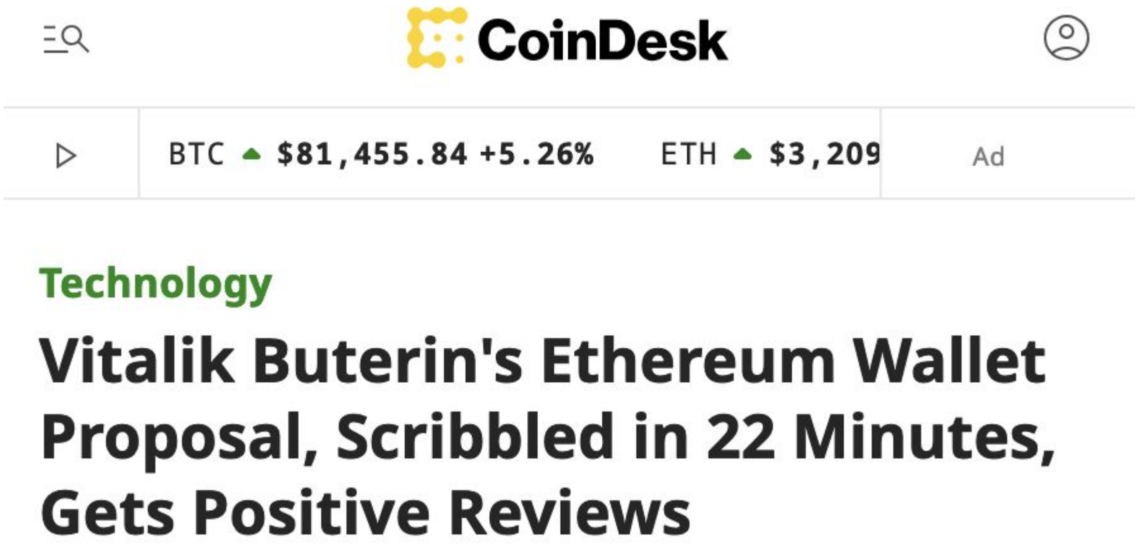
EIP-7702

@lightclients



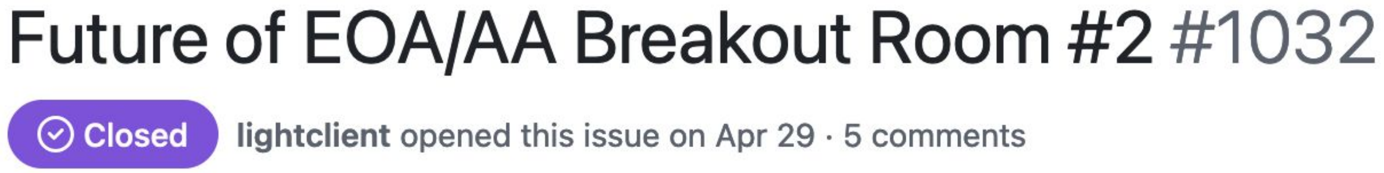
CoinDesk

BTC $81,455.84 +5.26% ETH $3,209

Ad

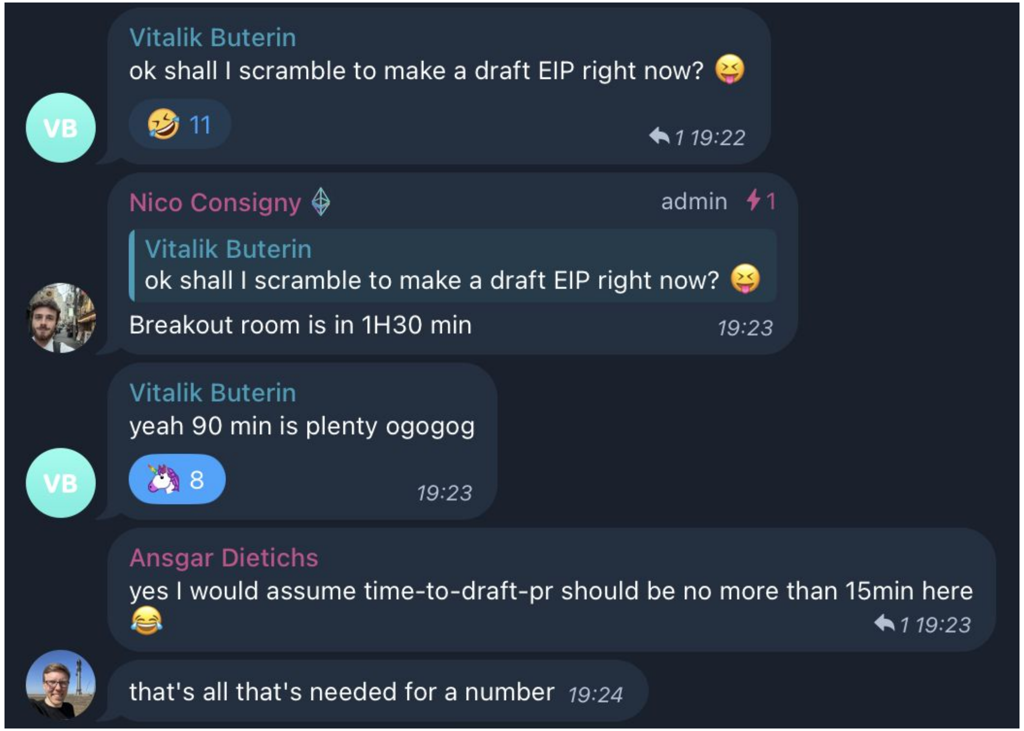
Technology

Vitalik Buterin's Ethereum Wallet Proposal, Scribbled in 22 Minutes, Gets Positive Reviews



Future of EOA/AA Breakout Room #2 #1032

Closed lightclient opened this issue on Apr 29 · 5 comments



Vitalik Buterin
ok shall I scramble to make a draft EIP right now? 😝
🤣 11
1 19:22

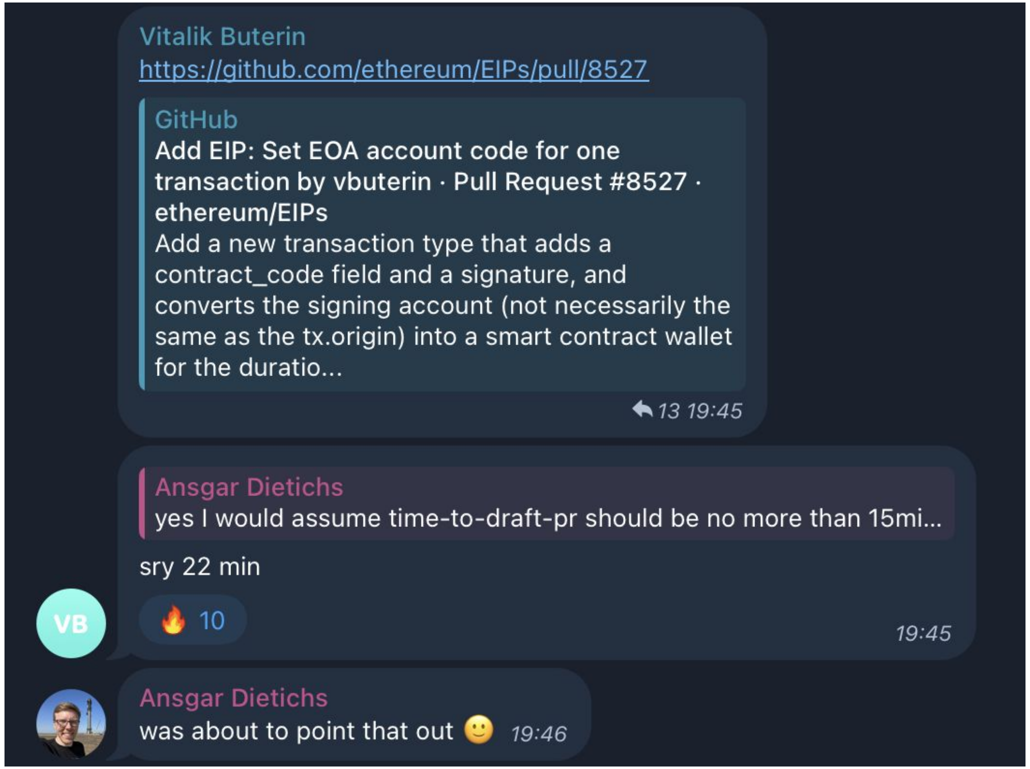
Nico Consigny
admin 1
> Vitalik Buterin
> ok shall I scramble to make a draft EIP right now? 😝

Breakout room is in 1H30 min 19:23

Vitalik Buterin
yeah 90 min is plenty ogogog
🦄 8
19:23

Ansgar Dietichs
yes I would assume time-to-draft-pr should be no more than 15min here 😂
1 19:23

that's all that's needed for a number 19:24



Vitalik Buterin
https://github.com/ethereum/EIPs/pull/8527

GitHub
Add EIP: Set EOA account code for one transaction by vbuterin · Pull Request #8527 · ethereum/EIPs
Add a new transaction type that adds a contract_code field and a signature, and converts the signing account (not necessarily the same as the tx.origin) into a smart contract wallet for the duratio...

13 19:45

> Ansgar Dietichs
> yes I would assume time-to-draft-pr should be no more than 15mi...

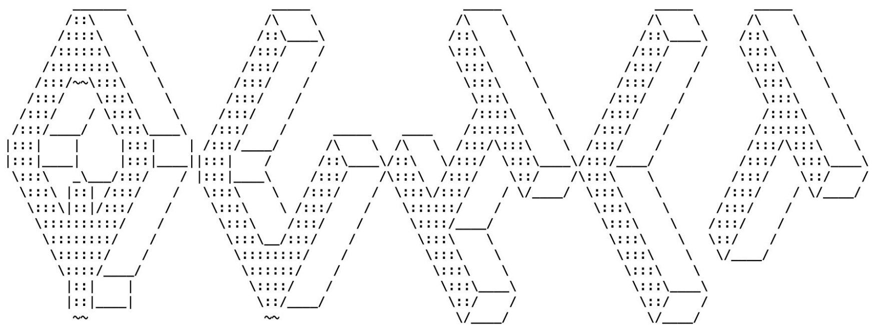
sry 22 min

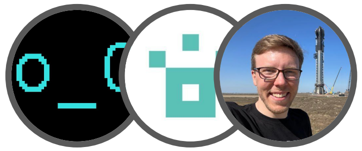
🔥 10

19:45

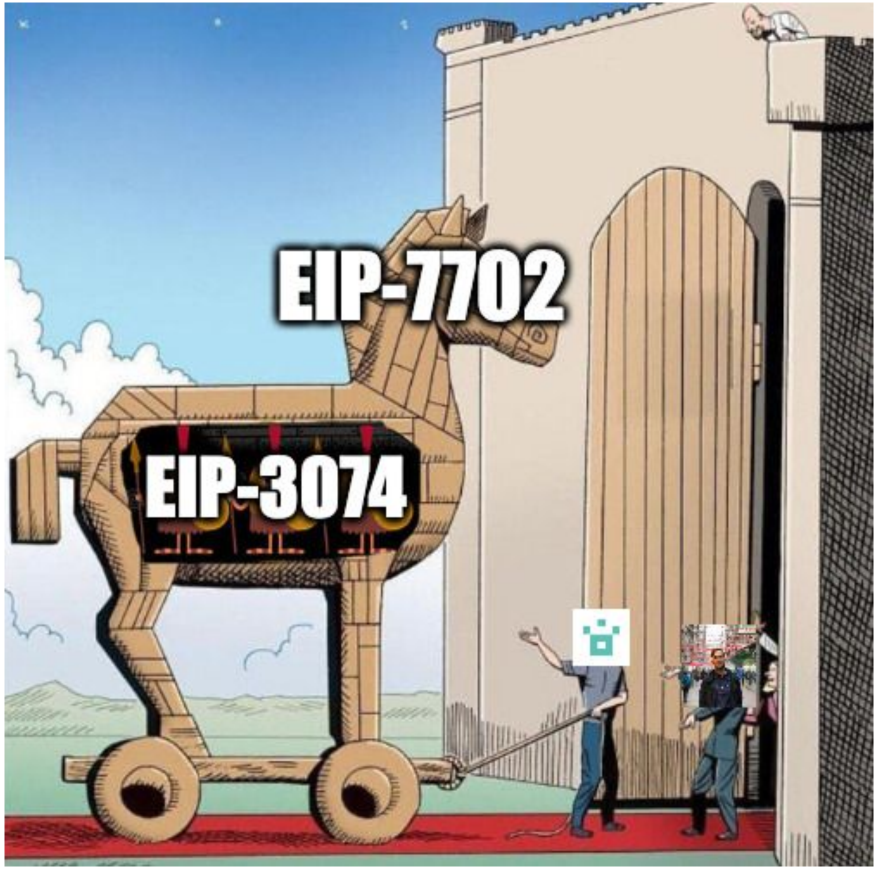
Ansgar Dietichs
was about to point that out 🙂 19:46

pre-7702





3074


EIP-7702
EIP-3074

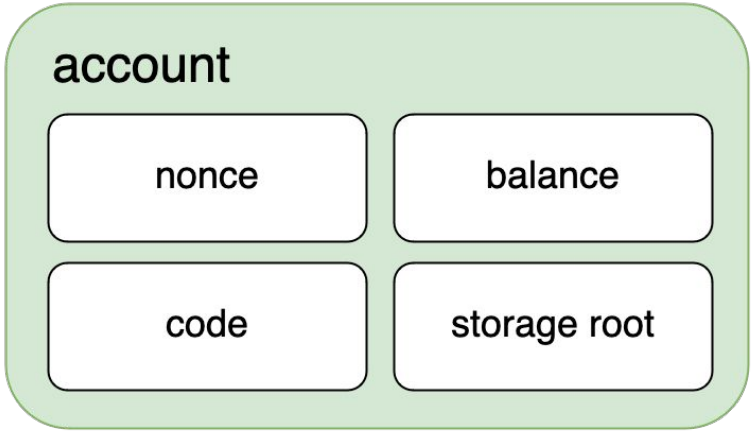
EIP-7702: Set EOA code

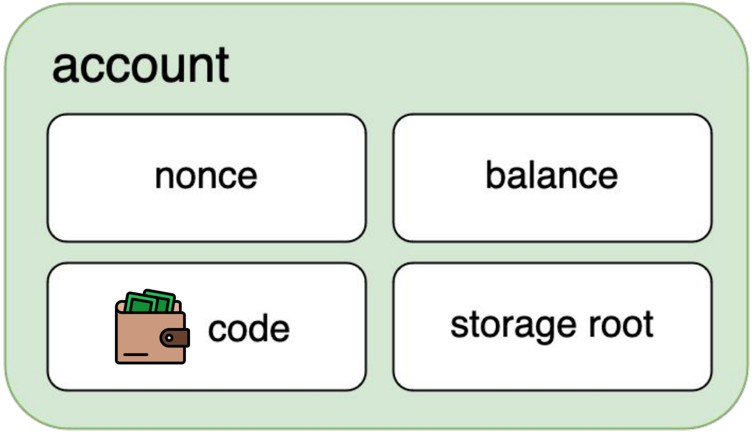
motivation

- batching
- sponsorship
- privilege de-escalation
- bootstrapping

motivation

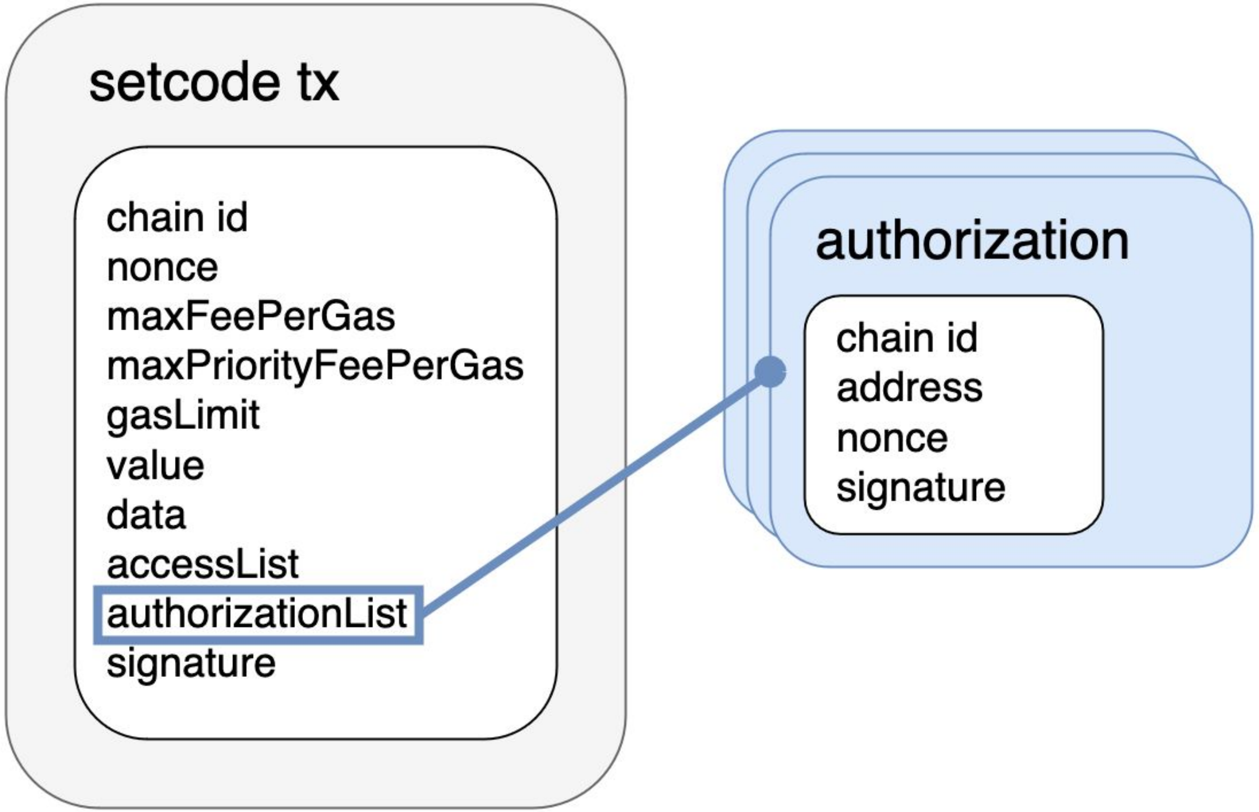
- batching
- sponsorship
- privilege de-escalation
- bootstrapping

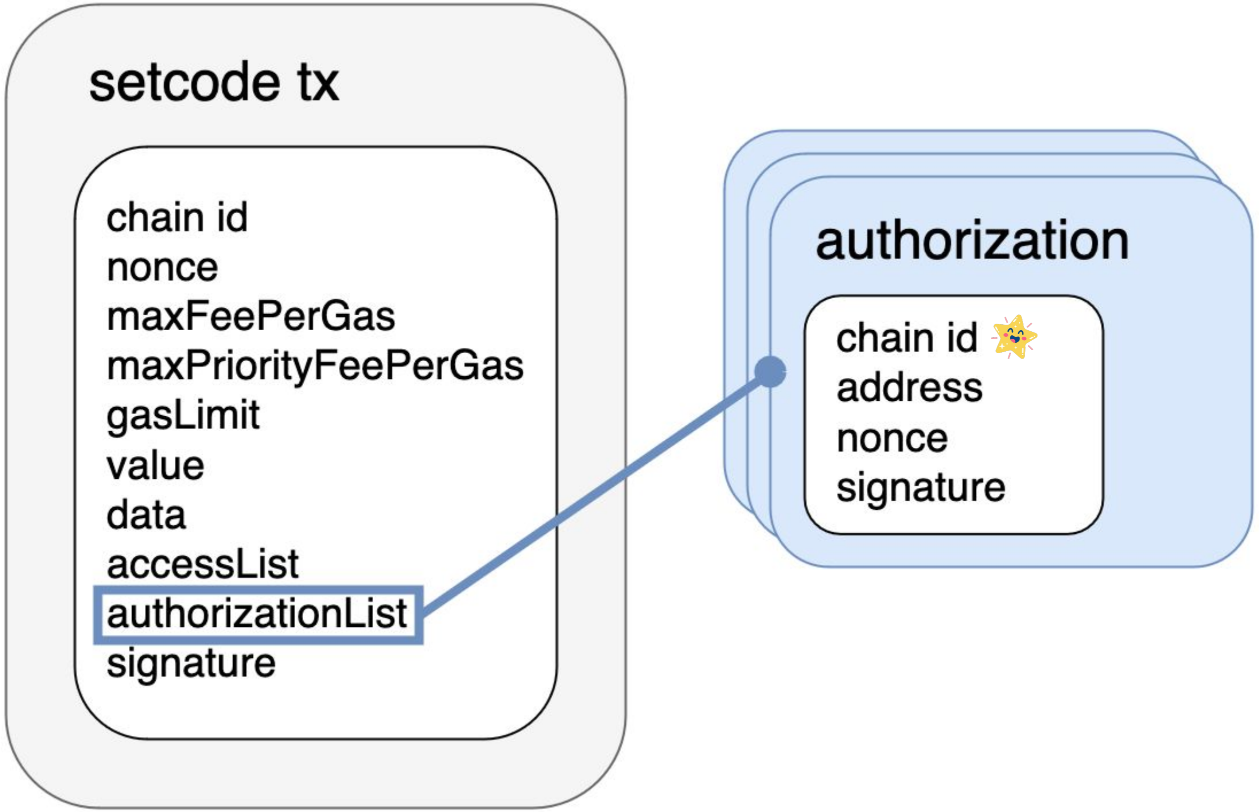

account
nonce
balance
code
storage root

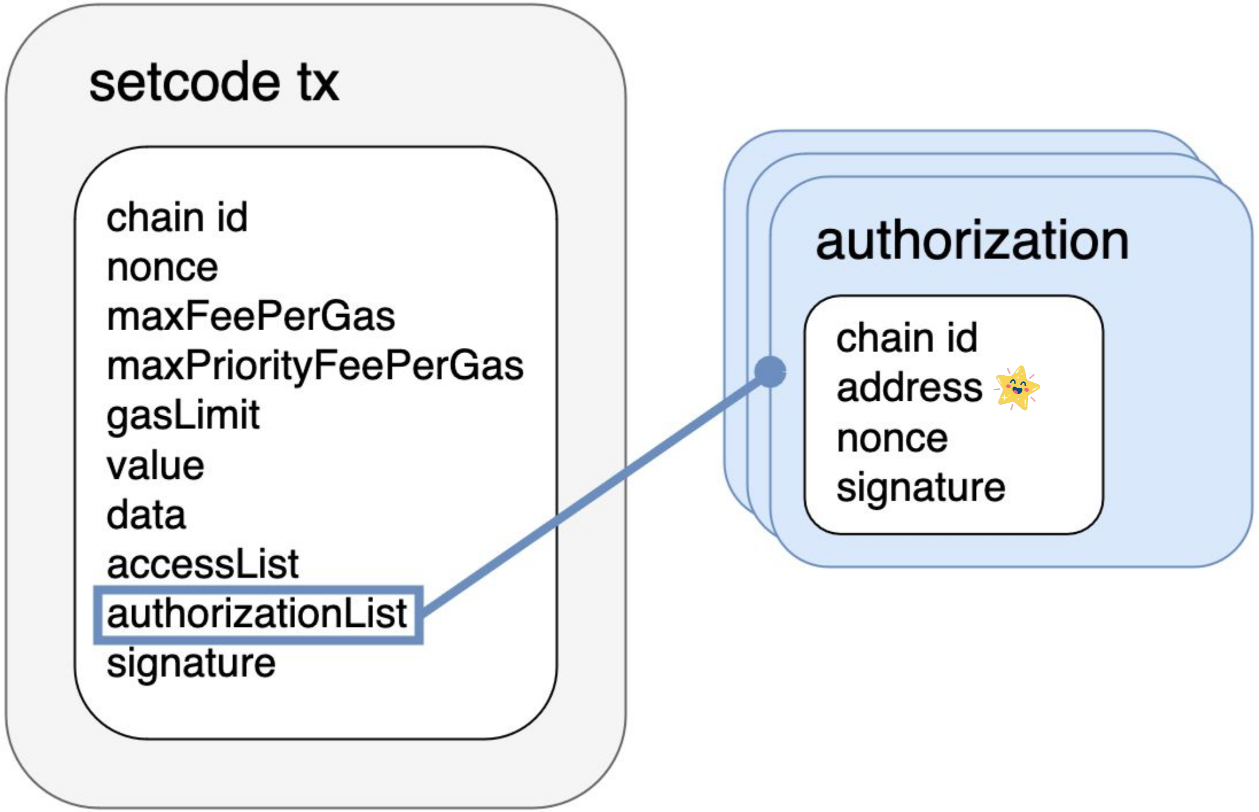

account
nonce
balance
code
storage root

setcode tx

chain id
nonce
maxFeePerGas
maxPriorityFeePerGas
gasLimit
value
data
accessList
authorizationList
signature


setcode tx
chain id
nonce
maxFeePerGas
maxPriorityFeePerGas
gasLimit
value
data
accessList
authorizationList
signature
authorization
chain id
address
nonce
signature


setcode tx
chain id
nonce
maxFeePerGas
maxPriorityFeePerGas
gasLimit
value
data
accessList
authorizationList
signature
authorization
chain id
address
nonce
signature


setcode tx
chain id
nonce
maxFeePerGas
maxPriorityFeePerGas
gasLimit
value
data
accessList
authorizationList
signature
authorization
chain id
address
nonce
signature

creation by template

- instead of running initcode, write the target template code to the account
  - initialize later via regular call
- less calldata needed -> 20 bytes for template address versus ~25-30 extra bytes for setup
- minimizes odds that users change delegation frequently
- due to EIP-3607, transactions from an account w/ code are invalid
  - need to work around this

delegation designator

```
0xef0100a94f5374fce5edbc8e2a8697c15331677e6ebf0b
```

delegation designator

$$\texttt{0x}\underbrace{\texttt{ef}}_{3541}\texttt{0100a94f5374fce5edbc8e2a8697c15331677e6ebf0b}$$

delegation designator

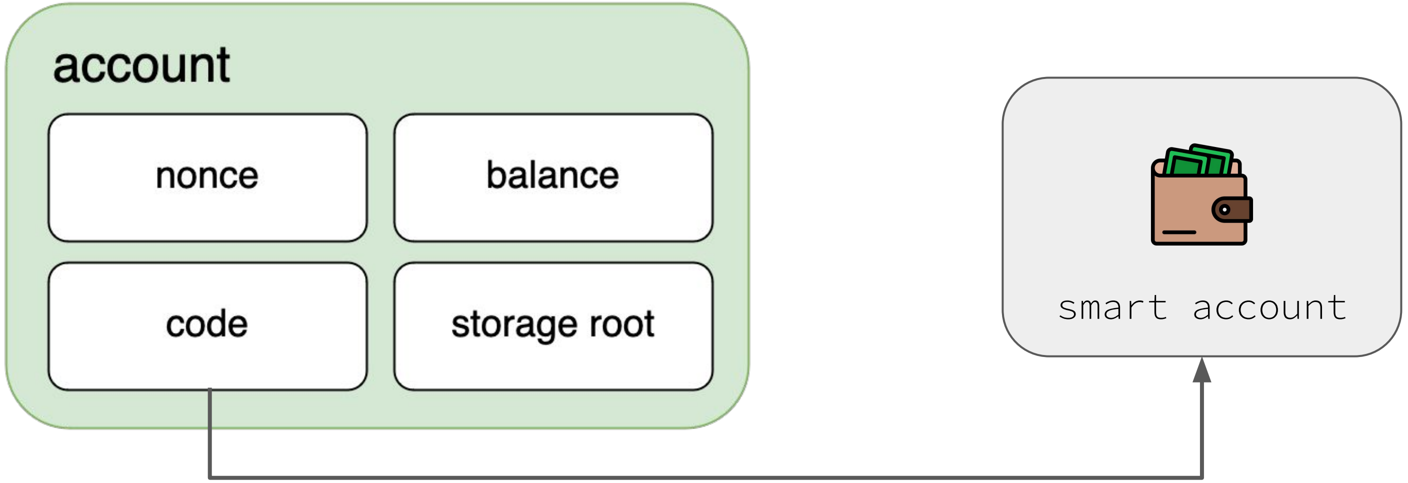
$$\texttt{0x}\underbrace{\texttt{ef}}_{3541}\underbrace{\texttt{0100}}_{7702}\texttt{a94f5374fce5edbc8e2a8697c15331677e6ebf0b}$$

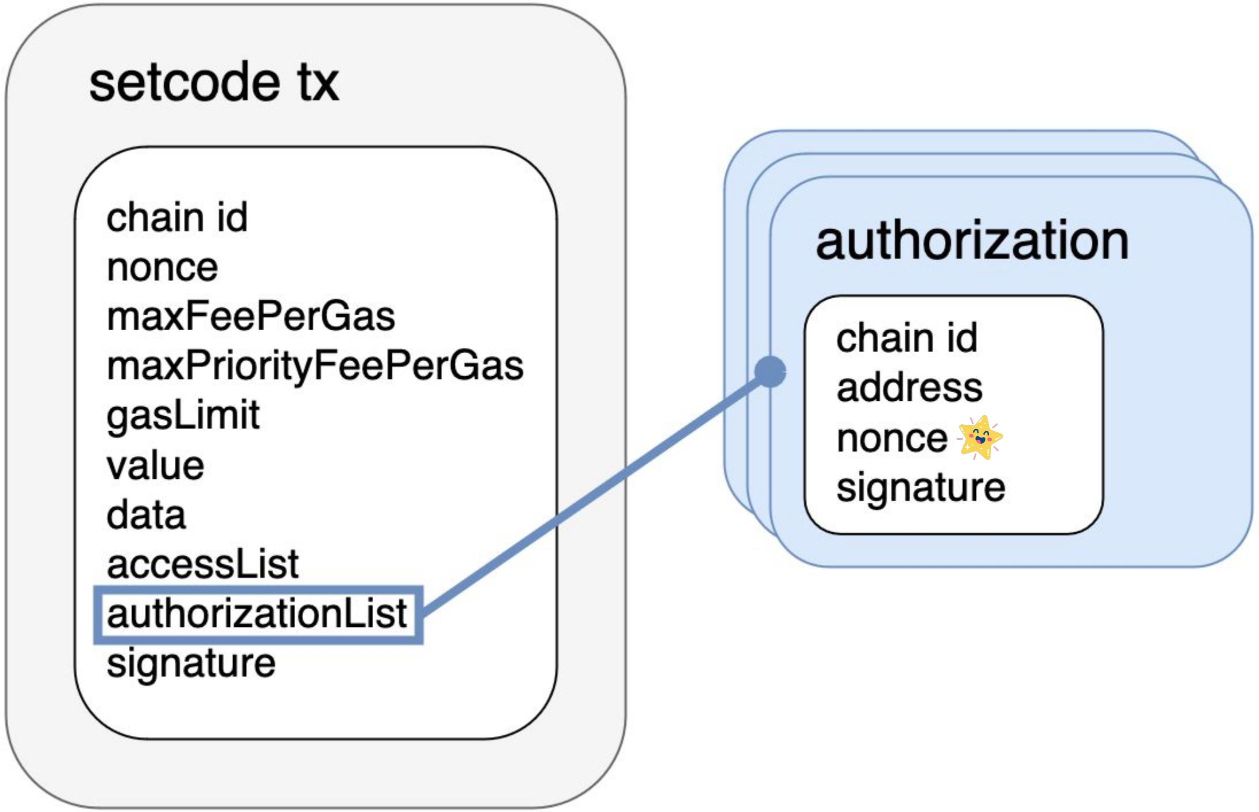
delegation designator

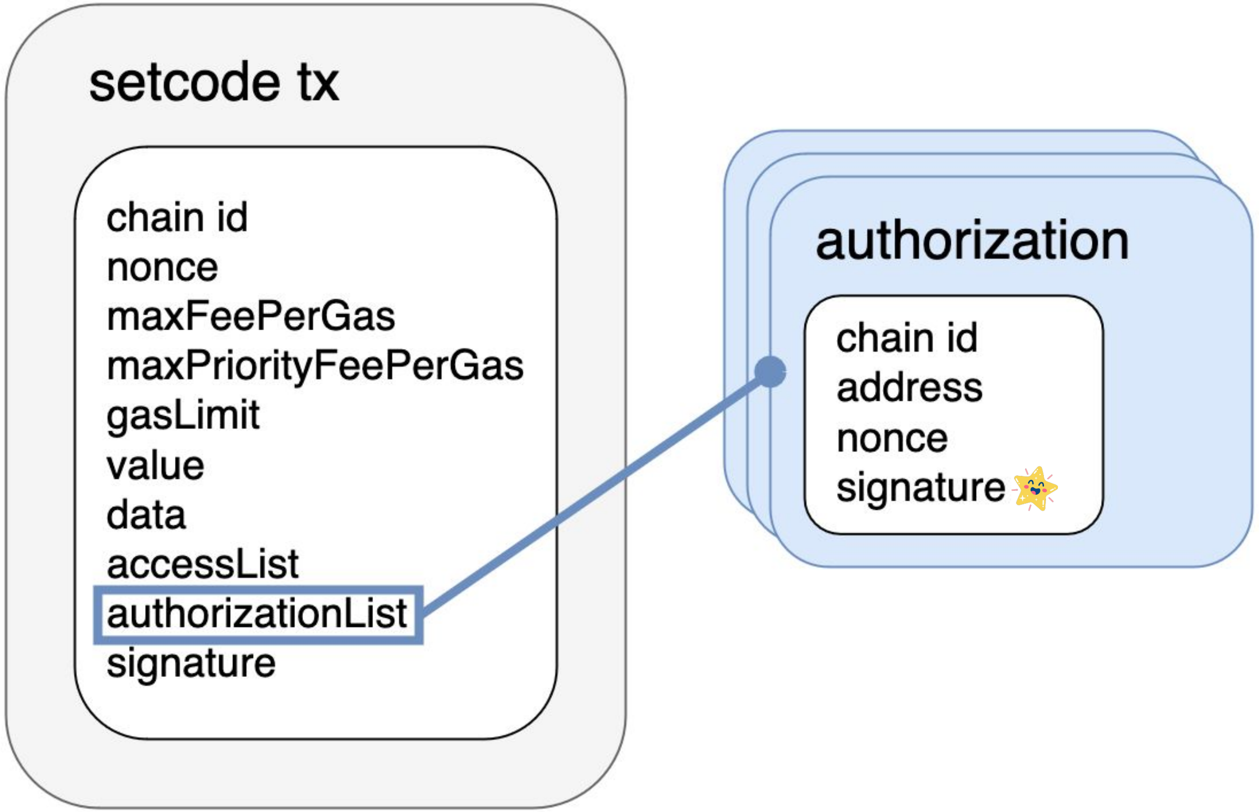
$$\texttt{0x}\underbrace{\texttt{ef}}_{\texttt{3541}}\underbrace{\texttt{0100}}_{\texttt{7702}}\underbrace{\texttt{a94f5374fce5edbc8e2a8697c15331677e6ebf0b}}_{\texttt{target address}}$$

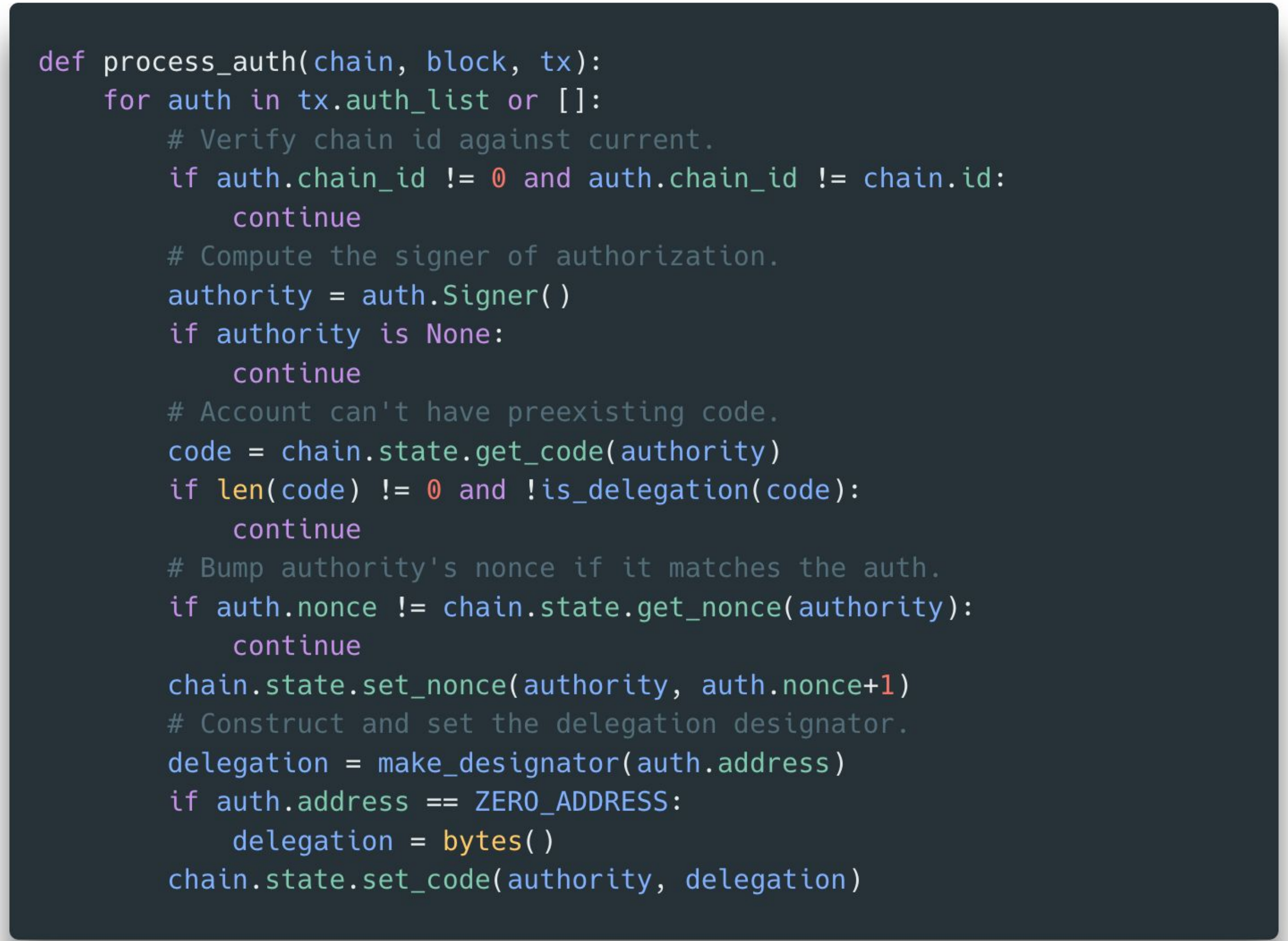
delegation designator

$$\texttt{0x}\underbrace{\texttt{ef}}_{\texttt{3541}}\underbrace{\texttt{0100}}_{\texttt{7702}}\underbrace{\texttt{a94f5374fce5edbc8e2a8697c15331677e6ebf0b}}_{\texttt{target address}}$$

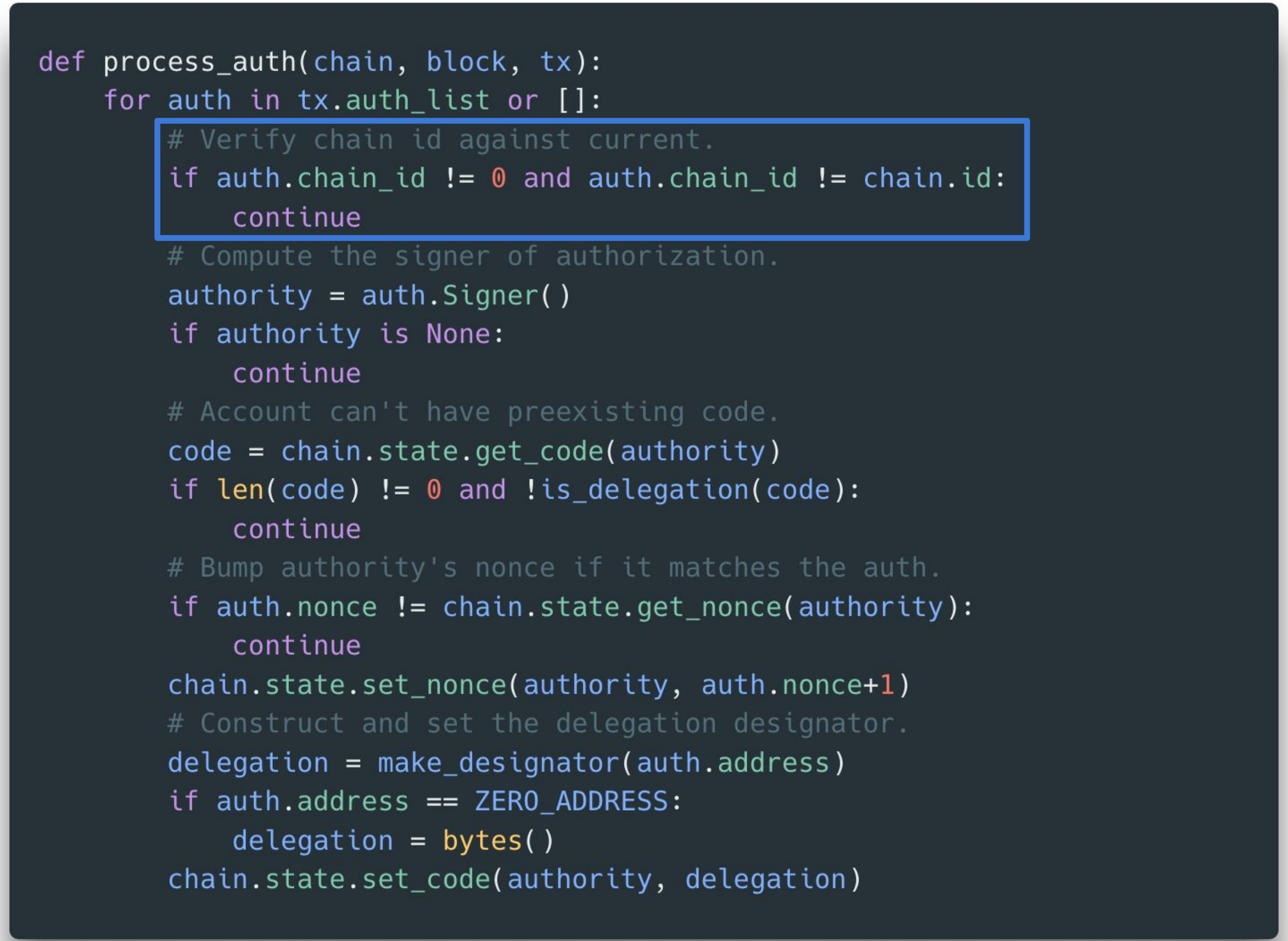



setcode tx
chain id
nonce
maxFeePerGas
maxPriorityFeePerGas
gasLimit
value
data
accessList
authorizationList
signature
authorization
chain id
address
nonce
signature

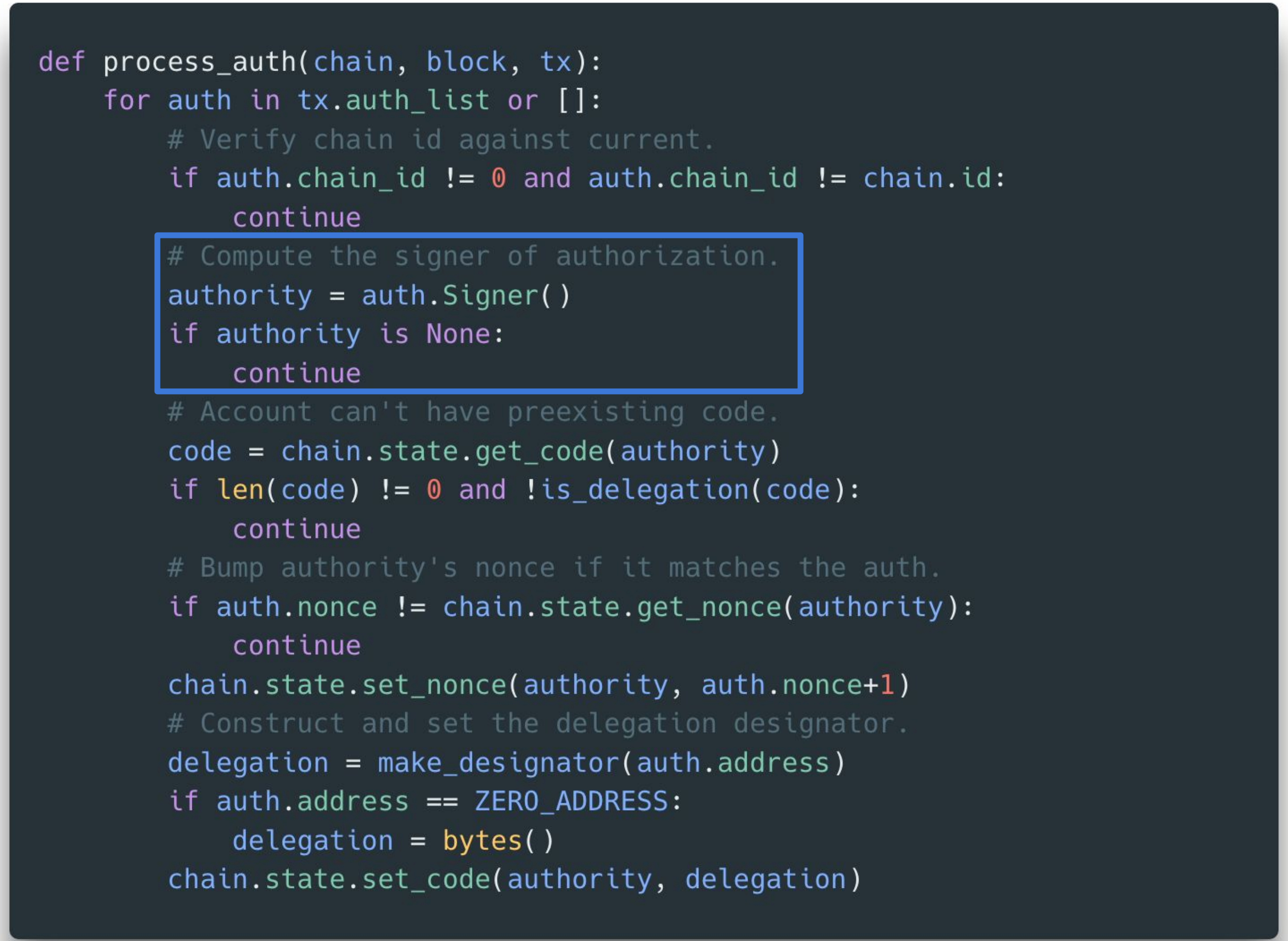

setcode tx
chain id
nonce
maxFeePerGas
maxPriorityFeePerGas
gasLimit
value
data
accessList
authorizationList
signature
authorization
chain id
address
nonce
signature



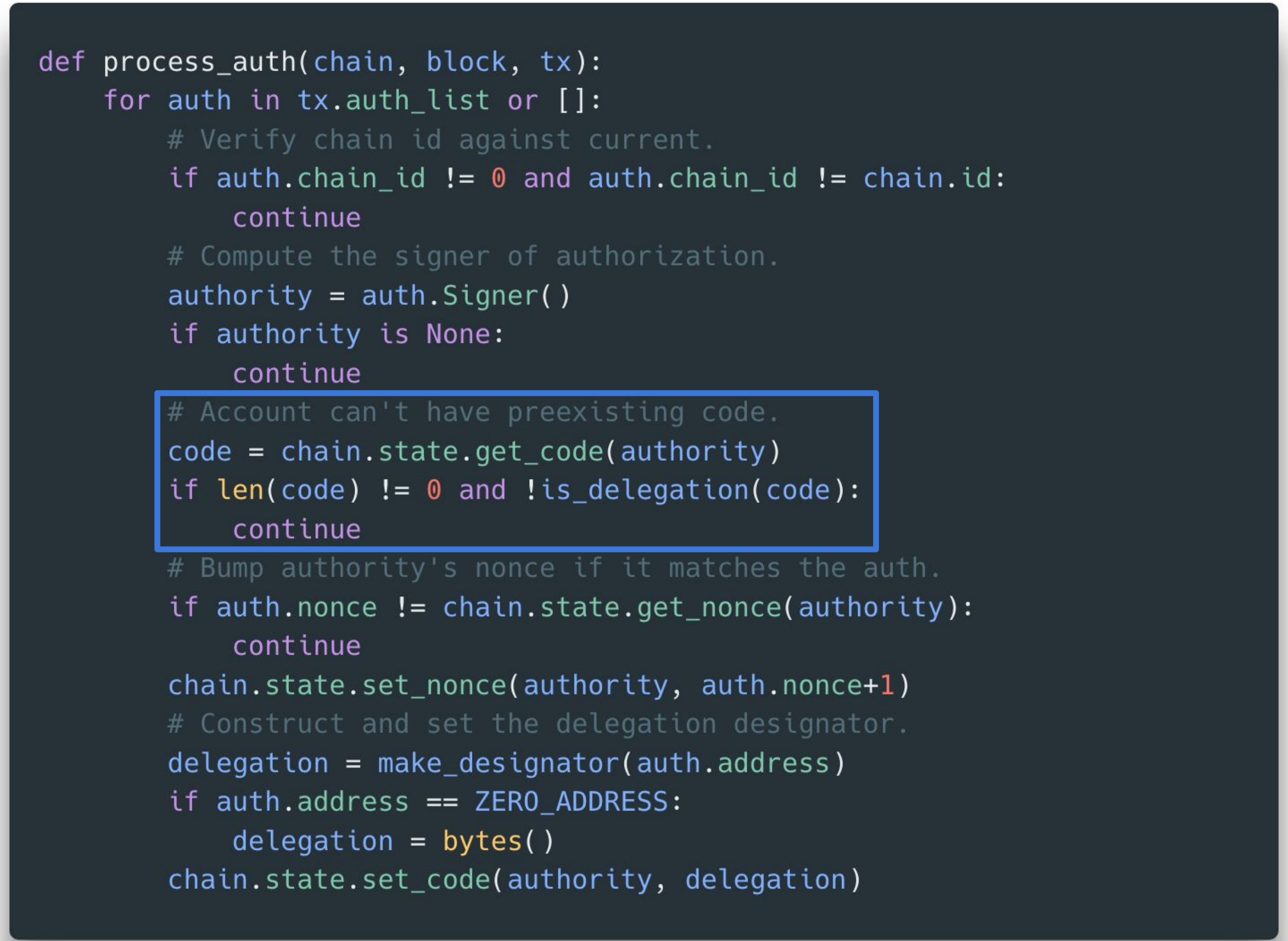
```
def process_auth(chain, block, tx):
    for auth in tx.auth_list or []:
        # Verify chain id against current.
        if auth.chain_id != 0 and auth.chain_id != chain.id:
            continue
        # Compute the signer of authorization.
        authority = auth.Signer()
        if authority is None:
            continue
        # Account can't have preexisting code.
        code = chain.state.get_code(authority)
        if len(code) != 0 and !is_delegation(code):
            continue
        # Bump authority's nonce if it matches the auth.
        if auth.nonce != chain.state.get_nonce(authority):
            continue
        chain.state.set_nonce(authority, auth.nonce+1)
        # Construct and set the delegation designator.
        delegation = make_designator(auth.address)
        if auth.address == ZERO_ADDRESS:
            delegation = bytes()
        chain.state.set_code(authority, delegation)
```



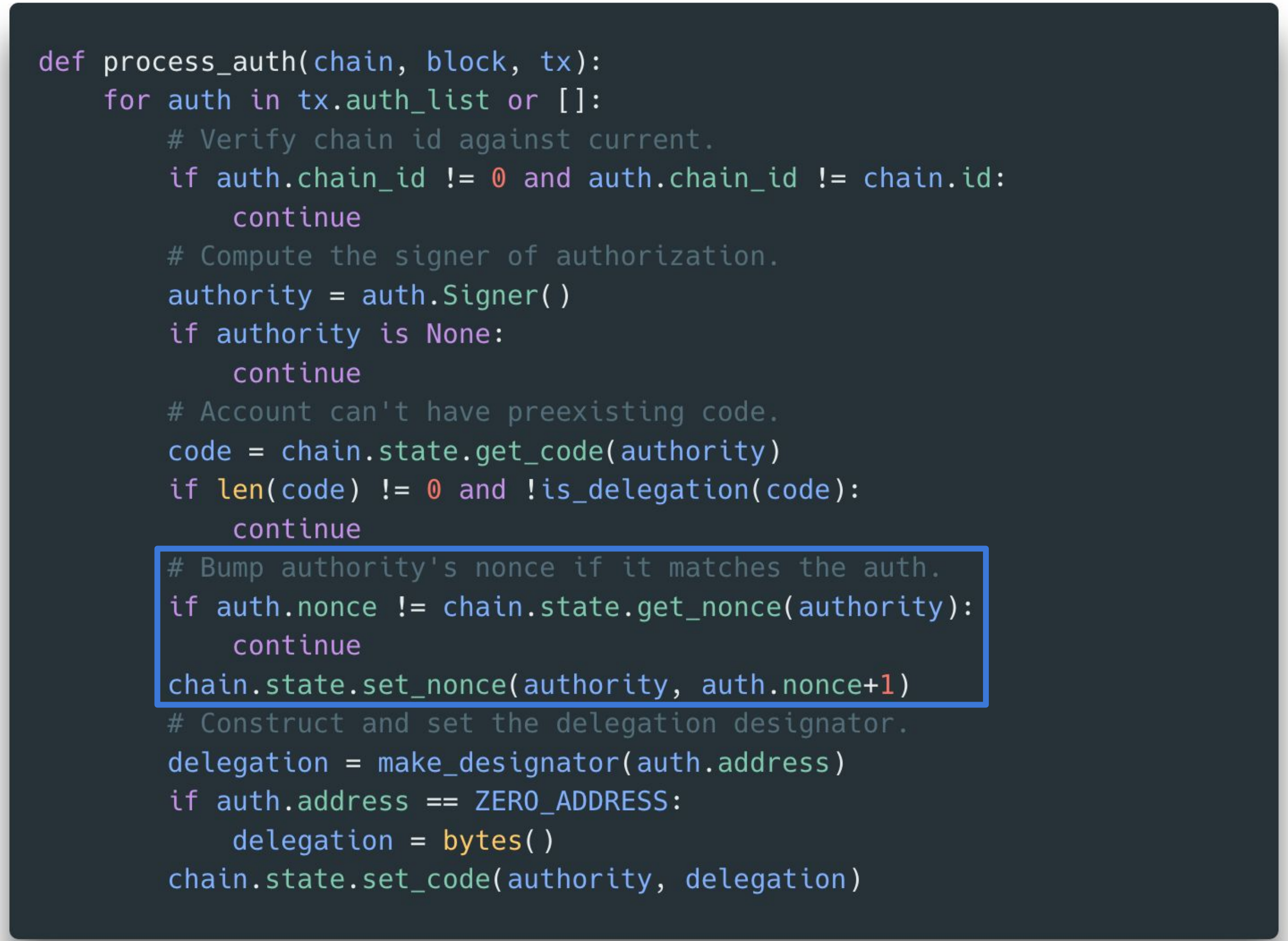
```
def process_auth(chain, block, tx):
    for auth in tx.auth_list or []:
        # Verify chain id against current.
        if auth.chain_id != 0 and auth.chain_id != chain.id:
            continue
        # Compute the signer of authorization.
        authority = auth.Signer()
        if authority is None:
            continue
        # Account can't have preexisting code.
        code = chain.state.get_code(authority)
        if len(code) != 0 and !is_delegation(code):
            continue
        # Bump authority's nonce if it matches the auth.
        if auth.nonce != chain.state.get_nonce(authority):
            continue
        chain.state.set_nonce(authority, auth.nonce+1)
        # Construct and set the delegation designator.
        delegation = make_designator(auth.address)
        if auth.address == ZERO_ADDRESS:
            delegation = bytes()
        chain.state.set_code(authority, delegation)
```



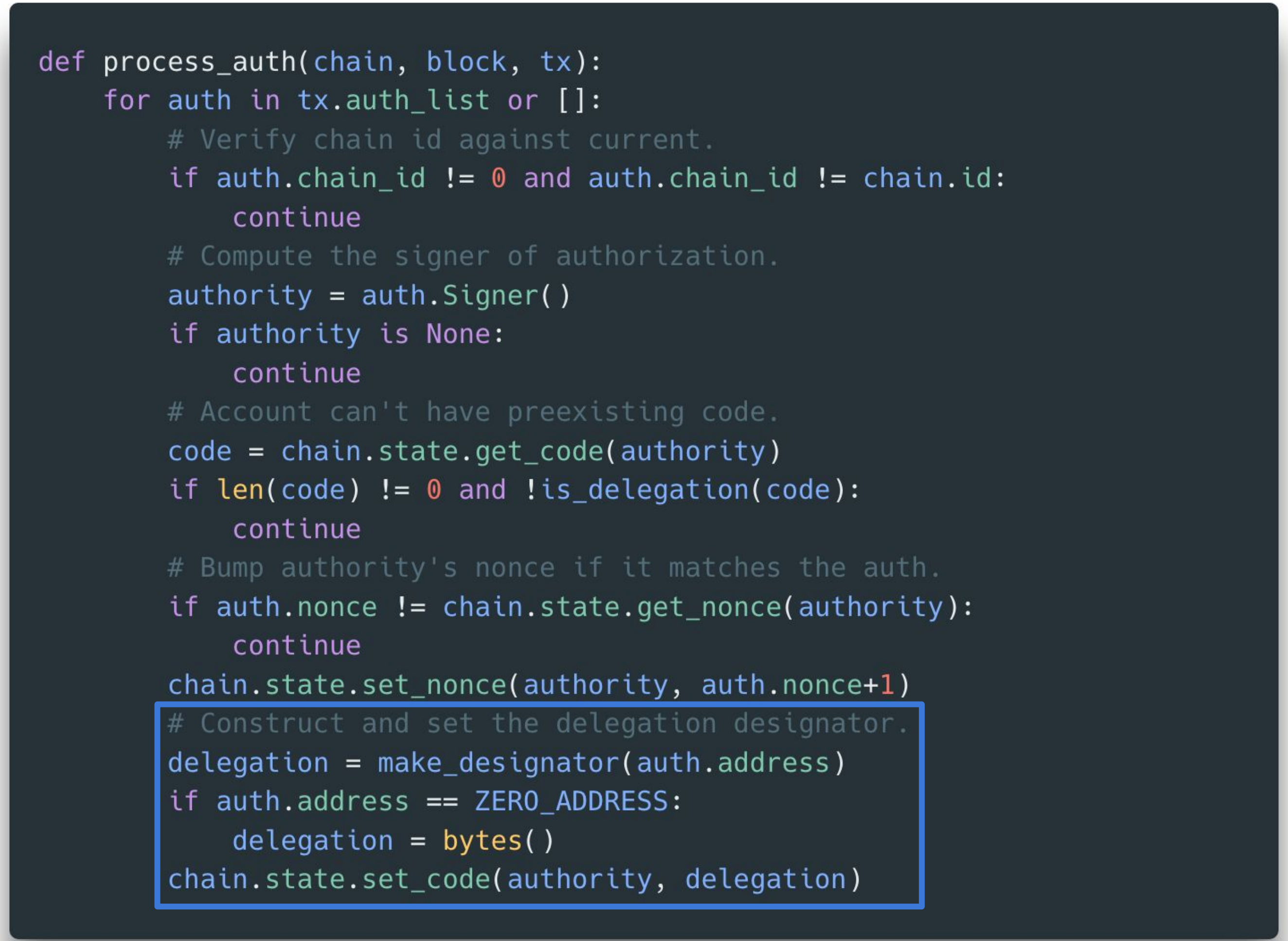
```
def process_auth(chain, block, tx):
    for auth in tx.auth_list or []:
        # Verify chain id against current.
        if auth.chain_id != 0 and auth.chain_id != chain.id:
            continue
        # Compute the signer of authorization.
        authority = auth.Signer()
        if authority is None:
            continue
        # Account can't have preexisting code.
        code = chain.state.get_code(authority)
        if len(code) != 0 and !is_delegation(code):
            continue
        # Bump authority's nonce if it matches the auth.
        if auth.nonce != chain.state.get_nonce(authority):
            continue
        chain.state.set_nonce(authority, auth.nonce+1)
        # Construct and set the delegation designator.
        delegation = make_designator(auth.address)
        if auth.address == ZERO_ADDRESS:
            delegation = bytes()
        chain.state.set_code(authority, delegation)
```



```
def process_auth(chain, block, tx):
    for auth in tx.auth_list or []:
        # Verify chain id against current.
        if auth.chain_id != 0 and auth.chain_id != chain.id:
            continue
        # Compute the signer of authorization.
        authority = auth.Signer()
        if authority is None:
            continue
        # Account can't have preexisting code.
        code = chain.state.get_code(authority)
        if len(code) != 0 and !is_delegation(code):
            continue
        # Bump authority's nonce if it matches the auth.
        if auth.nonce != chain.state.get_nonce(authority):
            continue
        chain.state.set_nonce(authority, auth.nonce+1)
        # Construct and set the delegation designator.
        delegation = make_designator(auth.address)
        if auth.address == ZERO_ADDRESS:
            delegation = bytes()
        chain.state.set_code(authority, delegation)
```

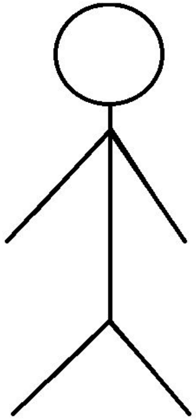


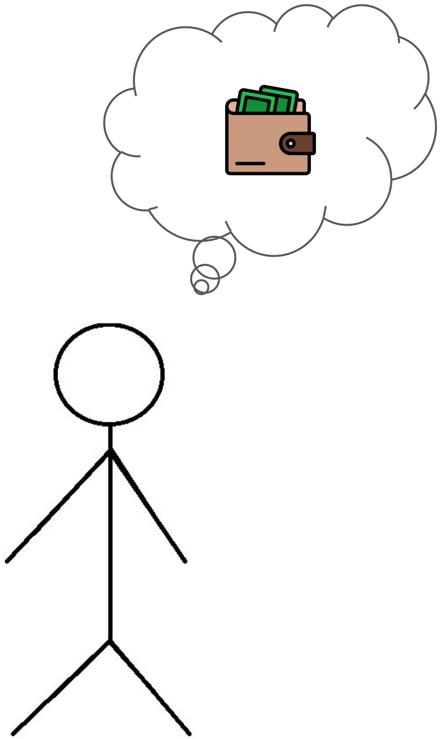
```
def process_auth(chain, block, tx):
    for auth in tx.auth_list or []:
        # Verify chain id against current.
        if auth.chain_id != 0 and auth.chain_id != chain.id:
            continue
        # Compute the signer of authorization.
        authority = auth.Signer()
        if authority is None:
            continue
        # Account can't have preexisting code.
        code = chain.state.get_code(authority)
        if len(code) != 0 and !is_delegation(code):
            continue
        # Bump authority's nonce if it matches the auth.
        if auth.nonce != chain.state.get_nonce(authority):
            continue
        chain.state.set_nonce(authority, auth.nonce+1)
        # Construct and set the delegation designator.
        delegation = make_designator(auth.address)
        if auth.address == ZERO_ADDRESS:
            delegation = bytes()
        chain.state.set_code(authority, delegation)
```

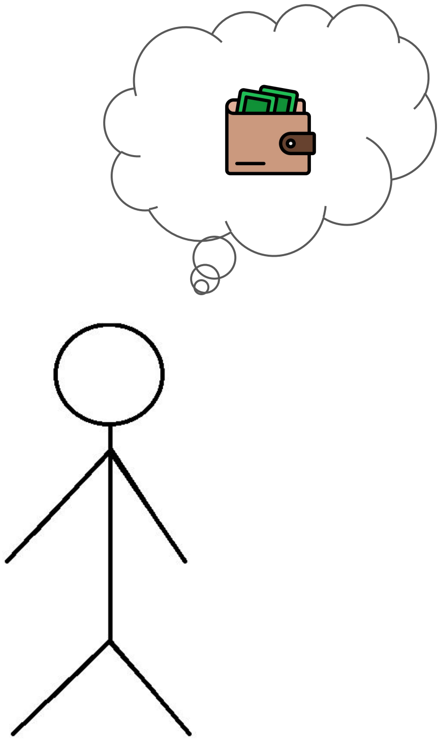


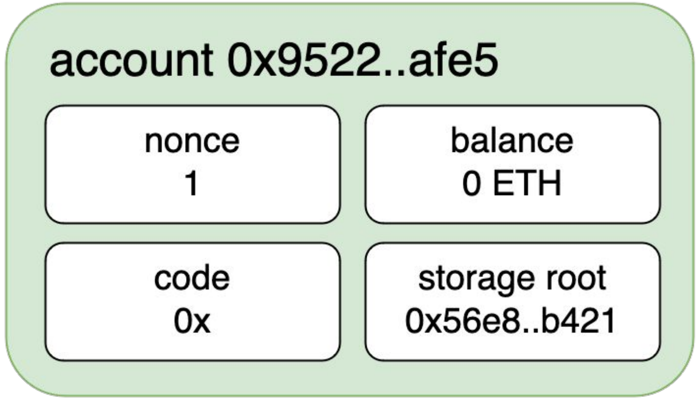
```
def process_auth(chain, block, tx):
    for auth in tx.auth_list or []:
        # Verify chain id against current.
        if auth.chain_id != 0 and auth.chain_id != chain.id:
            continue
        # Compute the signer of authorization.
        authority = auth.Signer()
        if authority is None:
            continue
        # Account can't have preexisting code.
        code = chain.state.get_code(authority)
        if len(code) != 0 and !is_delegation(code):
            continue
        # Bump authority's nonce if it matches the auth.
        if auth.nonce != chain.state.get_nonce(authority):
            continue
        chain.state.set_nonce(authority, auth.nonce+1)
        # Construct and set the delegation designator.
        delegation = make_designator(auth.address)
        if auth.address == ZERO_ADDRESS:
            delegation = bytes()
        chain.state.set_code(authority, delegation)
```

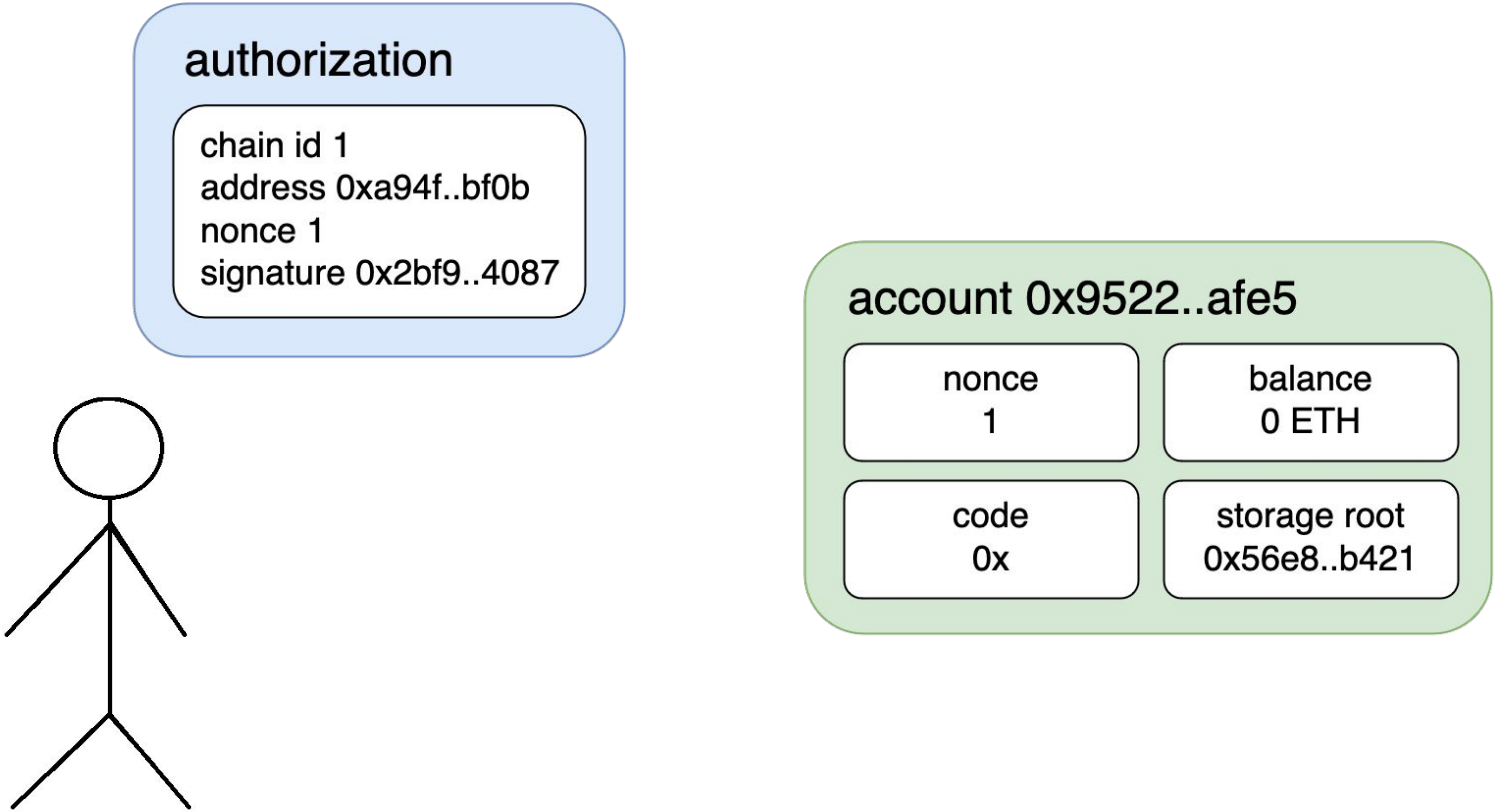
example





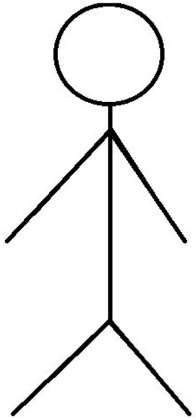


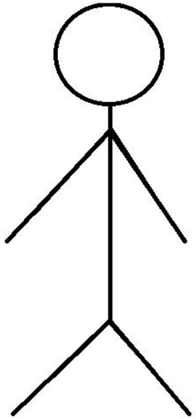

account 0x9522..afe5
nonce
1
balance
0 ETH
code
0x
storage root
0x56e8..b421


authorization
chain id 1
address 0xa94f..bf0b
nonce 1
signature 0x2bf9..4087
account 0x9522..afe5
nonce
1
balance
0 ETH
code
0x
storage root
0x56e8..b421

authorization

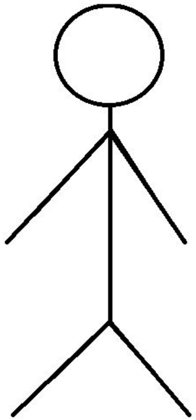
chain id 1
address 0xa94f..bf0b
nonce 1
signature 0x2bf9..4087

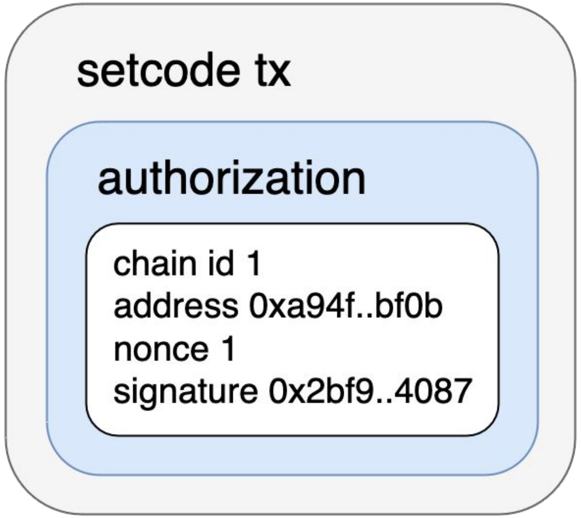




authorization

chain id 1
address 0xa94f..bf0b
nonce 1
signature 0x2bf9..4087

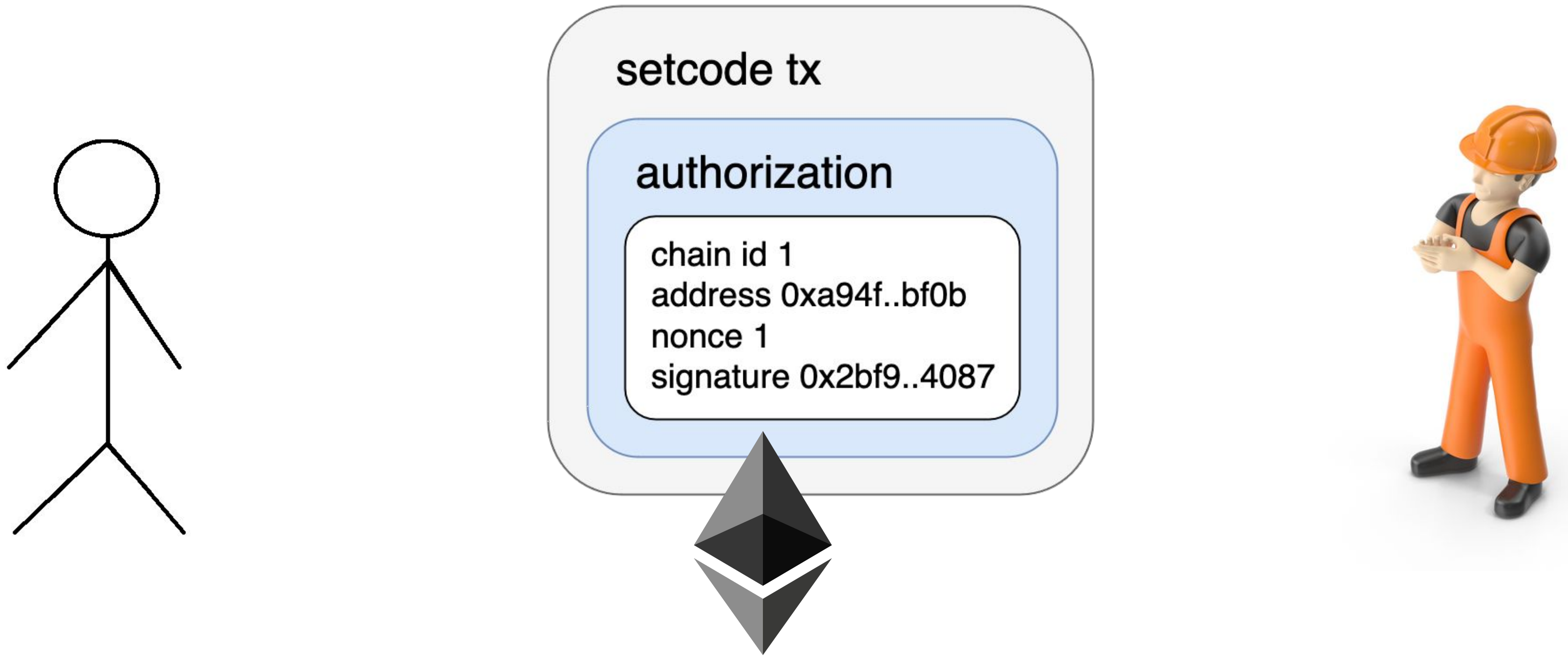


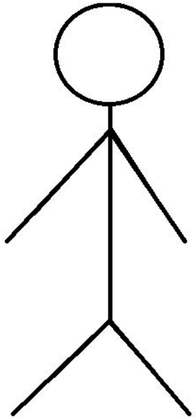


setcode tx

authorization

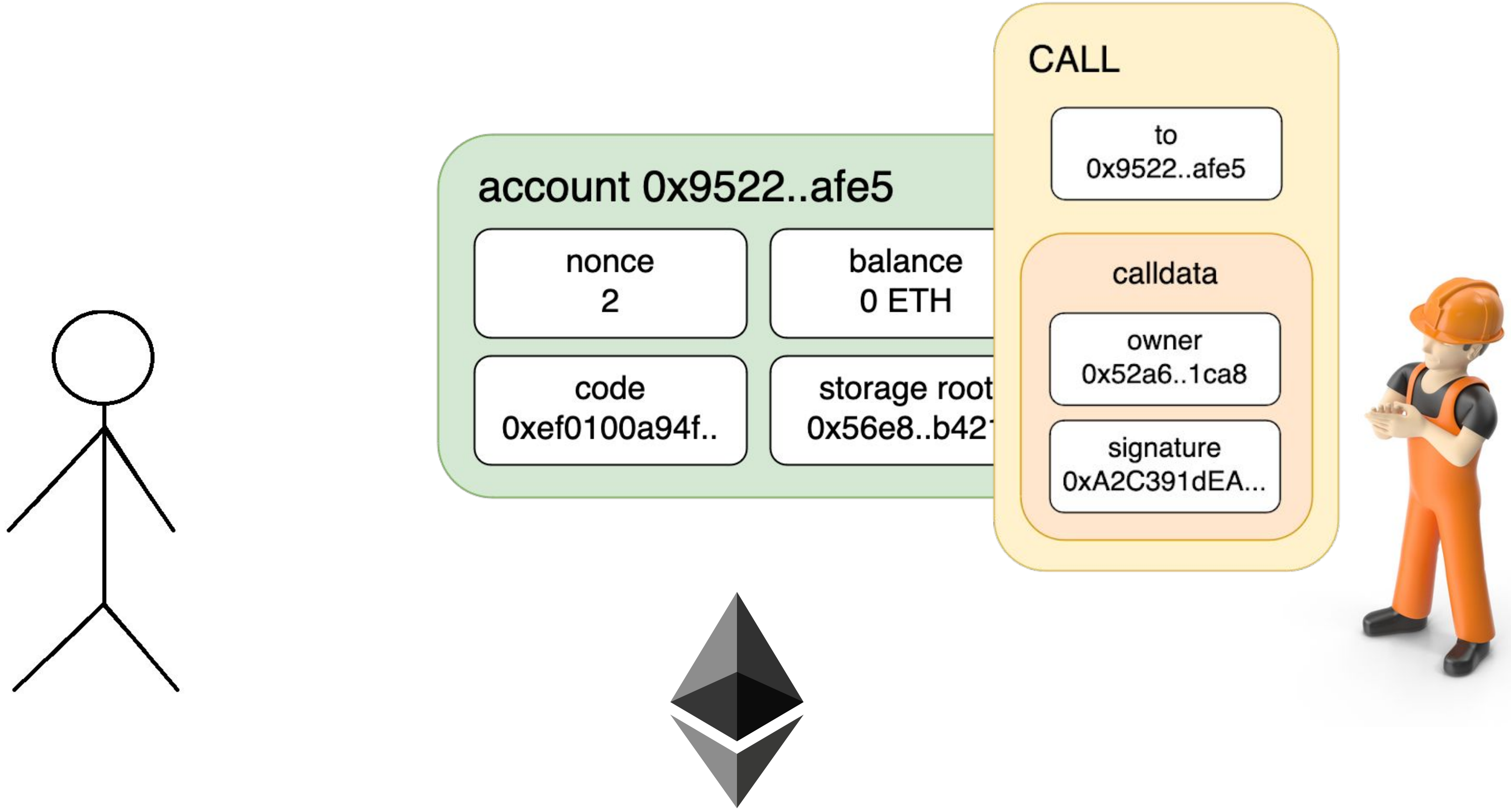
chain id 1
address 0xa94f..bf0b
nonce 1
signature 0x2bf9..4087

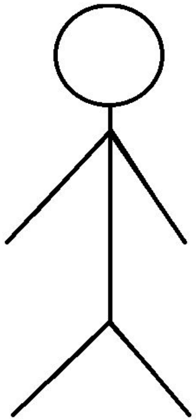

setcode tx
authorization
chain id 1
address 0xa94f..bf0b
nonce 1
signature 0x2bf9..4087

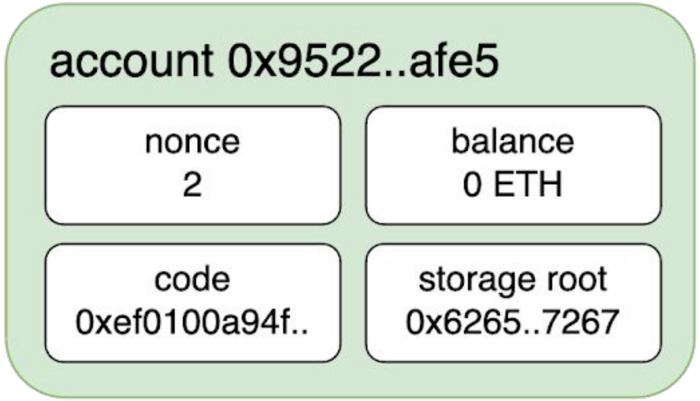


account 0x9522..afe5

nonce 2	balance 0 ETH
code 0xef0100a94f..	storage root 0x56e8..b421


account 0x9522..afe5
nonce
2
balance
0 ETH
code
0xef0100a94f..
storage root
0x56e8..b421
CALL
to
0x9522..afe5
calldata
owner
0x52a6..1ca8
signature
0xA2C391dEA...

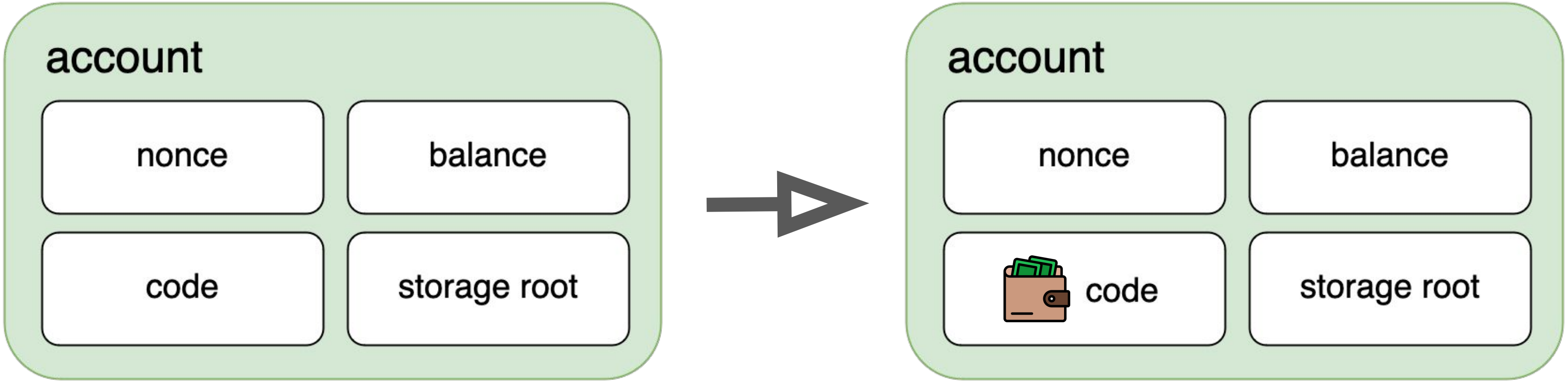



account 0x9522..afe5
nonce
2
balance
0 ETH
code
0xef0100a94f..
storage root
0x6265..7267

gas costs

- intrinsic costs is extended to charge 25k gas per each authorization
    - if the account already exists, a refund of 12.5k gas is issued
    - refund is subject to global refund mechanics, i.e. 20% of the total gas used by the tx
- if tx destination is has a delegation designator, the target of the delegation is warmed
- during evm execution, if an account with a delegation designator is executed it is charged to warm both the target and the delegation target (if either is cold)

Prague (tbd, early 2025)



what comes next

- lot’s of talks and events this week on EIP-7702 x AA
- experiment!
  - rolling devnets with 7702
  - ithaca exp-0001
- contribute to application standards
  - need to formalize communication pattern dapp <> wallet that supports this new functionality

questions?

@lightclients