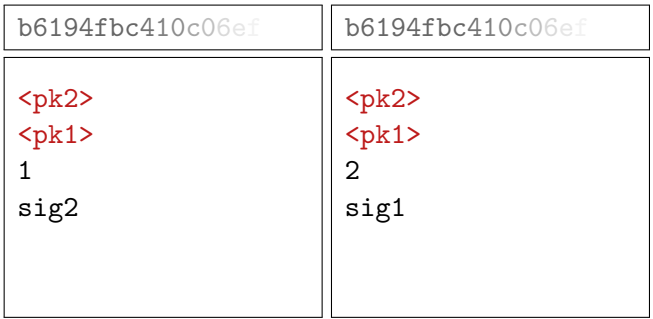

```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```



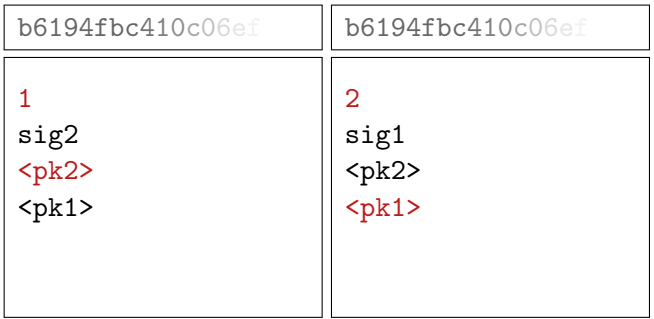

```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```

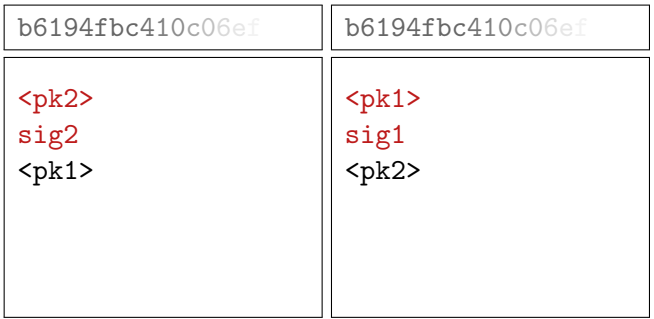
b6194fbc410c06ef	b6194fbc410c06ef
<pk1> 1 sig2	<pk1> 2 sig1

```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```

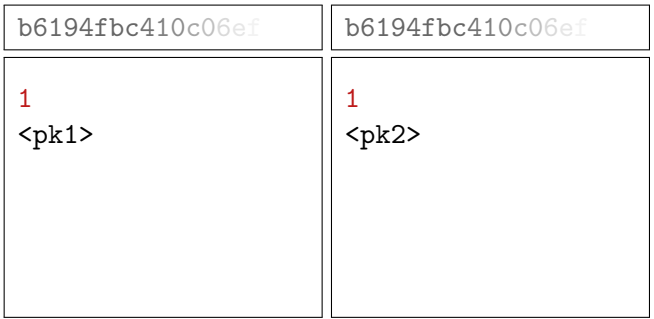



```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```

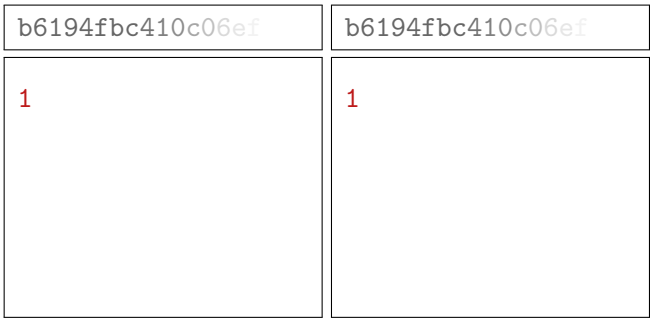

b6194fbc410c06ef
1
sig2
<pk2>
<pk1>
b6194fbc410c06ef
2
sig1
<pk2>
<pk1>
IFDUP NOT 1ADD <pk1> <pk2>
2SWAP ROLL CHECKSIG NIP




```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```




```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```



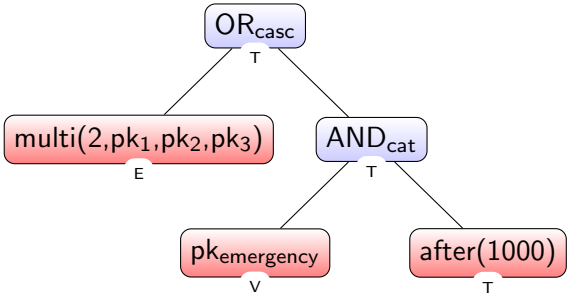

```
IFDUP NOT 1ADD <pk1> <pk2>
   2SWAP ROLL CHECKSIG NIP
```

Issues with Bitcoin Script

- Difficult to argue correctness (or other properties)
- Difficult to argue security (or malleability freeness)
- Difficult to estimate satisfaction cost
- Difficult to determine which signatures are needed
- Difficult to assemble a witness, even given signatures

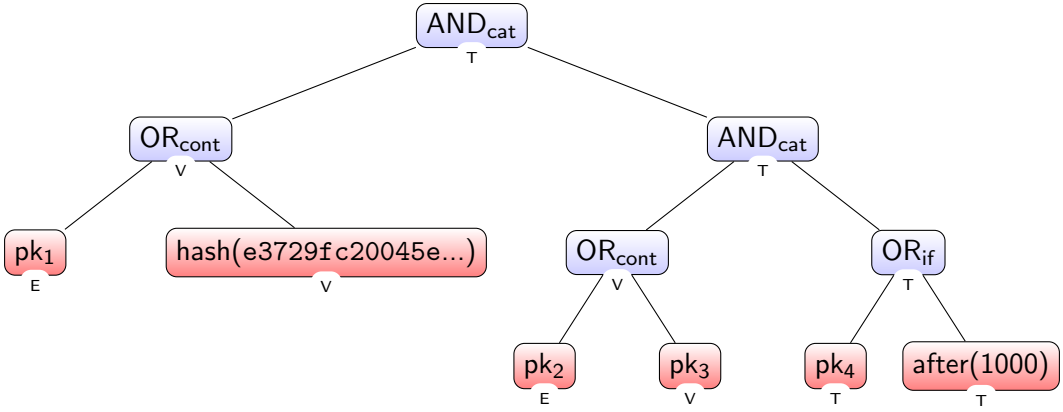
- Idea: create script templates for signature checks, hash-locks and time-locks
- Idea: create composable script templates for AND, OR and thresholds

```
2 pk1 pk2 pk3 3 CHECKMULTISIG
IFDUP NOTIF
    pkemergency CHECKSIGVERIFY
    1000 CSV
ENDIF
```



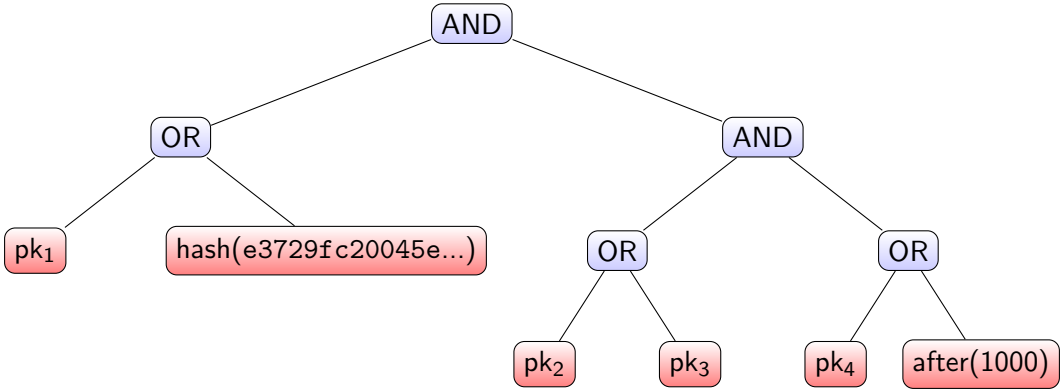
	Satisfaction	**Dissatisfaction**
T	non-0	0 or abort
V	-	abort
F	1	abort
Q	pubkey	abort
E	1	0
W	`[X 1]` or `[1 X]`	`[X 0]` or `[0 X]`

Under adversarial inputs, all conventions will simply abort the script.



pk$_1$ CHECKSIG NOTIF SIZE 32 EQUALVERIFY SHA256 e3729fc20045e8b5 EQUALVERIFY ENDIF

pk$_2$ CHECKSIG NOTIF pk$_3$ CHECKSIGVERIFY ENDIF IF pk$_4$ CHECKSIG ELSE 1000 CSV ENDIF



pk$_1$ CHECKSIG NOTIF SIZE 32 EQUALVERIFY SHA256 e3729fc20045e8b5 EQUALVERIFY ENDIF

pk$_2$ CHECKSIG NOTIF pk$_3$ CHECKSIGVERIFY ENDIF IF pk$_4$ CHECKSIG ELSE 1000 CSV ENDIF

Script and Miniscript

- In a technical sense, Miniscript is a subset of Script.
- In a non-technical sense, Miniscript works in a different paradigm than Script
- Miniscript describes conditions to satisfy, not instructions to execute

Future Work

- Better documentation and more robust tooling
- Extensions to support absolute timelocks, different hashes
- Extensions to support new Script constructions (pubkey hashes)
- Integration with PSBT

Thank You

Andrew Poelstra
miniscript@wpsoftware.net

https://bitcoin.sipa.be/miniscript