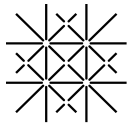


University of Basel

Center for Innovative Finance

Bitcoin, Blockchain and Cryptoassets

Bitcoin Script Transaction Types

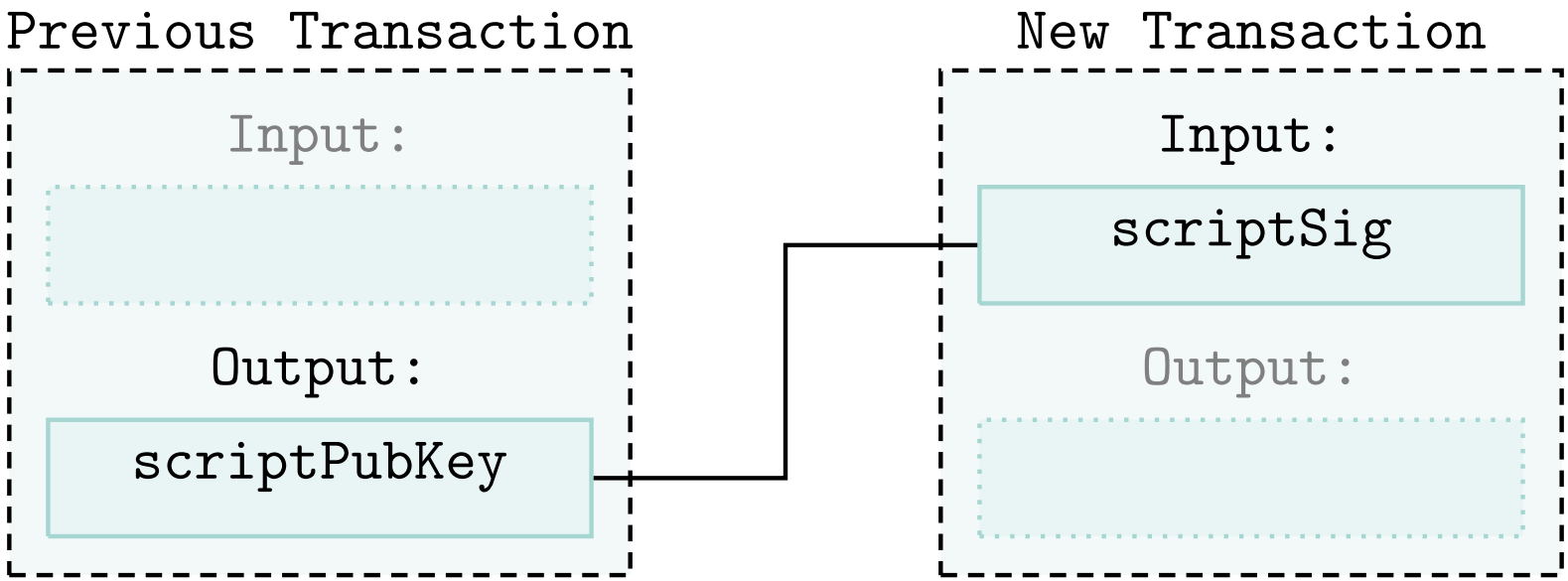
Prof. Dr. Fabian Schär
University of Basel

Release Ver.: (Local Release)
Version Hash: (None)
Version Date: (None)

Unlocking Condition and `Script`

- Unlocking conditions are written and verified in `Script`.
  $\rightarrow$ Stack-based scripting language with predetermined list of commands, called *OP Codes*.
- Only if all commands run through without errors and the end result of the stack is a `1` (`TRUE`), the transaction is considered valid.
- Does not contain any loops, as these enable potential attack vectors.

scriptPubKey and scriptSig in Transactions

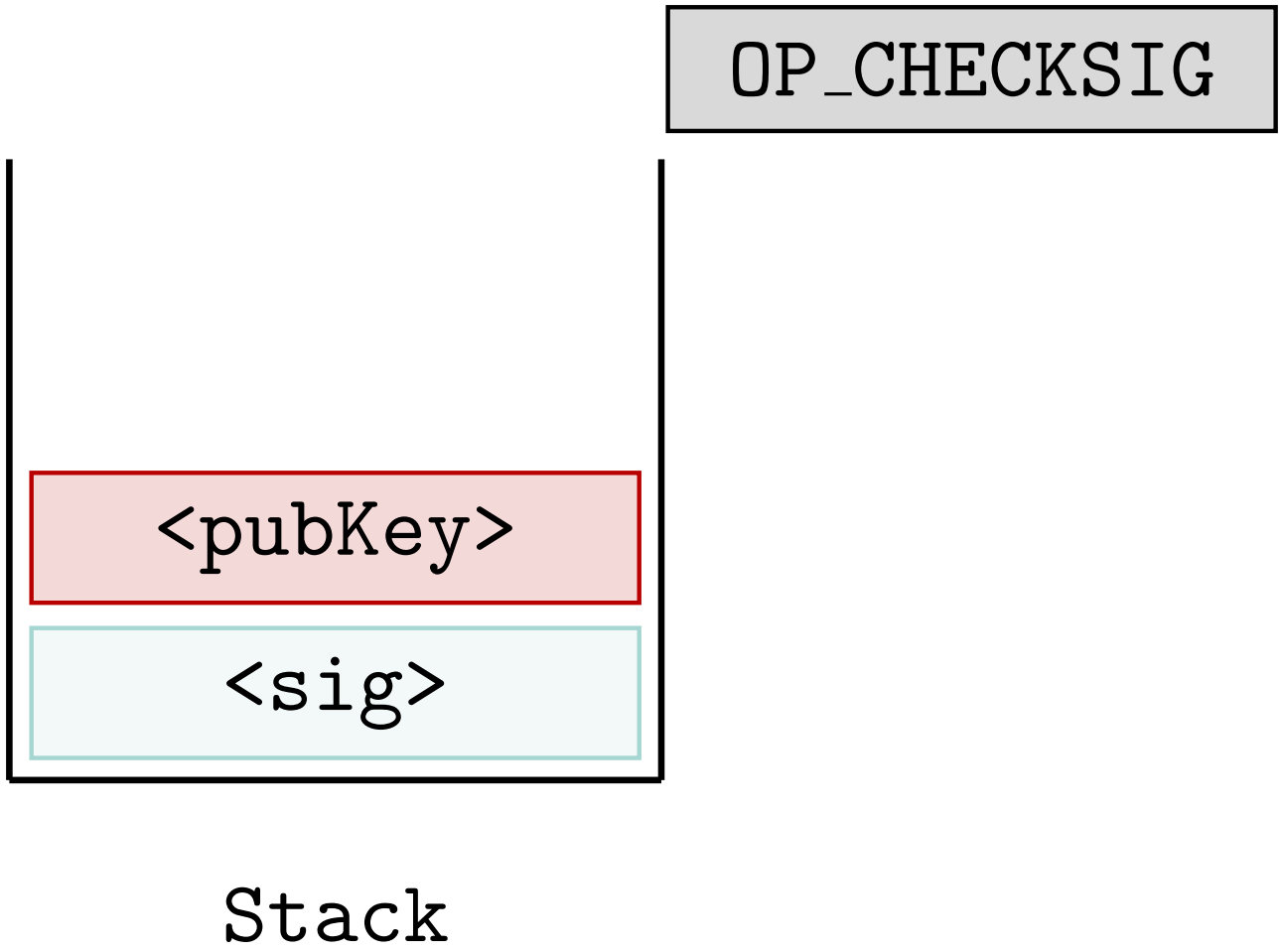


Pay-to-Public-Key

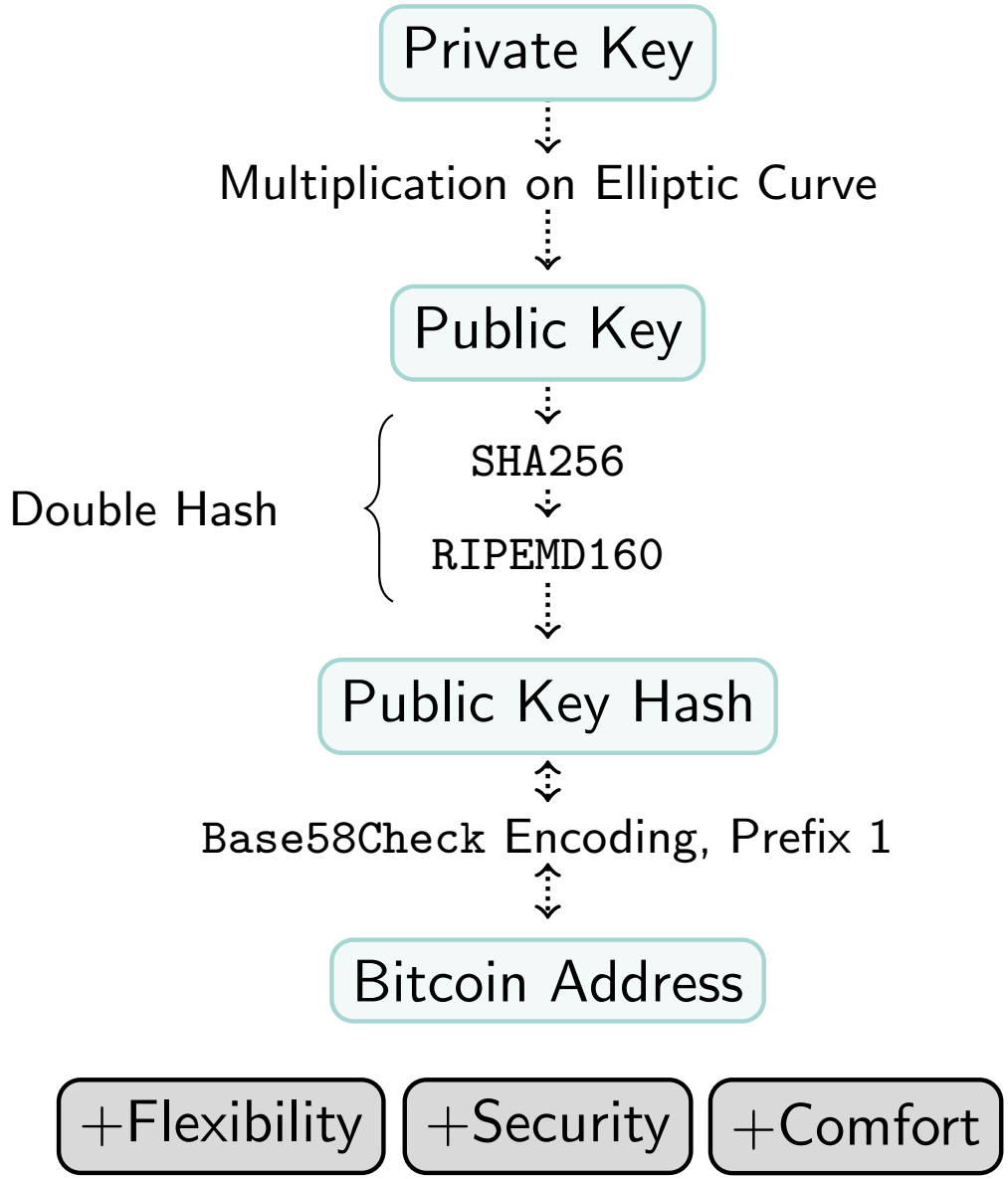
```
scriptSig:  <sig>
scriptPubKey:  <pubKey> OP_CHECKSIG
```

- Pay-to-Public-Key links output directly to the public key.
- Solution (`scriptSig`) only includes corresponding signature (`<sig>`).
- Public key (`<pubKey>`) is stored on Blockchain as part of the unlocking condition.

P2PK



Bitcoin Address

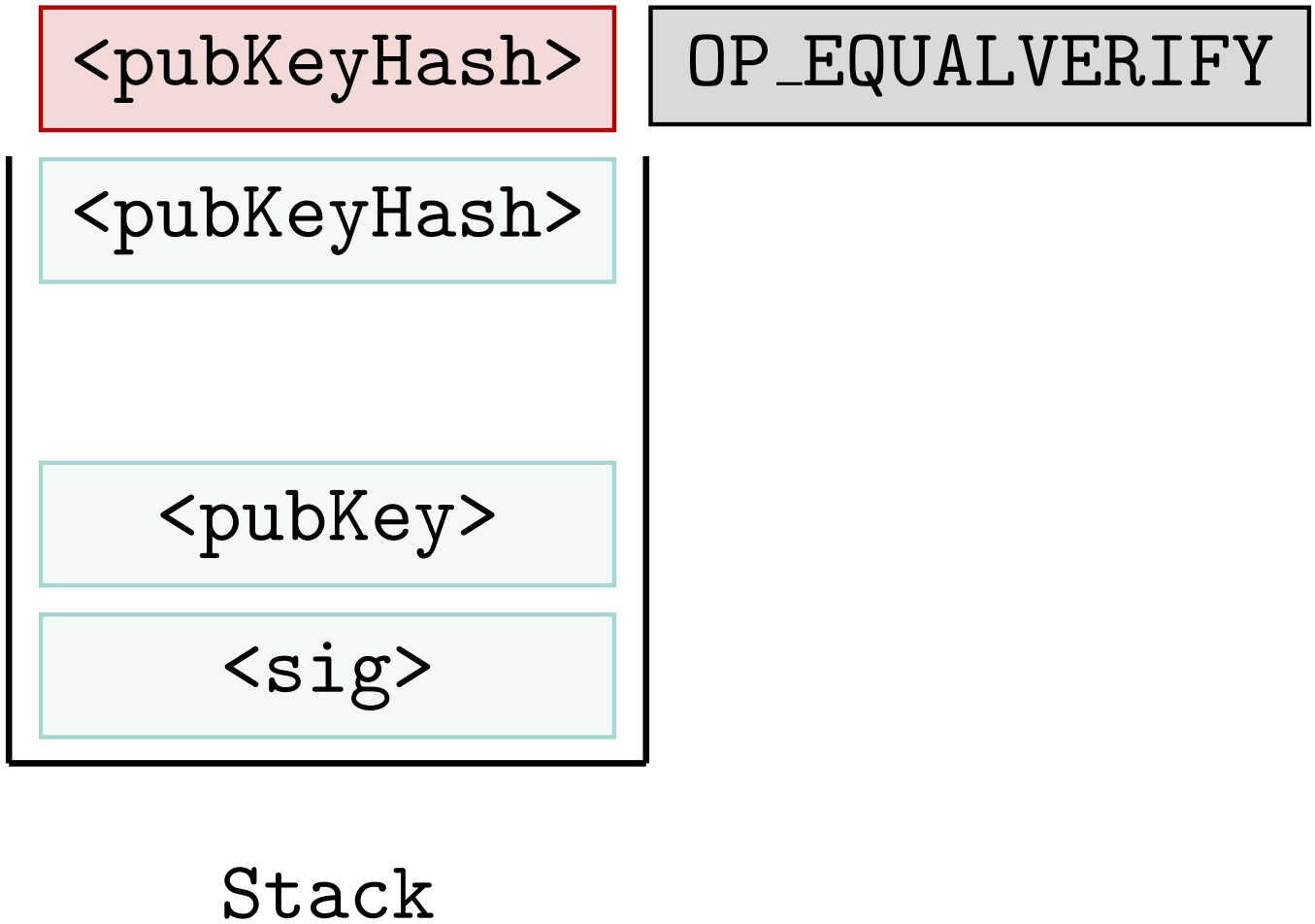


Pay-to-Address / Pay-to-Public-Key-Hash

```
scriptSig:  <sig> <pubKey>
scriptPubKey:  OP_DUP OP_HASH160 <pubKeyHash>
OP_EQUALVERIFY OP_CHECKSIG
```

- Output is linked to Bitcoin address instead of public key.
- Reference of an output with this unlocking condition only valid if the `scriptSig` contains:
  1. The public key (`<pubKey>`) associated with the address.
  2. A matching signature (`<sig>`), derived from the corresponding private key.

P2PKH

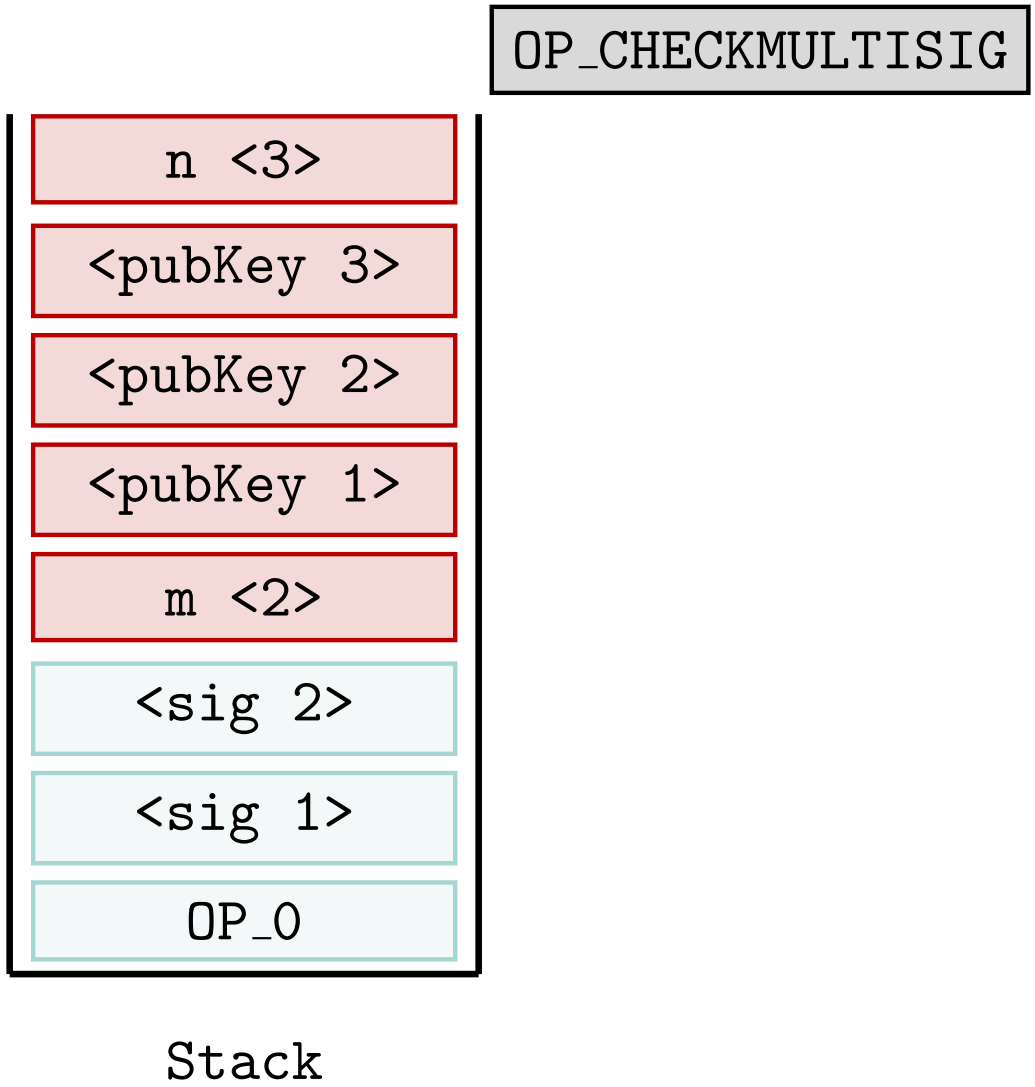


Multisig (M of N)

```
scriptSig:  OP_0 <sig1> <sig2>
scriptPubKey:  M <pubKey1>...<pubKeyN> N
OP_CHECKMULTISIG
```

- Enables payout conditions that require M of N signatures to reference the corresponding transaction output.
- Unlocking condition (`scriptPubKey`) includes N public keys.
- At least M of the corresponding private keys must provide a valid signature in `scriptSig`.
- Applications include increasing the security of funds and the possibility of imitating classic bank accounts (joint account, corporate account).

Pay-to-Multisig (2 out of 3 Script)

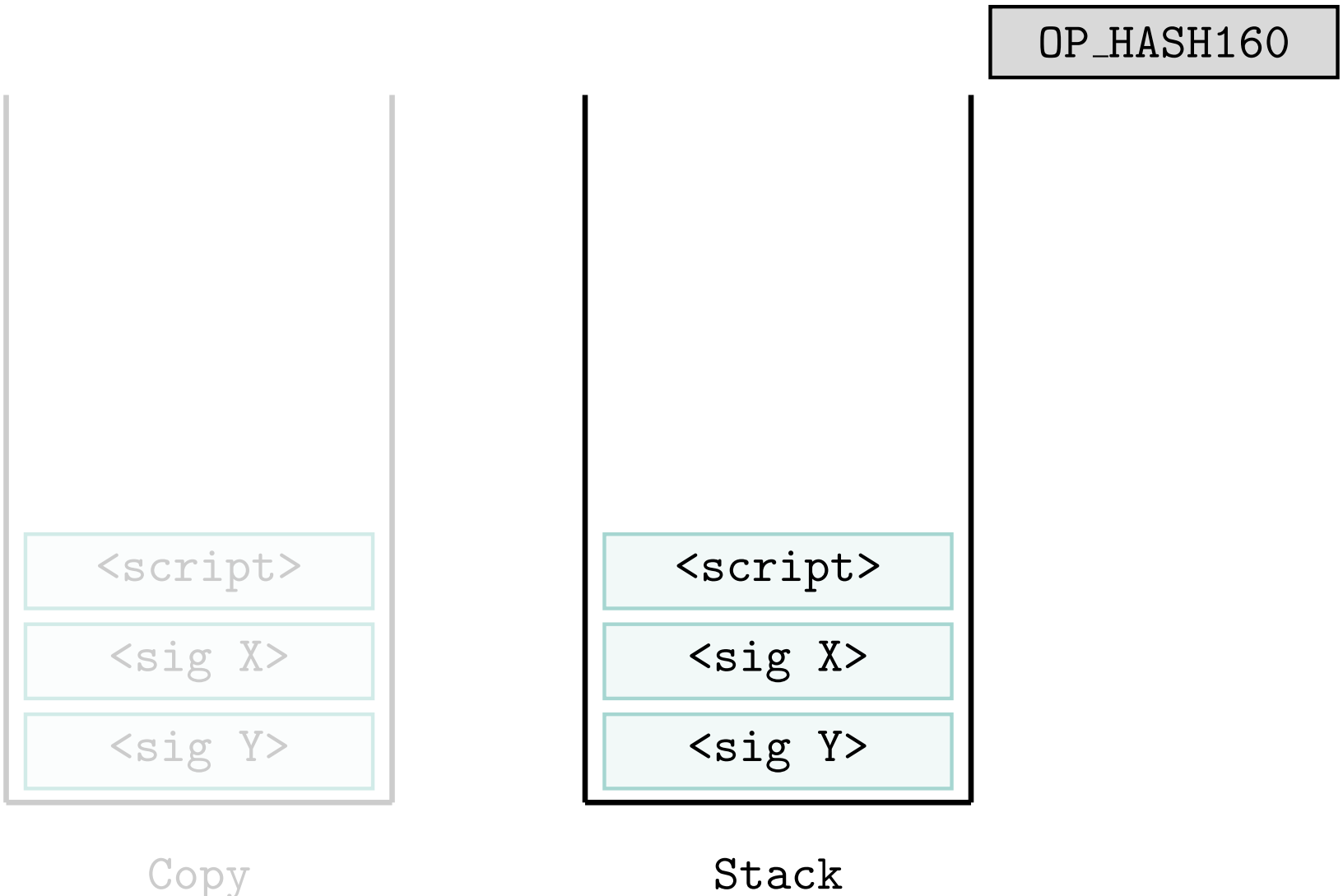


Pay-to-Script-Hash (Flexible Scripts)

```
scriptSig:    Any valid script
scriptPubKey:   OP_HASH160 <scriptHash>
OP_EQUALVERIFY
```

- In the `scriptPubKey`, only the hash value of a script is recorded.
- To reference the transaction output, a person must provide a `scriptSig` whose hash value corresponds to the hash value set in the `scriptPubKey`.
- If this succeeds, the complete script of the `scriptSig` is run through in the second step.

Multisig with P2SH



Characteristics and Risks of Pay-to-Script-Hash

- By using the cryptographic hash value, it can be ensured that the unlocking condition can only be solved using the previously specified script.
- High degree of flexibility, as Bitcoin units can be collateralised by a wide variety of scripts.
- Scripts only need to be submitted at the time of a transaction.
- Can be represented as a kind of Bitcoin address with prefix 3.

- Potential problems:
  - P2SH addresses based on invalid scripts.
  - Loss of script.

  $\Rightarrow$ Both result in loss of Bitcoin units.

Null Data (OP_RETURN)

```
scriptSig:  Non-existent.
scriptPubKey:  OP_RETURN <0 to 40 Bytes of Data>
```

- Null Data is not used to transfer Bitcoin units.
- Instead, it is used to store arbitrary data up to 40 bytes (320 bits) in the blockchain.
- Possible applications:
  - *Proof-of-Existence*: Proof that a certain file existed at a certain point in time. For this purpose, the hash value of a file is stored in the blockchain using a null data transaction.
  - Many *Colored Coin* implementations also use the additional data to identify the linked promises to pay.