zk-promises: Making zero-knowledge objects accept the call for banning and reputation

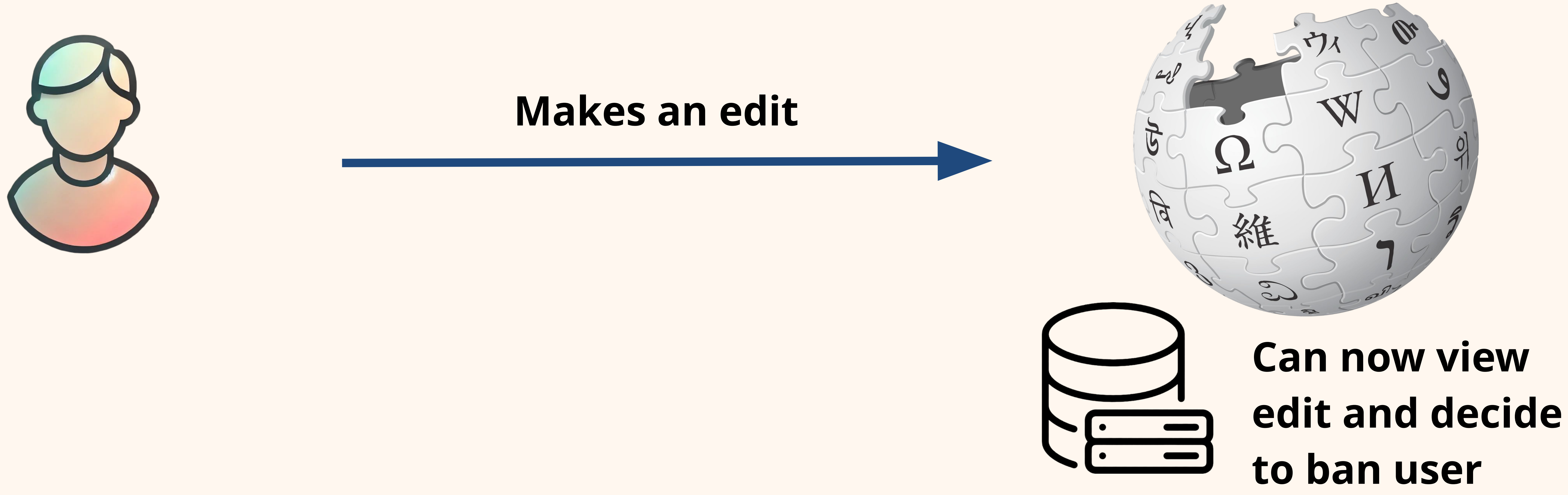
Maurice Shih, Michael Rosenberg, Hari Kailad, Ian Miers

Example: Wikipedia

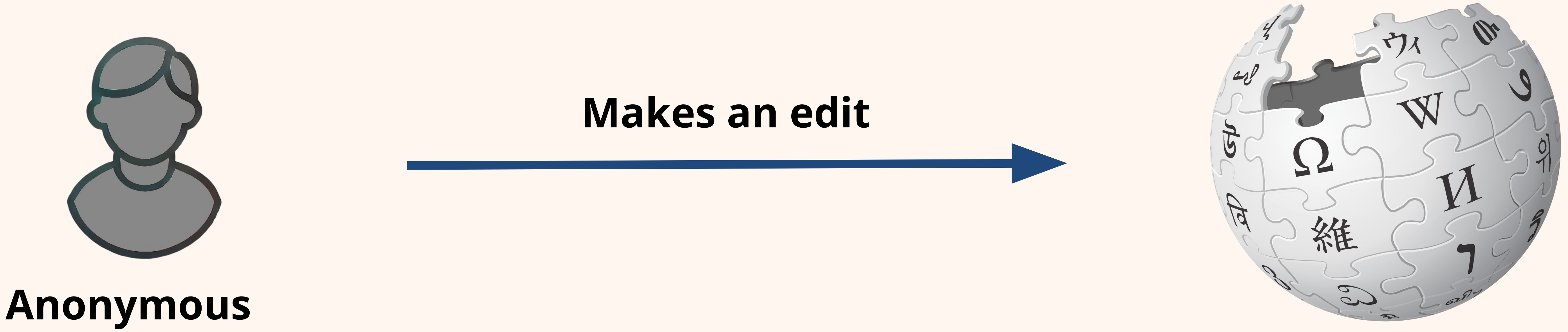
Example: Wikipedia

Makes an edit

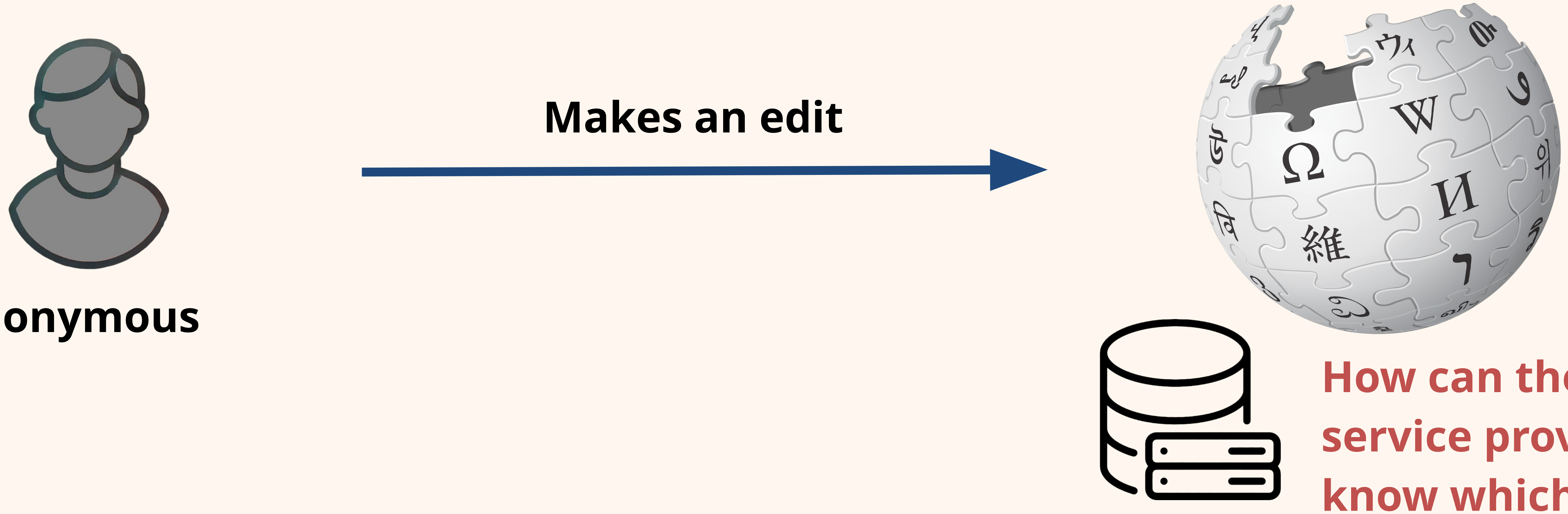
Example: Wikipedia



Example: Wikipedia

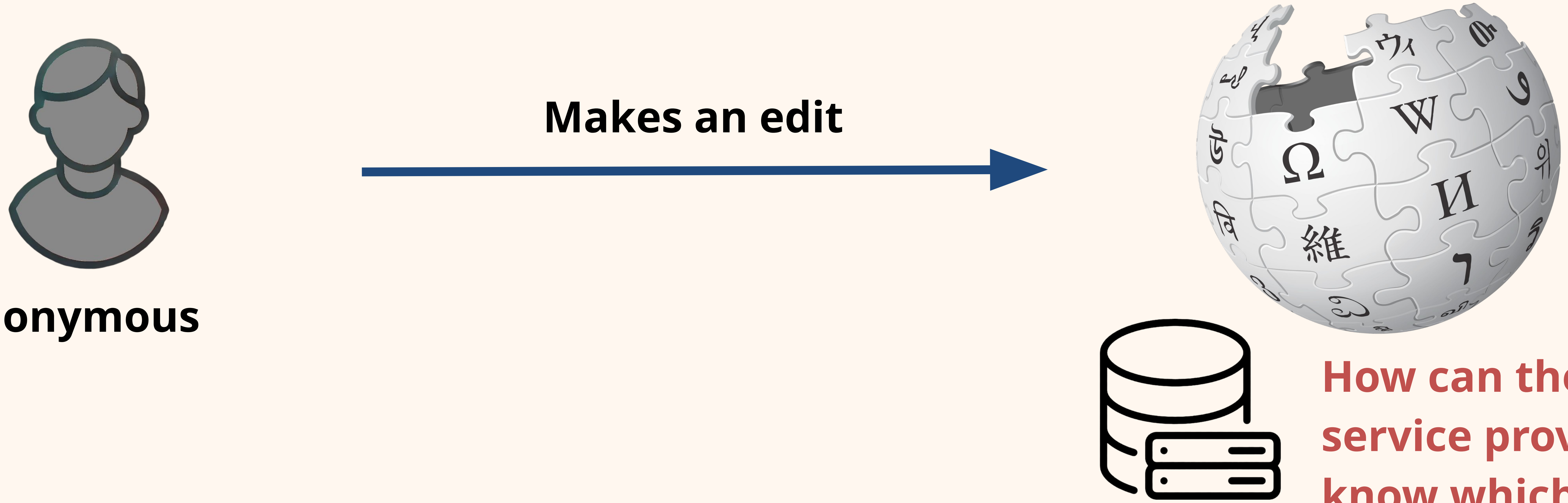


Example: Wikipedia



How can the service provider know which user to ban?

Example: Wikipedia

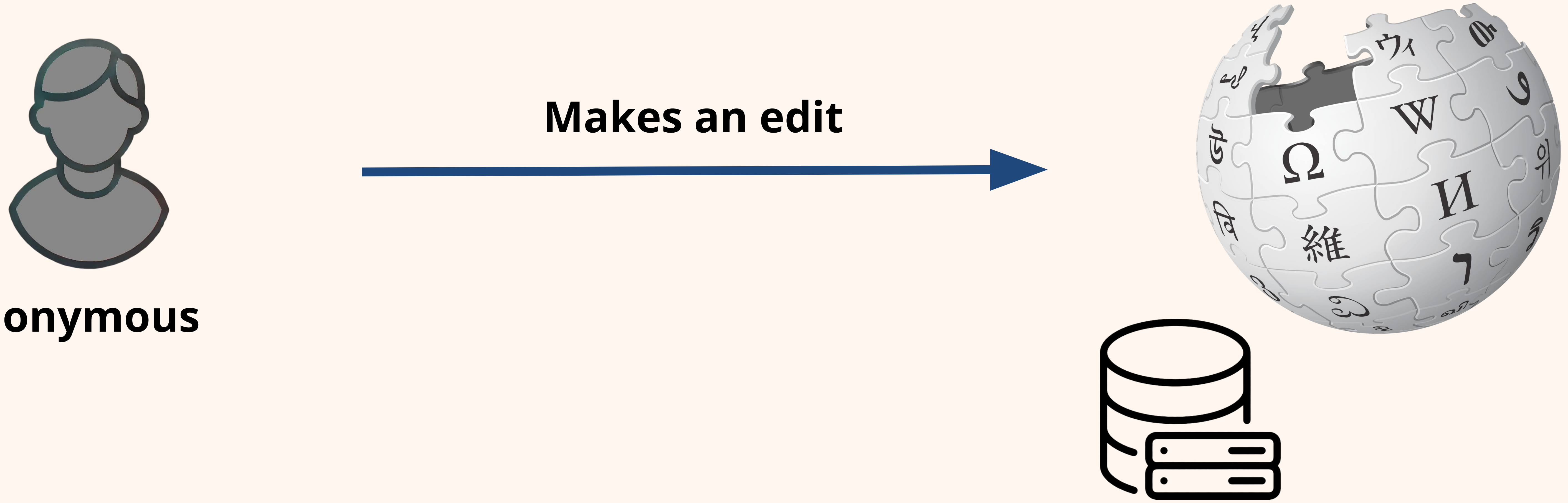


How can the service provider know which user to ban?

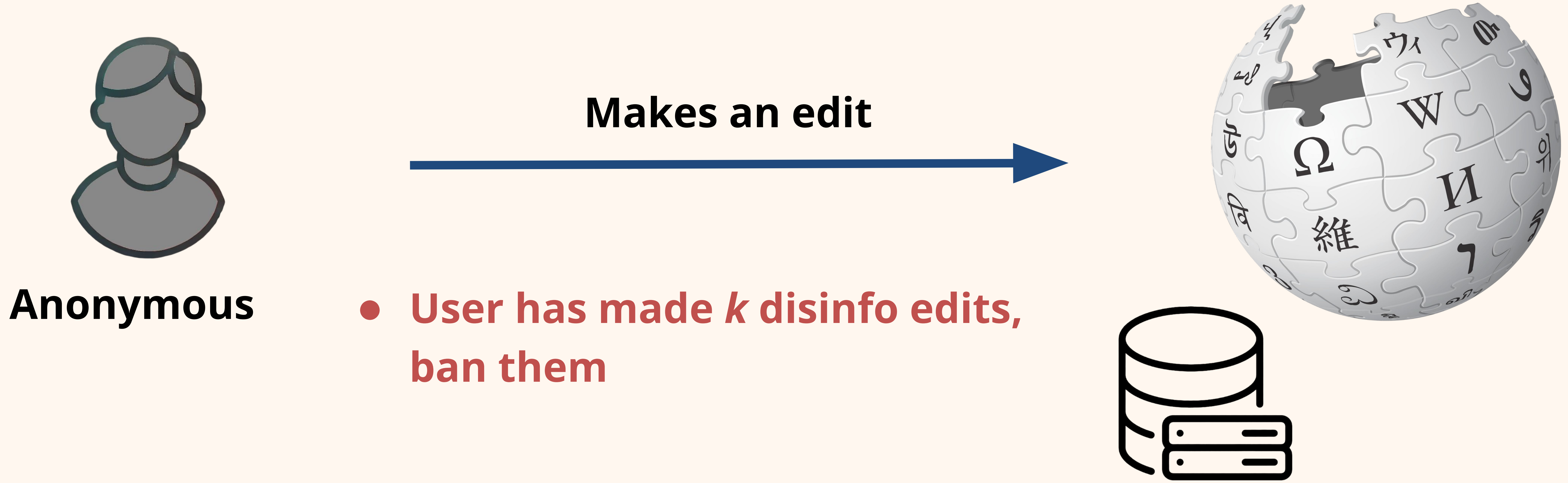
Answer: Anonymous Blocklisting

Complex feedback?

Complex feedback?

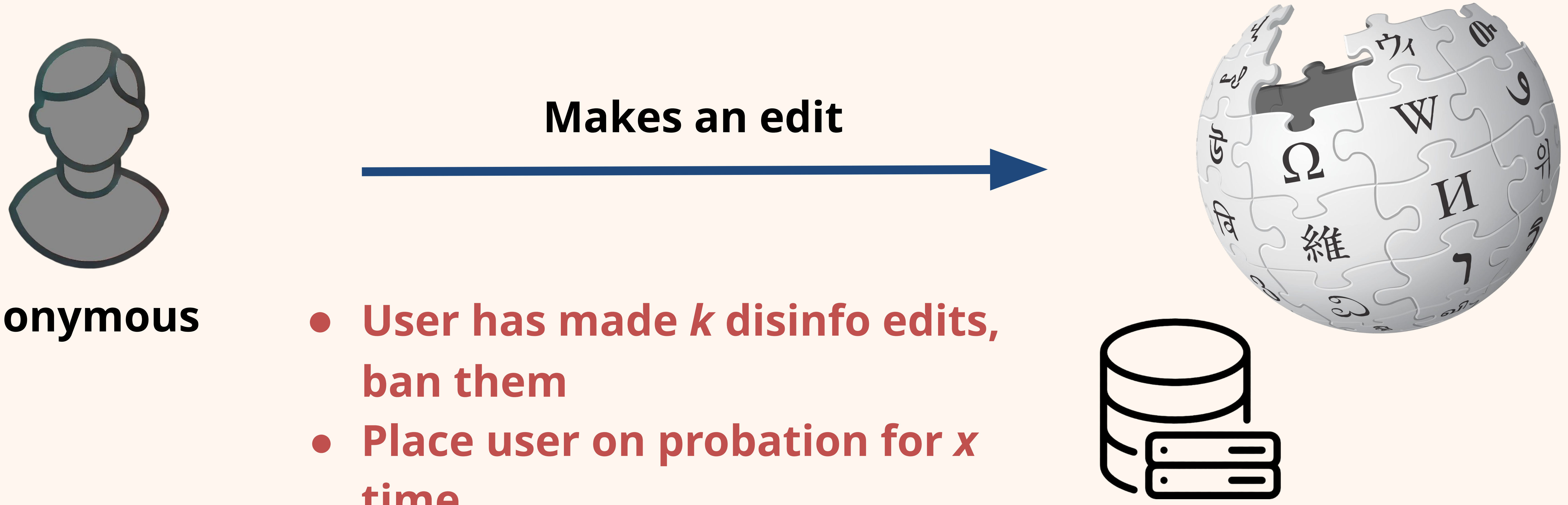


Complex feedback?



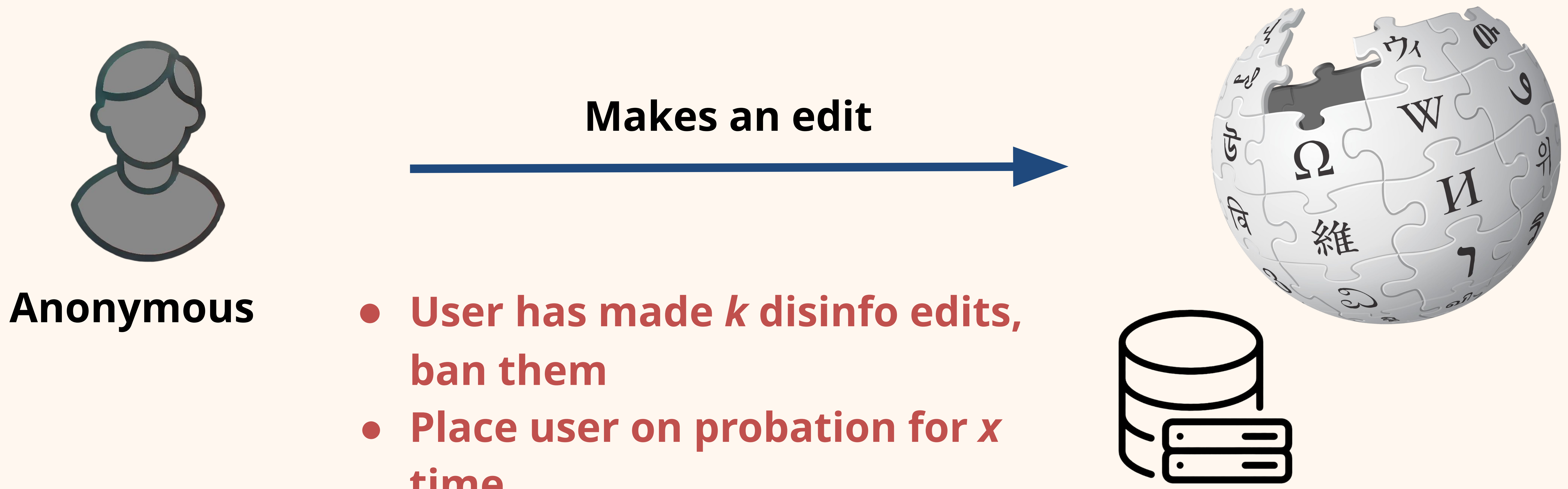
- User has made *k* disinfo edits, ban them

Complex feedback?



- User has made *k* disinfo edits, ban them
- Place user on probation for *x* time

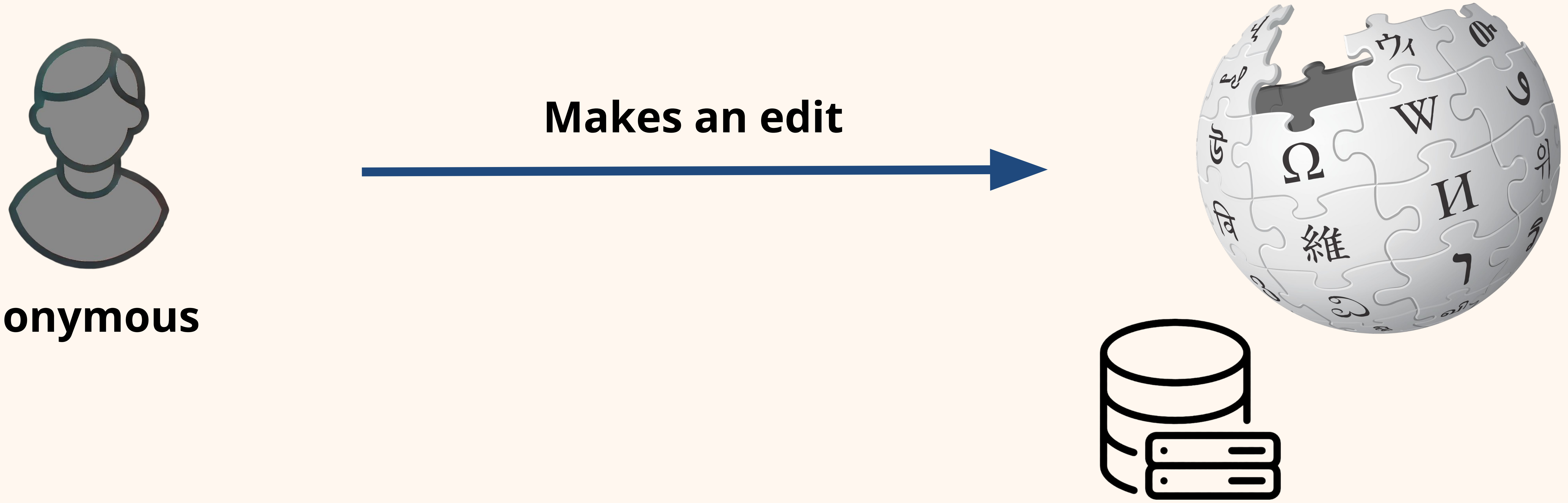
Complex feedback?



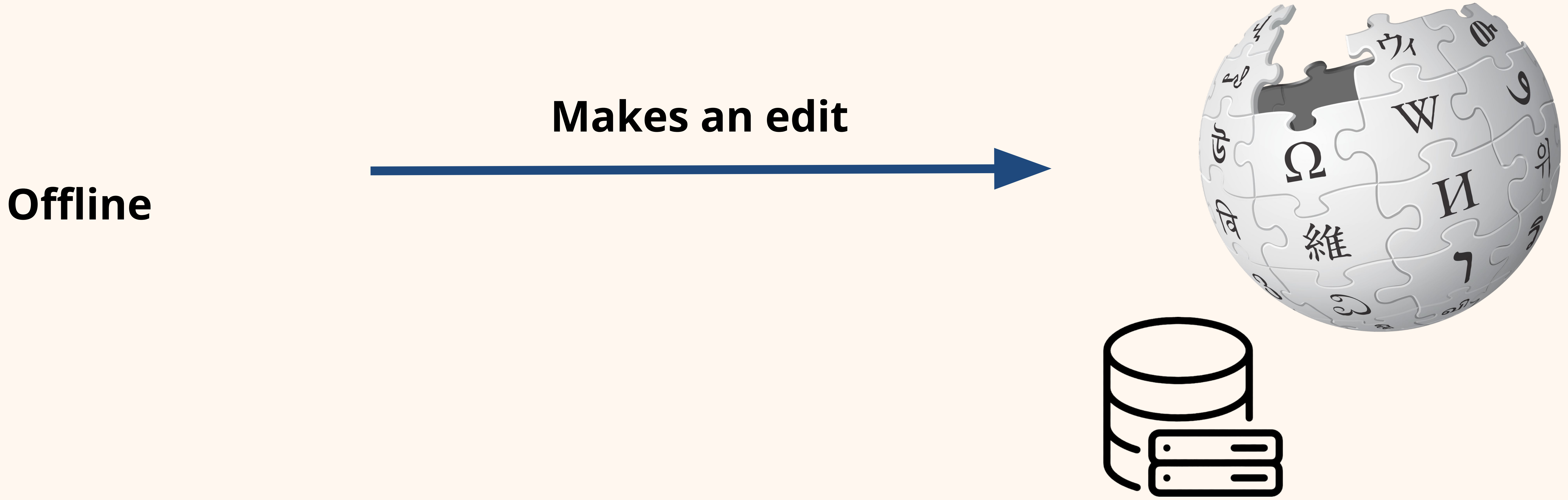
- User has made *k* disinfo edits, ban them
- Place user on probation for *x* time
- If the user has made *k* good edits, increase their number of edits per day

Asynchronous?

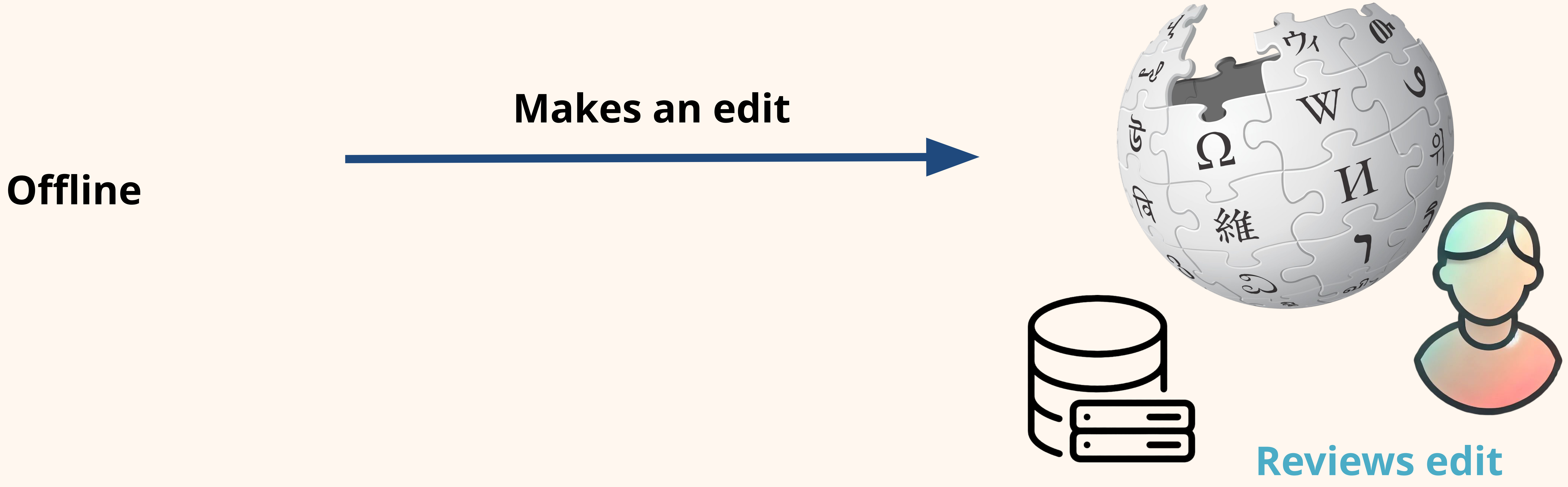
Asynchronous?



Asynchronous?

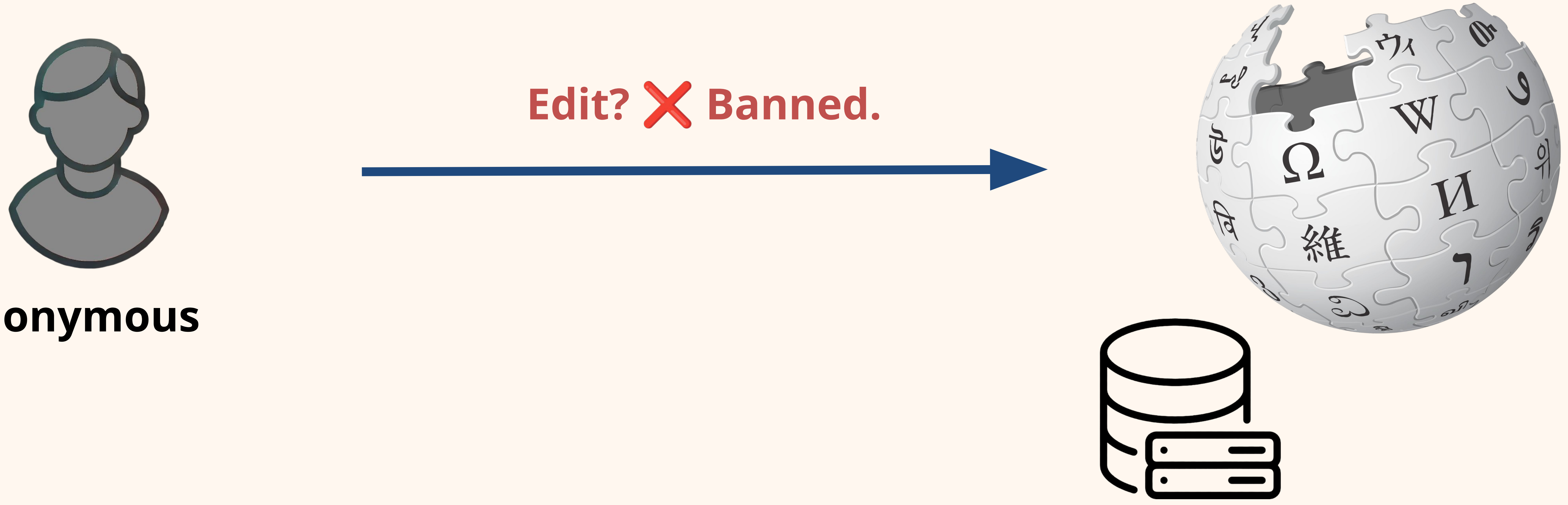


Asynchronous?



User has made *k* disinfo edits, ban them

Asynchronous?



Prior Work

PEREA
- Limited functionality: No complex feedback
- Fixed parameters for all users
- **Moderation halts** until oldest are processed

SnarkBlock
- Built only for blocklisting
- Doesn't support complex state

BLAC
- Does not support programmable logic or multidimensional state, only a simple counter
- Not asynchronous
- Global halting
- User does **linear** work in its actions

SMART
- State must be small
- Does not support arbitrary updates

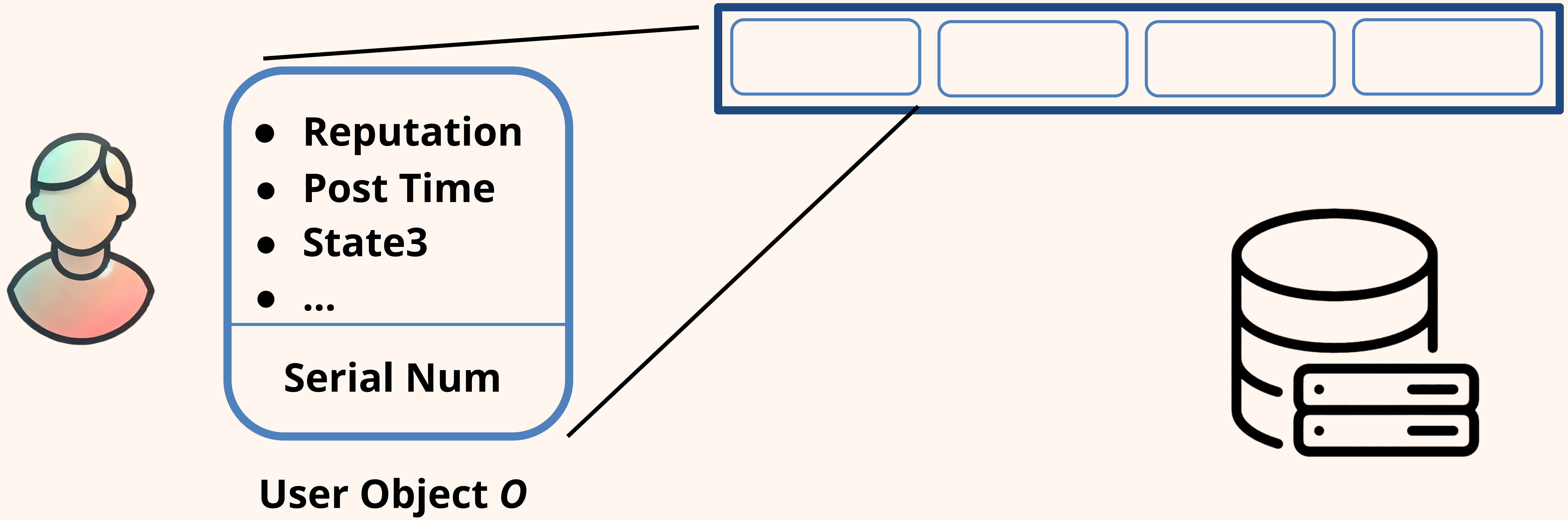
zk-promises

- Anonymous state
- Fully programmable logic and complex feedback
- Completely asynchronous
- Using zkSNARKs!

Can use for...

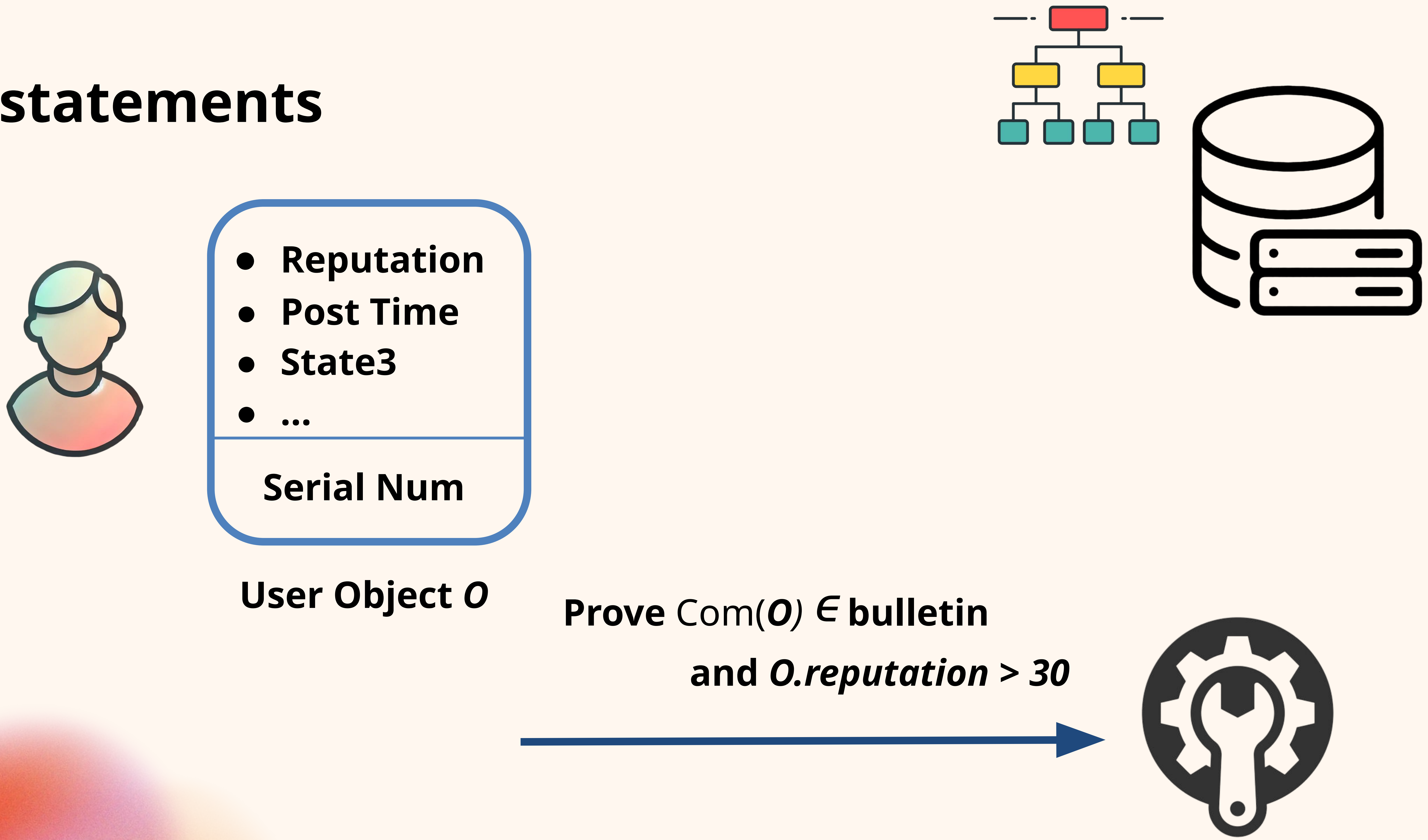
- Forums with moderation
- Oblivious VPNs (Apple, Cloudflare)
- Whistleblowing applications
- Cryptocurrency reputation

zk-objects



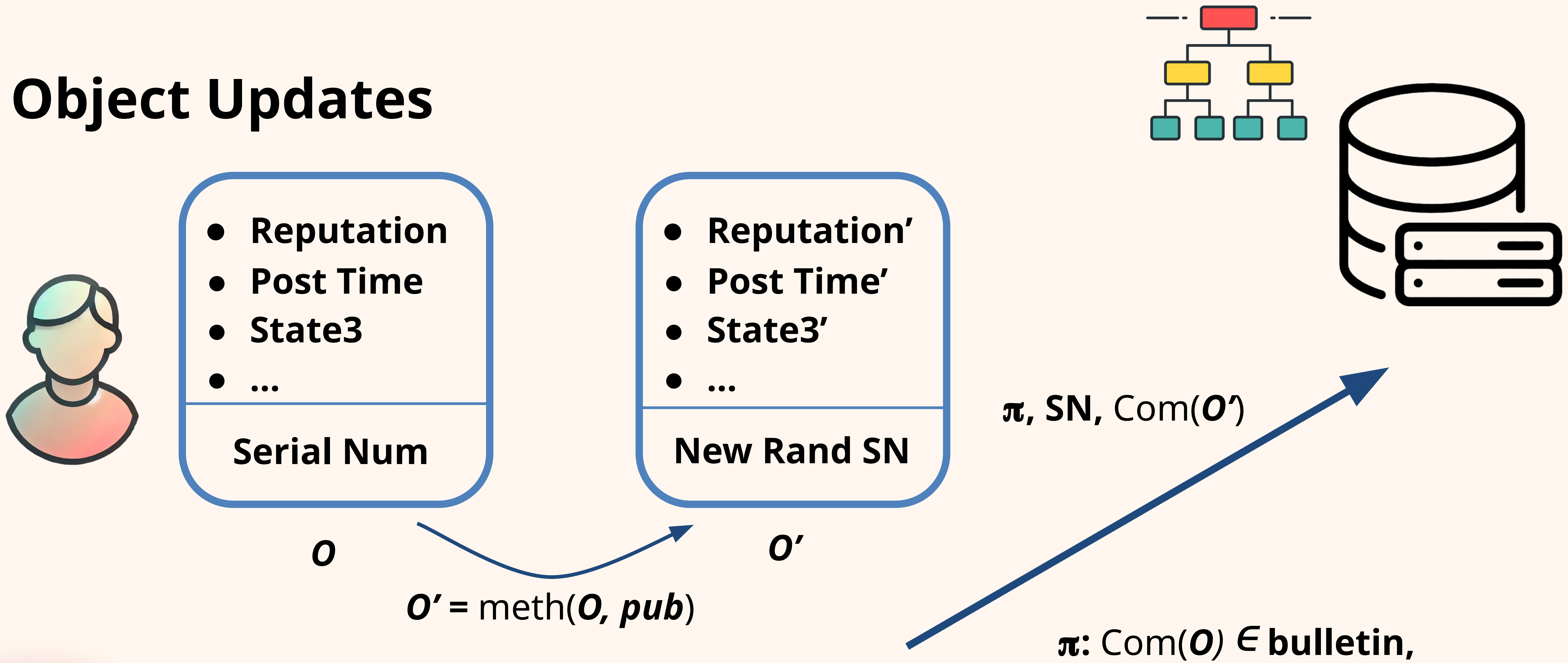
Commitments to Objects
Reputation
Post Time
State3
...
Serial Num
User Object O

Prove statements



- Reputation
- Post Time
- State3
- ...
Serial Num
User Object *O*
Prove Com(*O*) ∈ **bulletin**
and ***O*.reputation > 30**

Object Updates



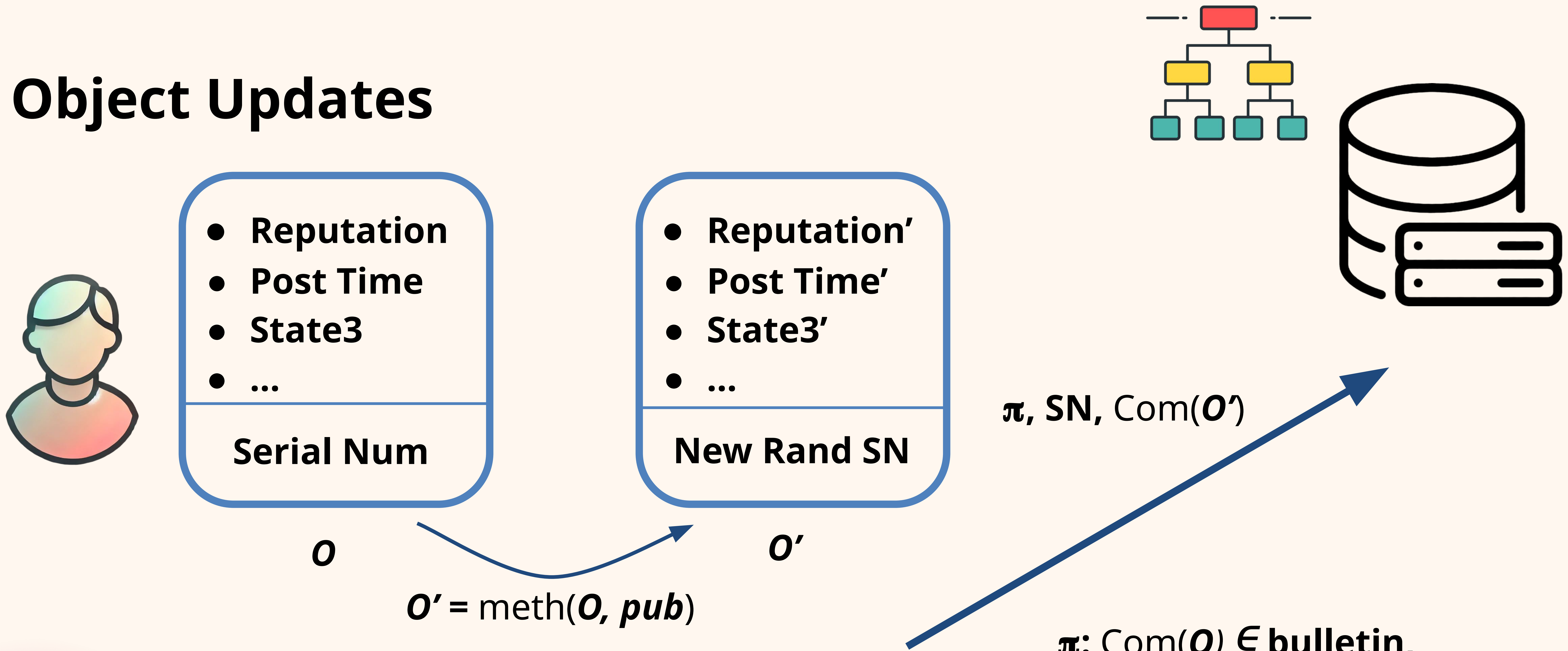
π: Com(O) $\in$ **bulletin**,

O.serial == SN,

Com(O') == Com(O')

$\Phi(O, O') == 1$

Object Updates



Only the object owner can call this method.

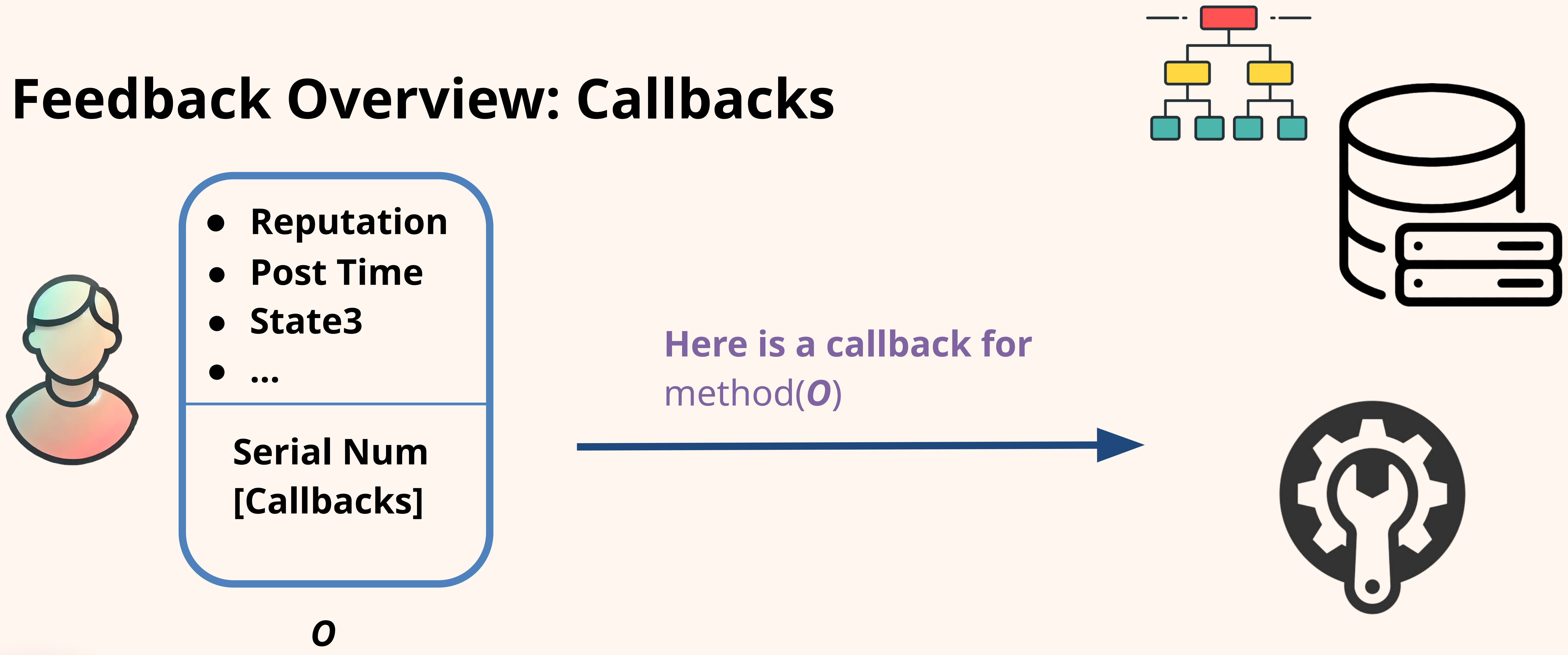
π: Com(O) $\in$ **bulletin**,

$O.serial == SN$,

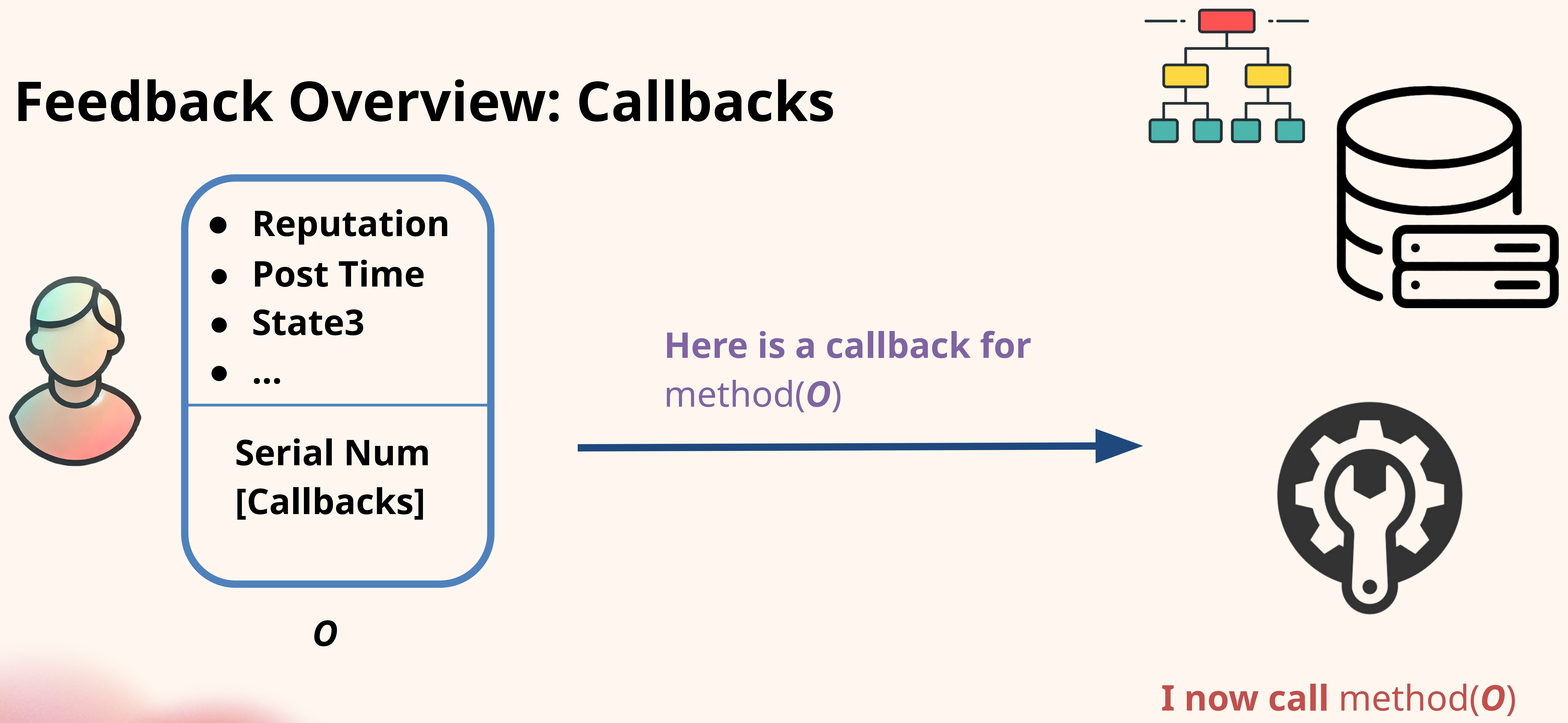
Com(O') == Com(O')

$\Phi(O, O') == 1$

Feedback Overview: Callbacks

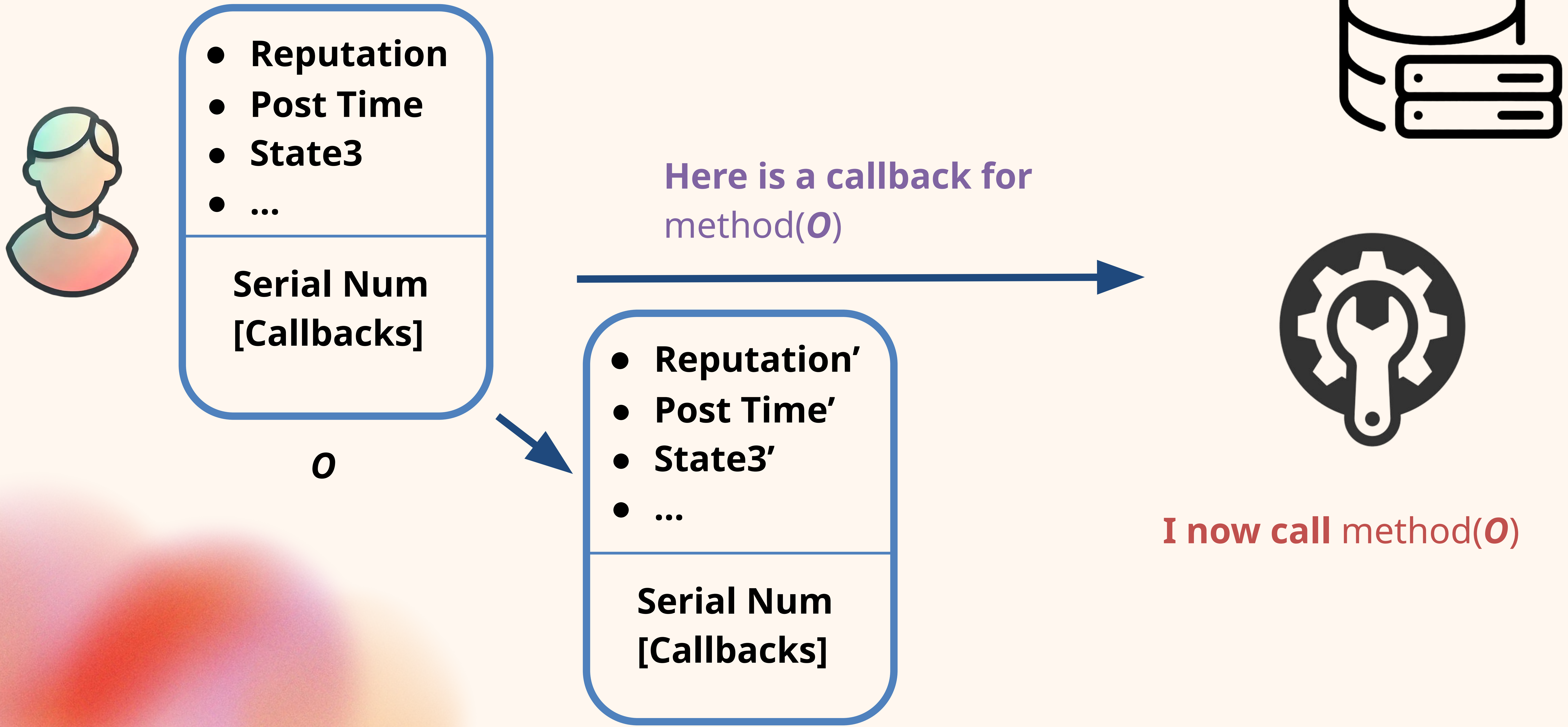


Feedback Overview: Callbacks



Reputation
Post Time
State3
...
Serial Num
[Callbacks]
O
Here is a callback for method(***O***)
I now call method(***O***)

Feedback Overview: Callbacks



zk-promises: Base construction (Create)

- Reputation
- Post Time
- State3
- …

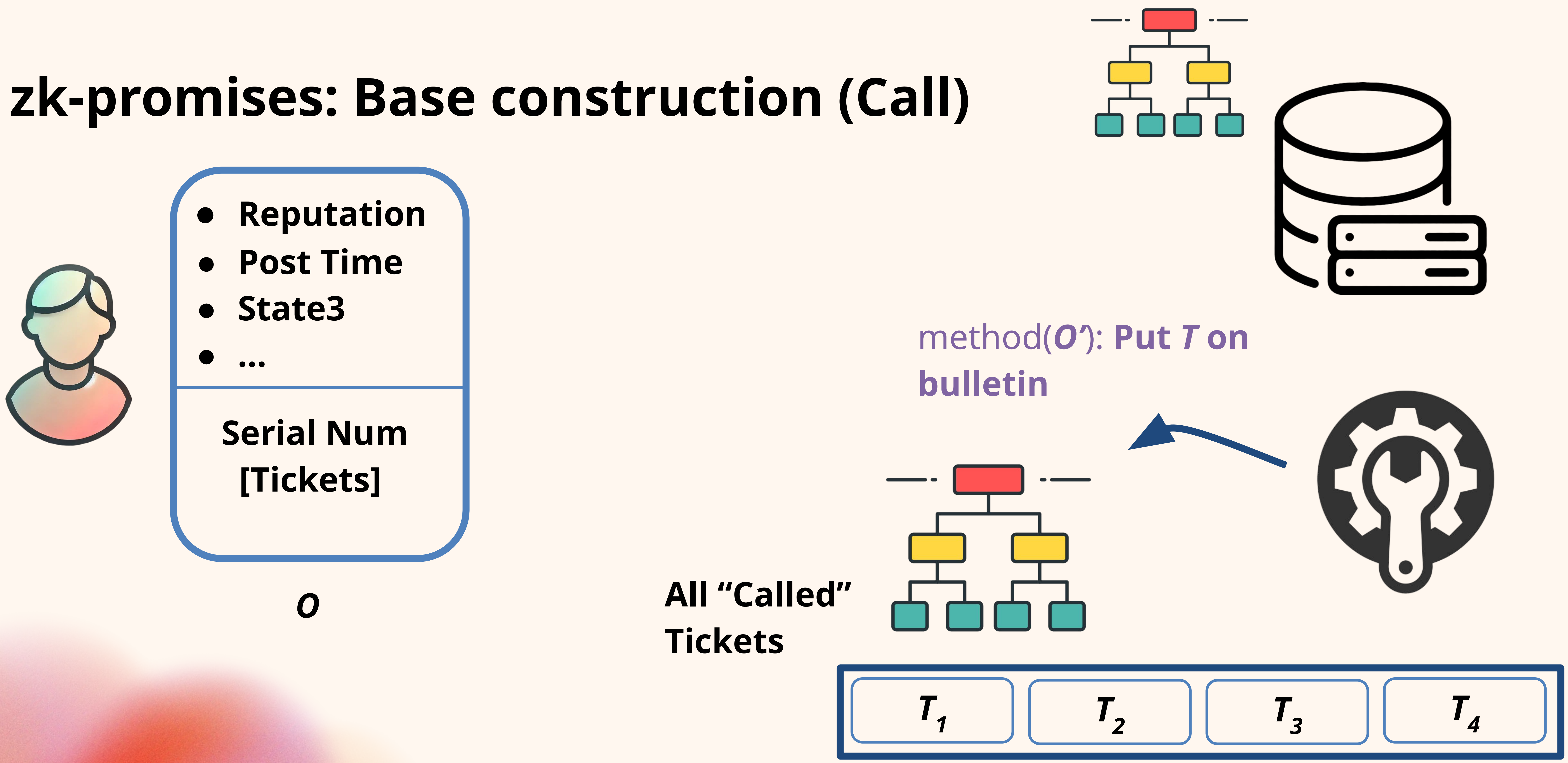
Serial Num
[Tickets]

O

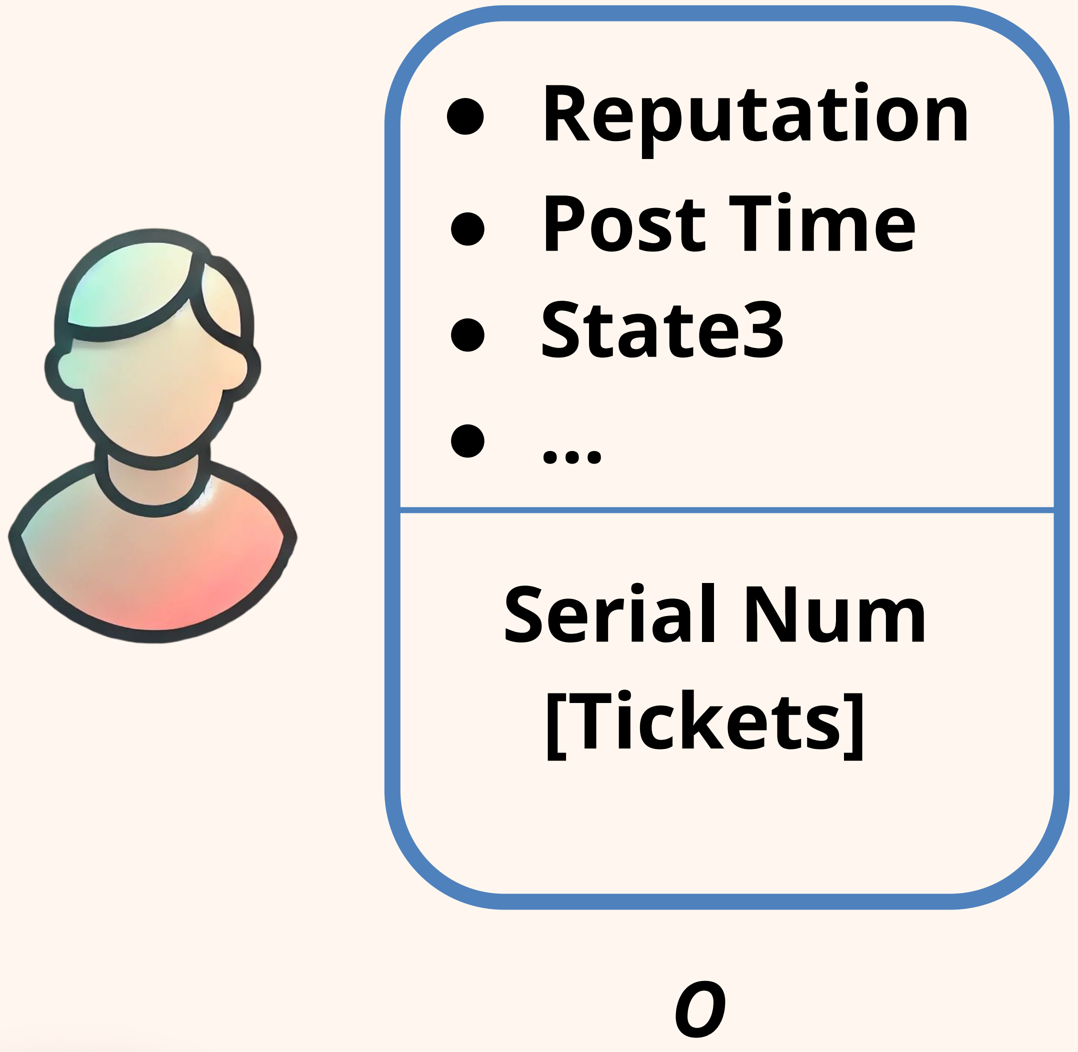
Here is a ticket T,
π, SN, Com(O')

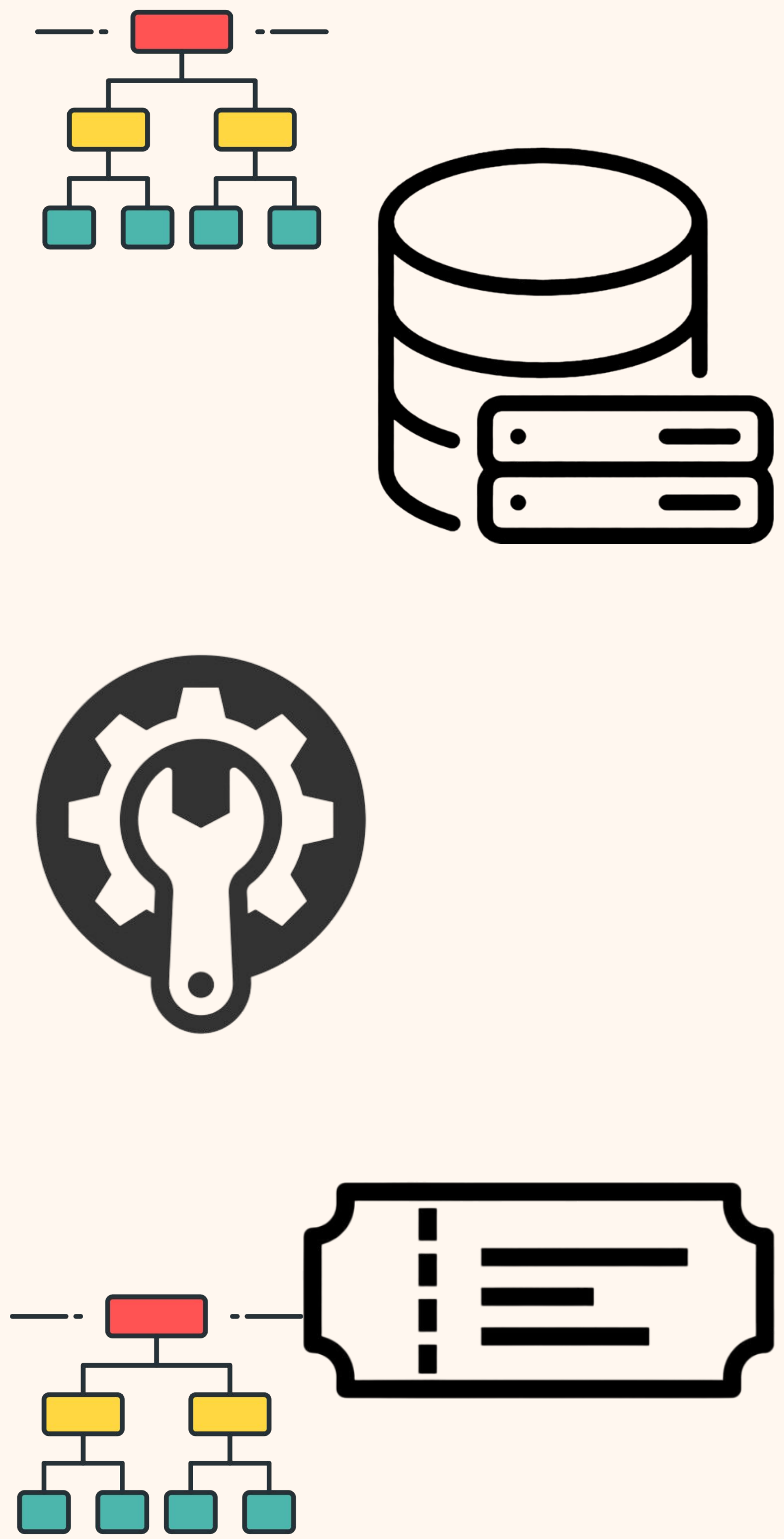
π: Com(O) $\in$ **bulletin,**
$O.serial == SN$,
Com(O') == Com(O')
$\Phi(O, O') == 1$
$O'.ticket\_list = O.ticket\_list + T$

zk-promises: Base construction (Call)



zk-promises: Base construction (Settle)





zk-promises: Base construction (Settle)

- Reputation
- Post Time
- State3
- …

Serial Num
[Tickets]

O

π: For $T \in$ [Tickets],

zk-promises: Base construction (Settle)

- **Reputation**
- **Post Time**
- **State3**
- **…**

Serial Num
[Tickets]

O

π: For *T* ∈ [Tickets],

- ***T* ∈ callback bulletin,**
  Removed *T* from [Tickets]
  O' **=** method(***O***)

zk-promises: Base construction (Settle)

- Reputation
- Post Time
- State3
- ...

Serial Num
[Tickets]

$\boldsymbol{O}$

π: For *T* ∈ [Tickets],

- ***T* ∈ callback bulletin,**
  Removed *T* from [Tickets]
  O'* =** method(O***)
- ***T* ∉ callback bulletin,**
  ***T* remains in [Tickets]**
  O'* = *O

zk-promises: Base construction (Settle)

- Reputation
- Post Time
- State3
- …

Serial Num
[Tickets]

O

π: For *T* ∈ [Tickets],

- ***T* ∈ callback bulletin,**
  Removed *T* from [Tickets]
  ***O′* = method(*O*)**
- ***T* ∉ callback bulletin,**
  ***T* remains in [Tickets]**
  O′* = *O

Requires non-membership check

zk-promises: Base construction (Settle)

- Reputation
- Post Time
- State3
- ...

Serial Num
[Tickets]

O

π: For $T \in$ [Tickets],

- $T \in$ callback bulletin,
  Removed T from [Tickets]
  O' = method(O)
- $T \notin$ callback bulletin,
  T remains in [Tickets]
  $O' = O$

Requires non-membership check

How do we have a list?

Callback List

- Reputation
- Post Time
- State3
- ...

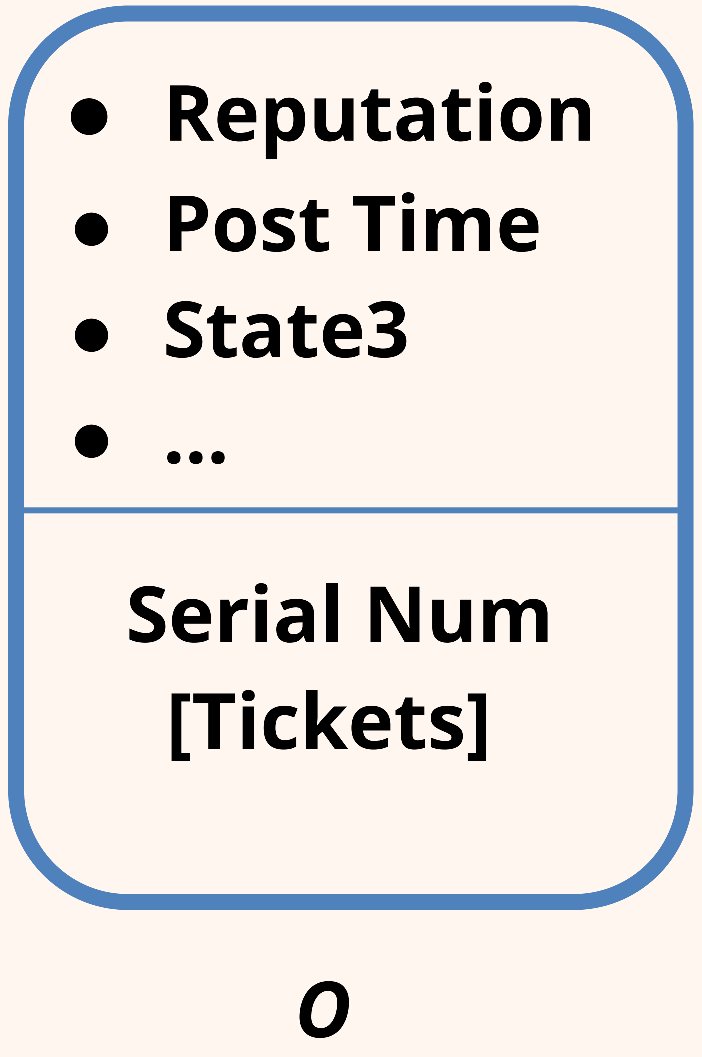
Serial Num
[Tickets]

O

List: Hash chain!

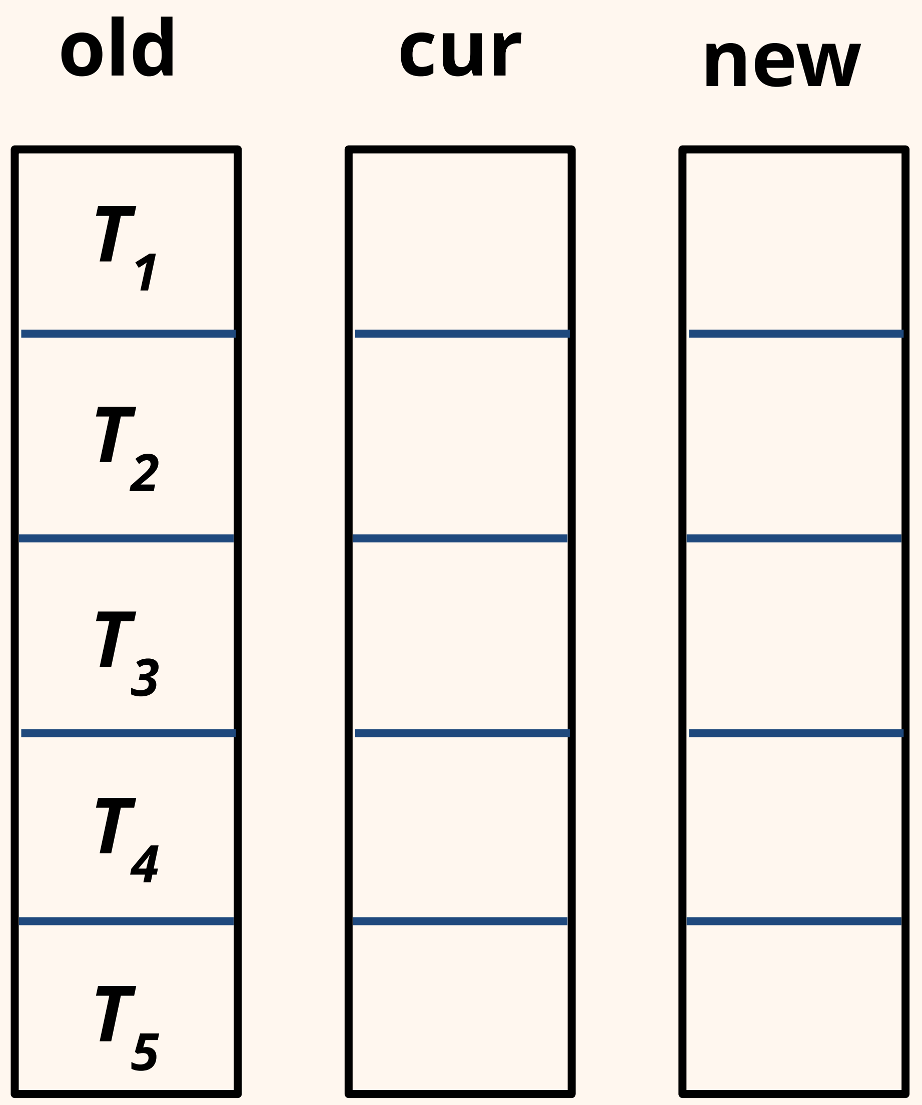
Concretely: [T_1 T_2 ...] is H(H(T_1), T_2)

Callback List



List: Hash chain!

Concretely: [T_1 T_2 ...] is H(H(T_1), T_2)



Callback List

- Reputation
- Post Time
- State3
- ...

Serial Num
[Tickets]

O

List: Hash chain!
Concretely: [T_1 T_2 ...] is H(H(T_1), T_2)

old	cur	new
T_1	T_1	T_1
T_2		
T_3		
T_4		
T_5		

Callback List

- Reputation
- Post Time
- State3
- ...

Serial Num
[Tickets]

O

List: Hash chain!
Concretely: $[T_1\ T_2 \ldots]$ is $H(H(T_1), T_2)$

old	cur	new
T_1	T_1	T_1
T_2	T_2	T_2
T_3		
T_4		
T_5		

Not been called

Callback List

- Reputation
- Post Time
- State3
- ...

Serial Num
[Tickets]

O

List: Hash chain!
Concretely: [T_1 T_2 ...] is H(H(T_1), T_2)

old	cur	new
T_1	T_1	T_1
T_2	T_2	T_2
T_3	T_3	✗
T_4		
T_5		

Not been called

Called!
apply method

Callback List

- Reputation
- Post Time
- State3
- ...

Serial Num
[Tickets]

O

List: Hash chain!
Concretely: $[T_1\ T_2 \ldots]$ is $\mathrm{H}(\mathrm{H}(T_1), T_2)$

old	cur	new
T_1	T_1	T_1
T_2	T_2	T_2
T_3	T_3	❌
T_4	T_4	T_4
T_5		

Not been called

Called!
apply method

Callback List

- Reputation
- Post Time
- State3
- ...

Serial Num
[Tickets]

O

List: Hash chain!
Concretely: $[T_1\ T_2 \ldots]$ is $\mathrm{H}(\mathrm{H}(T_1), T_2)$

old	cur	new
T_1	T_1	T_1
T_2	T_2	T_2
T_3	T_3	✗
T_4	T_4	T_4
T_5	T_5	✗

Not been called

Called!
apply method

zk-promises Base construction

- Reputation
- Post Time
- State3
- …

Serial Num
[old] [cur]
[new]

O

When making a forum post:

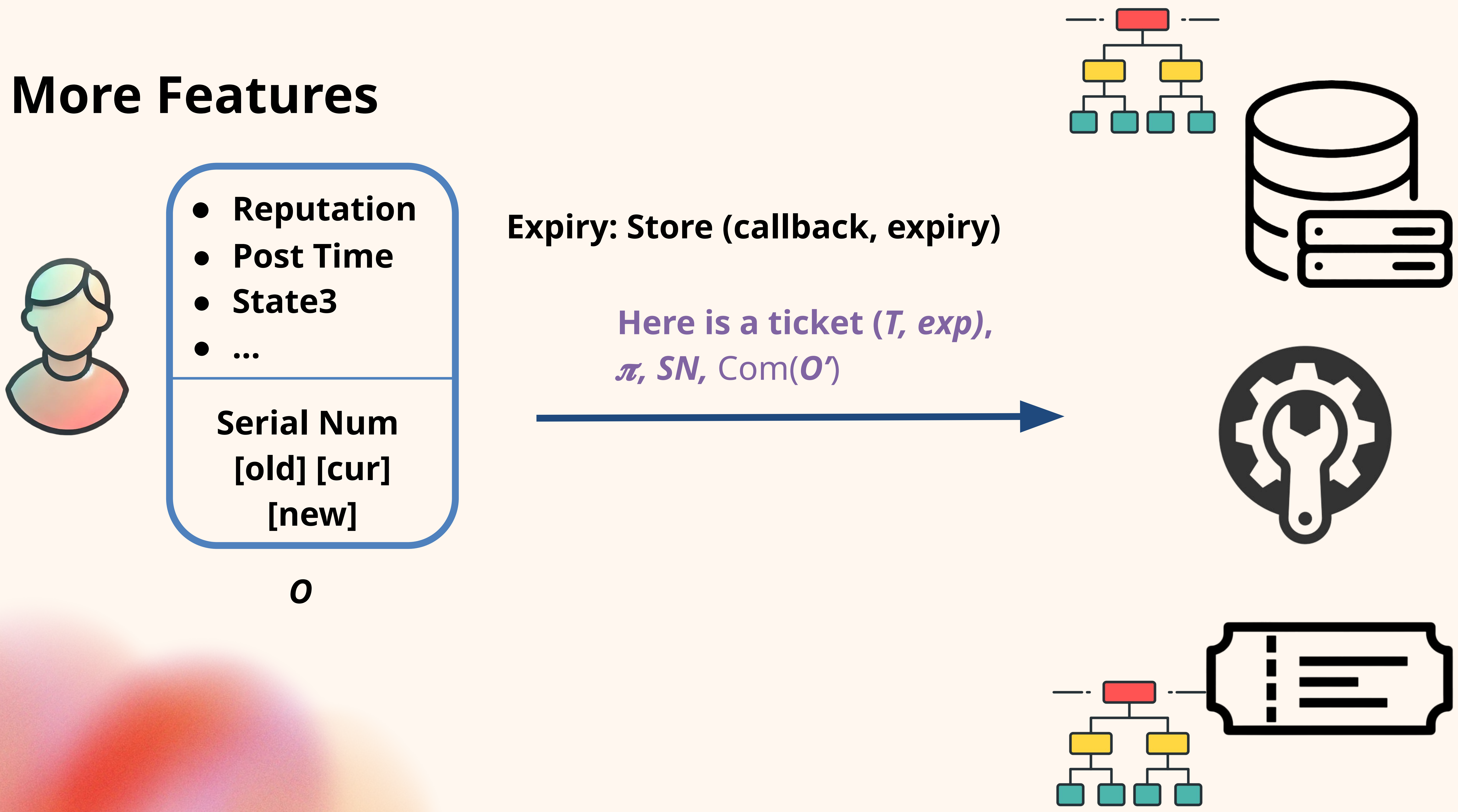
Here is a callback for
method(*O*), SN, π

- Reputation’
- Post Time’
- State3’
- …

New Rand SN
[old’] [cur’]
[new’]

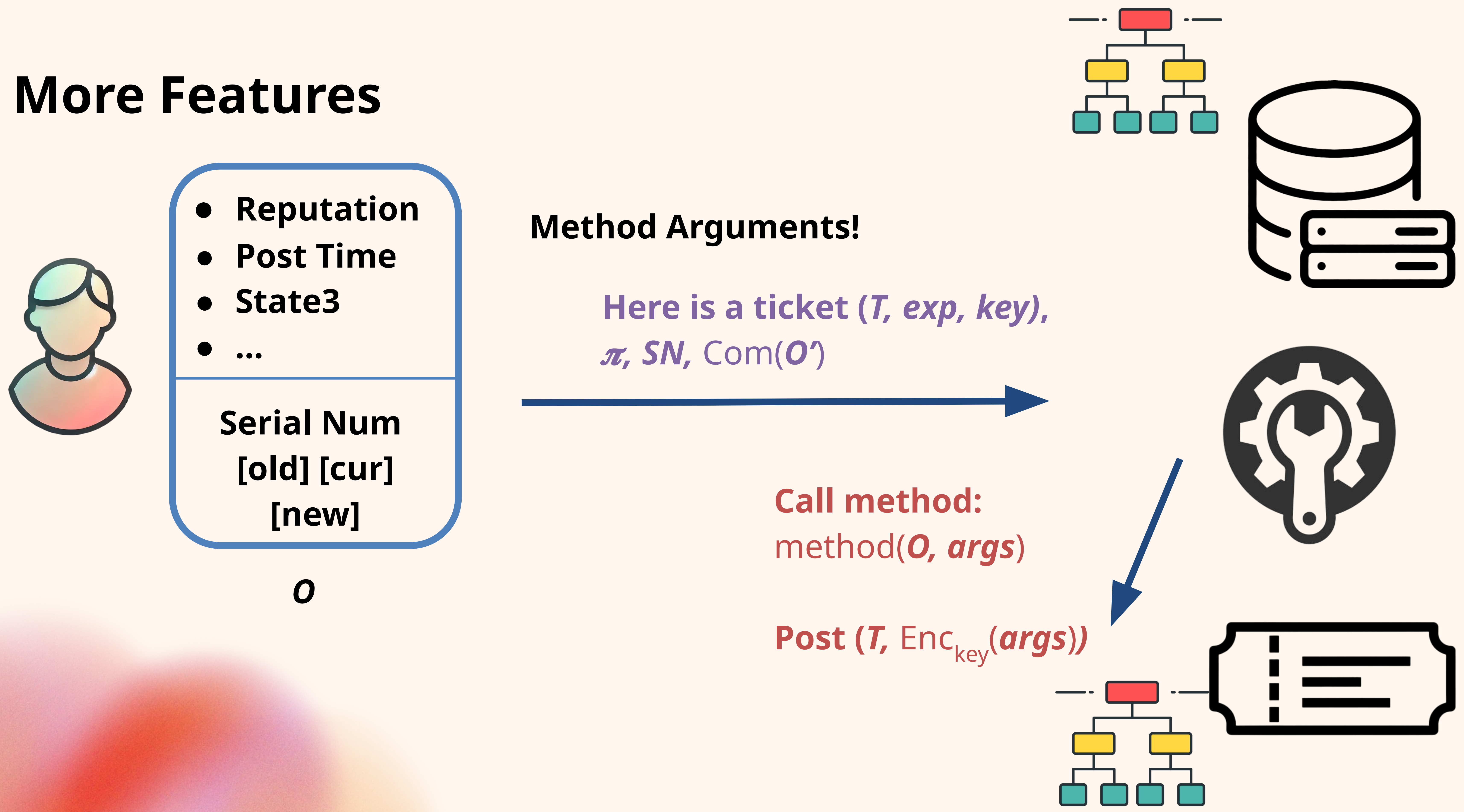
π: R_{update}
and R_{settle}
and R_{create}

More Features



Reputation
Post Time
State3
...
Serial Num [old] [cur] [new]
O
Expiry: Store (callback, expiry)
Here is a ticket (T, exp), π, SN, Com(O')

More Features



- Reputation
- Post Time
- State3
- …

Serial Num
[old] [cur]
[new]

O

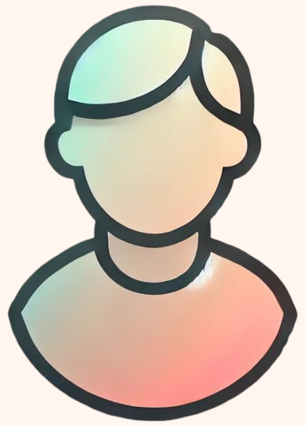
Method Arguments!

Here is a ticket (T, exp, key), π, SN, Com(O')

Call method:
method(O, $args$)

Post (T, Enc_{key}($args$))

More Features

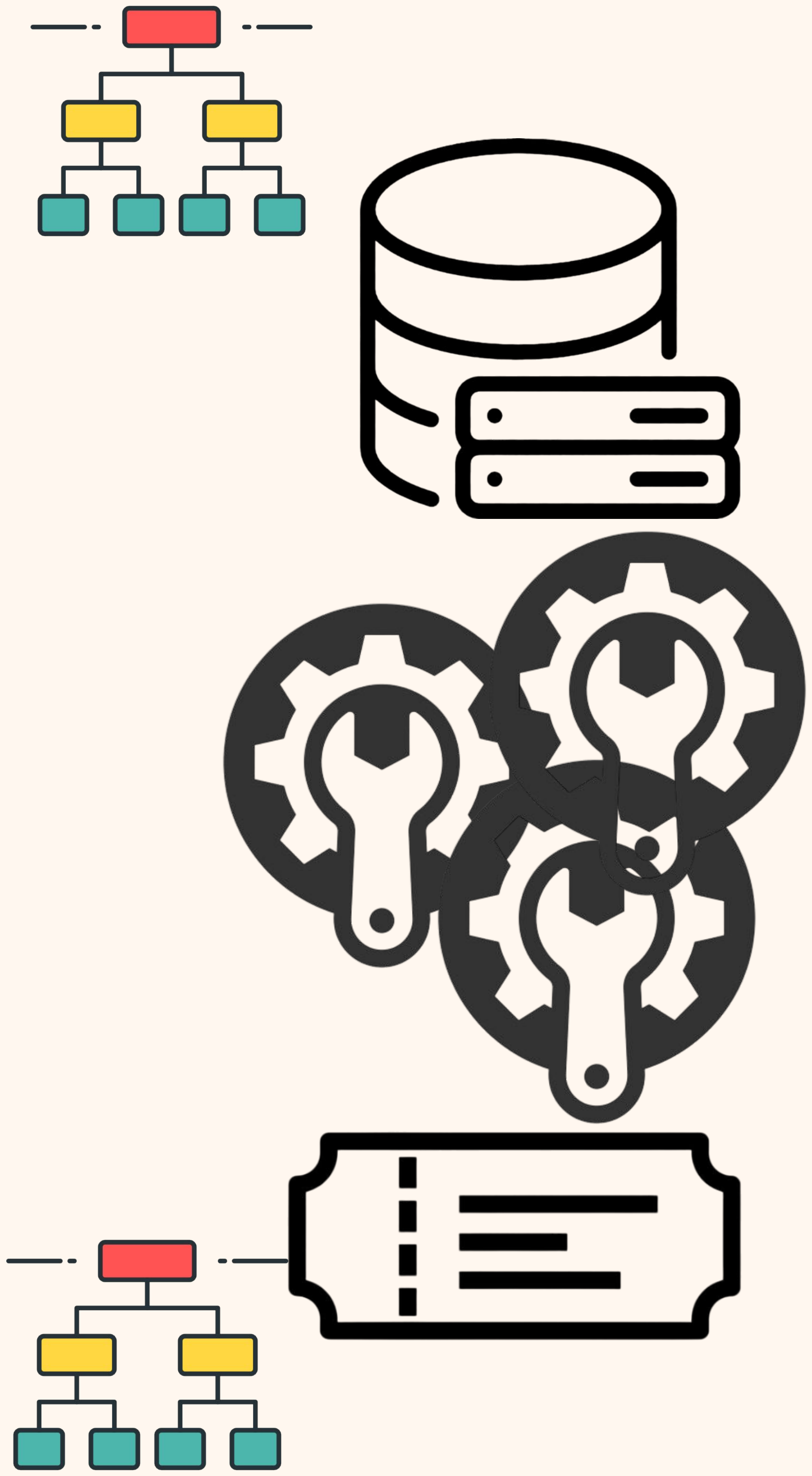


- Reputation
- Post Time
- State3
- ...

Serial Num
[old] [cur]
[new]

O

Separate Callback Bulletin
and Service Provider



More Features

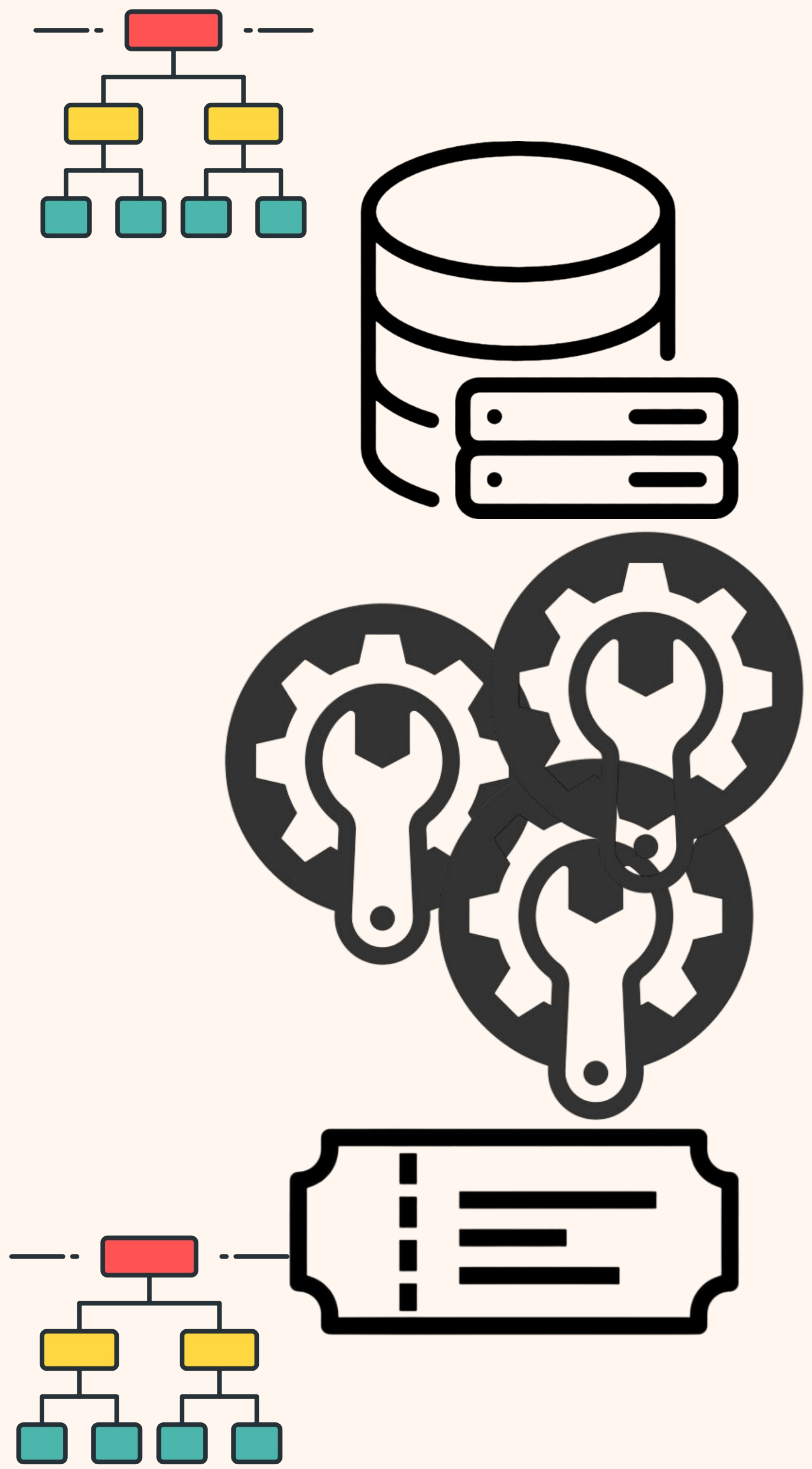
- Reputation
- Post Time
- State3
- ...

Serial Num
[old] [cur]
[new]

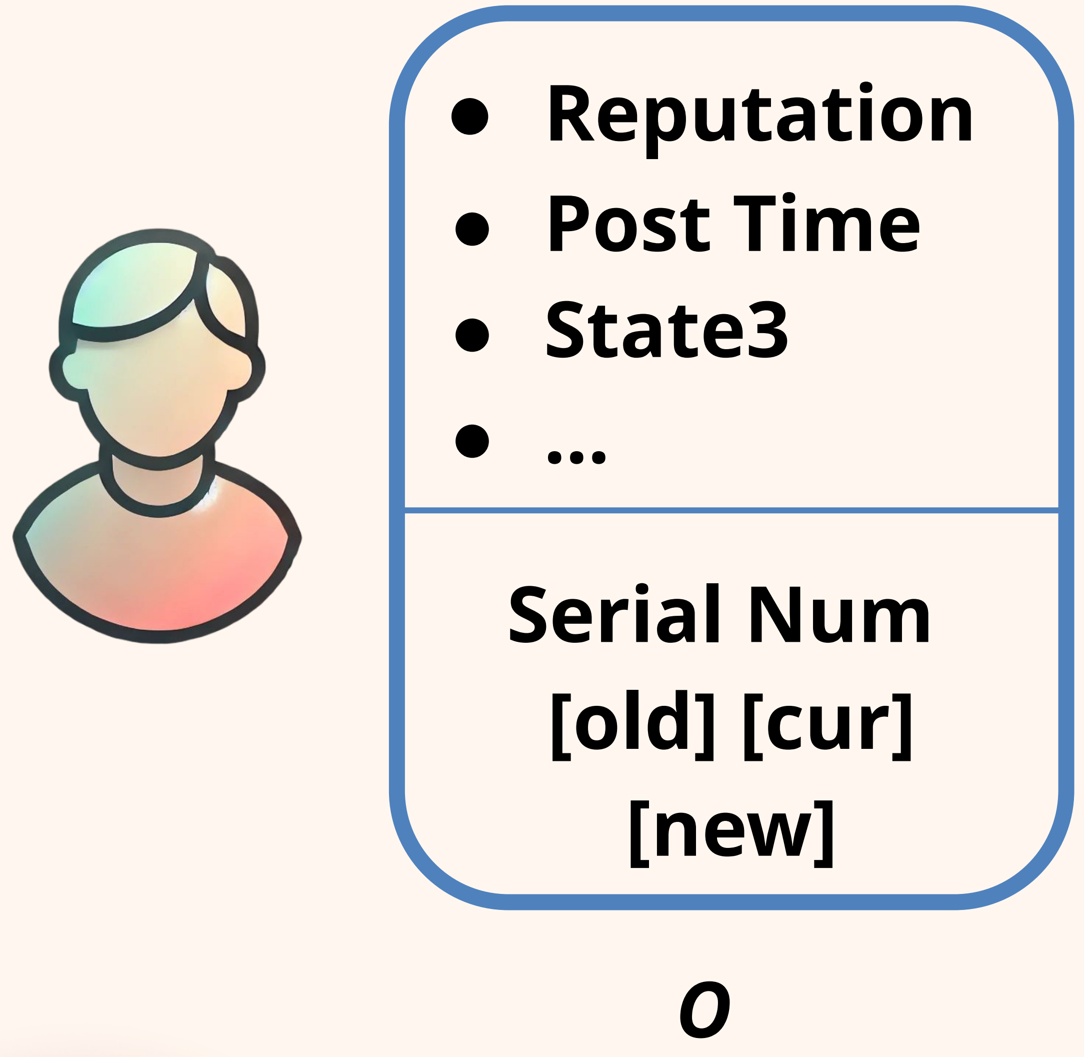
O

Separate Callback Bulletin and Service Provider

- **Create Post + Call unlinkability: Reveal** Com(***T***)



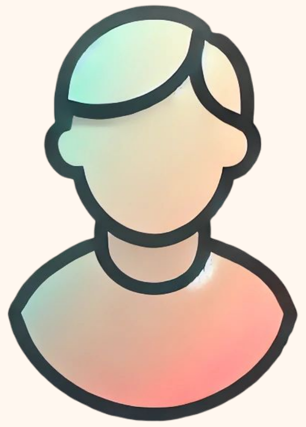
More Features



Separate Callback Bulletin and Service Provider

- **Create Post + Call unlinkability: Reveal** Com(***T***)
- **Ensure correct service provider: Sign arguments with ticket as public key**

Application Specific

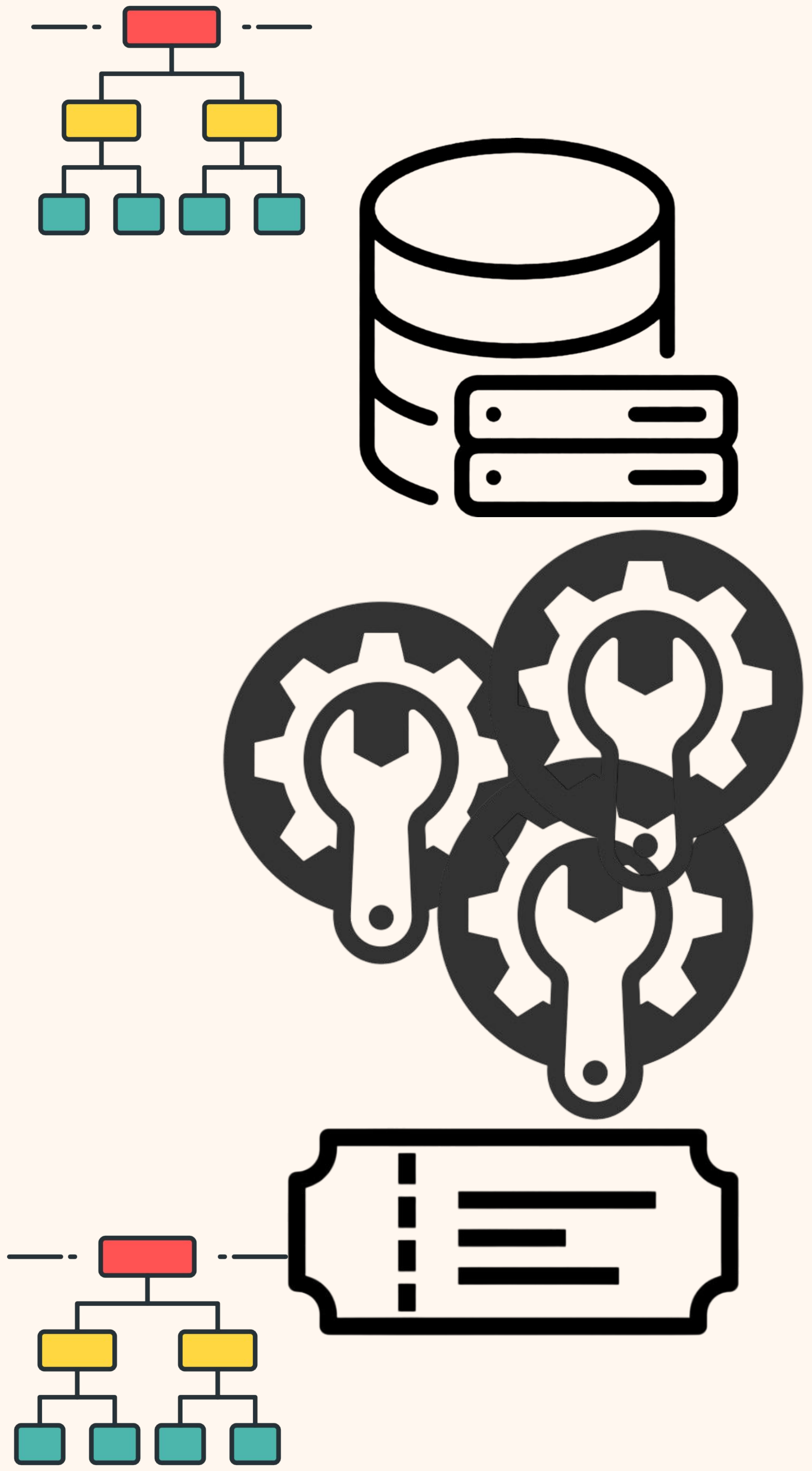


- Reputation
- Post Time
- State3
- ...

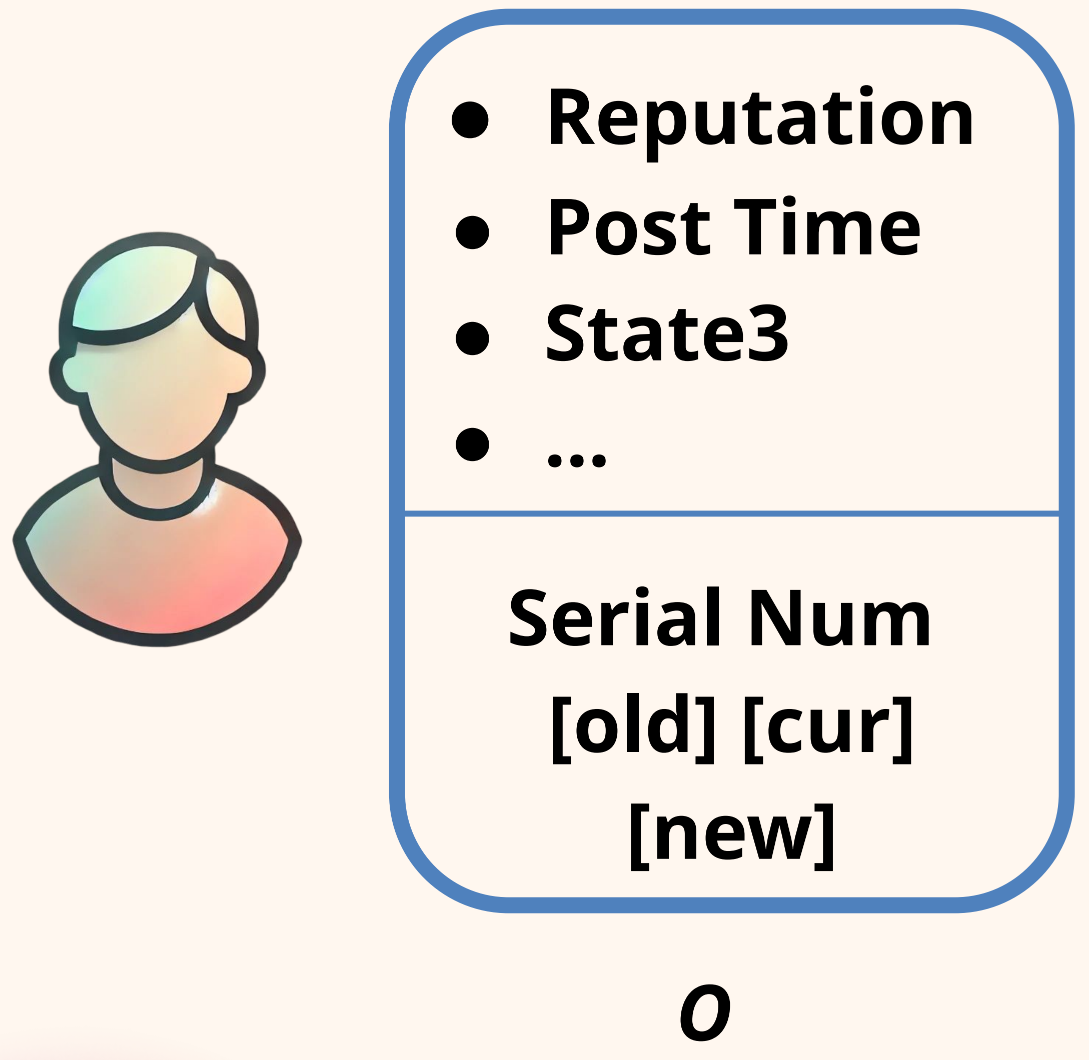
Serial Num
[old] [cur]
[new]

O

- Rate Limiting (leaky bucket)



Application Specific



- Rate Limiting (leaky bucket)
- Complex high dimensional reputation

Application Specific

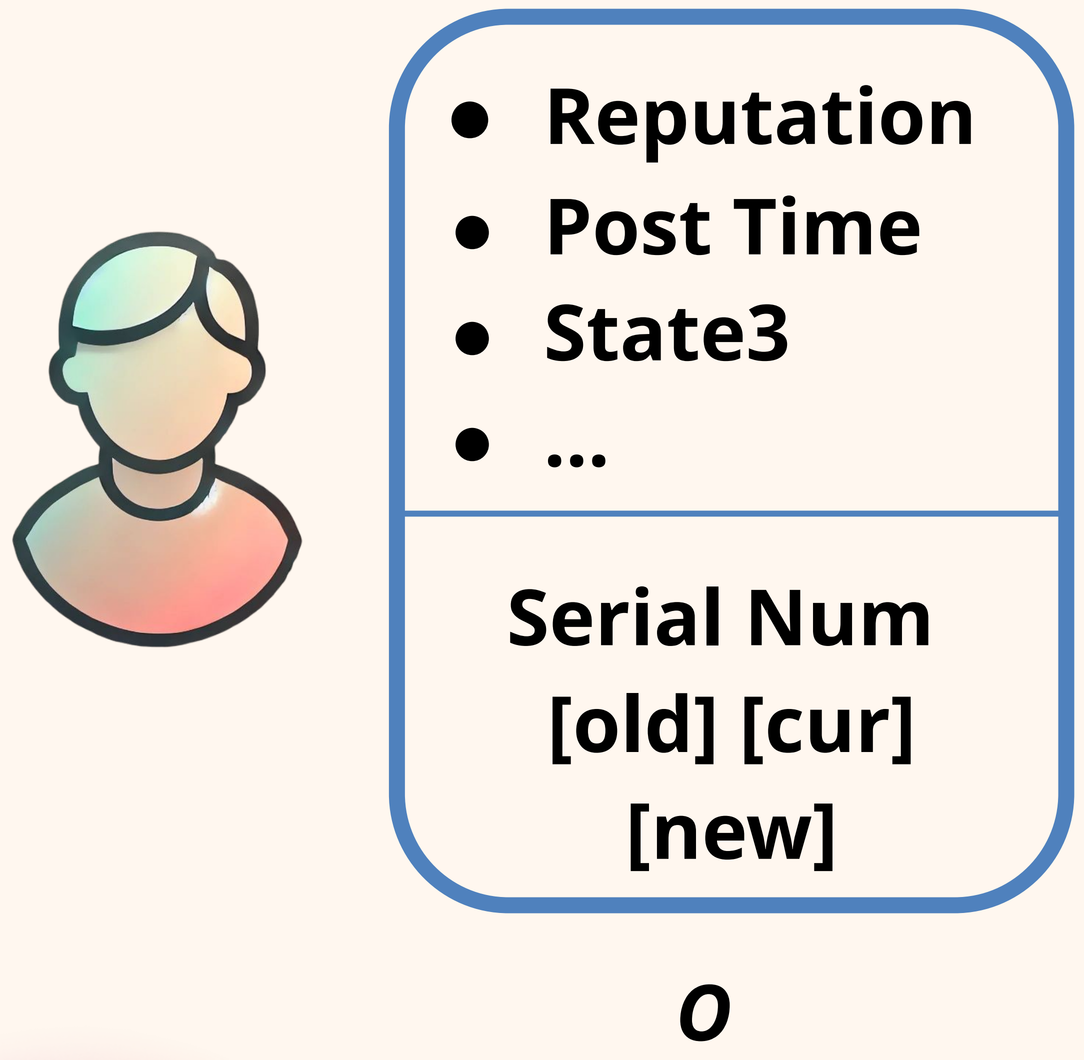
- Reputation
- Post Time
- State3
- …

Serial Num
[old] [cur]
[new]

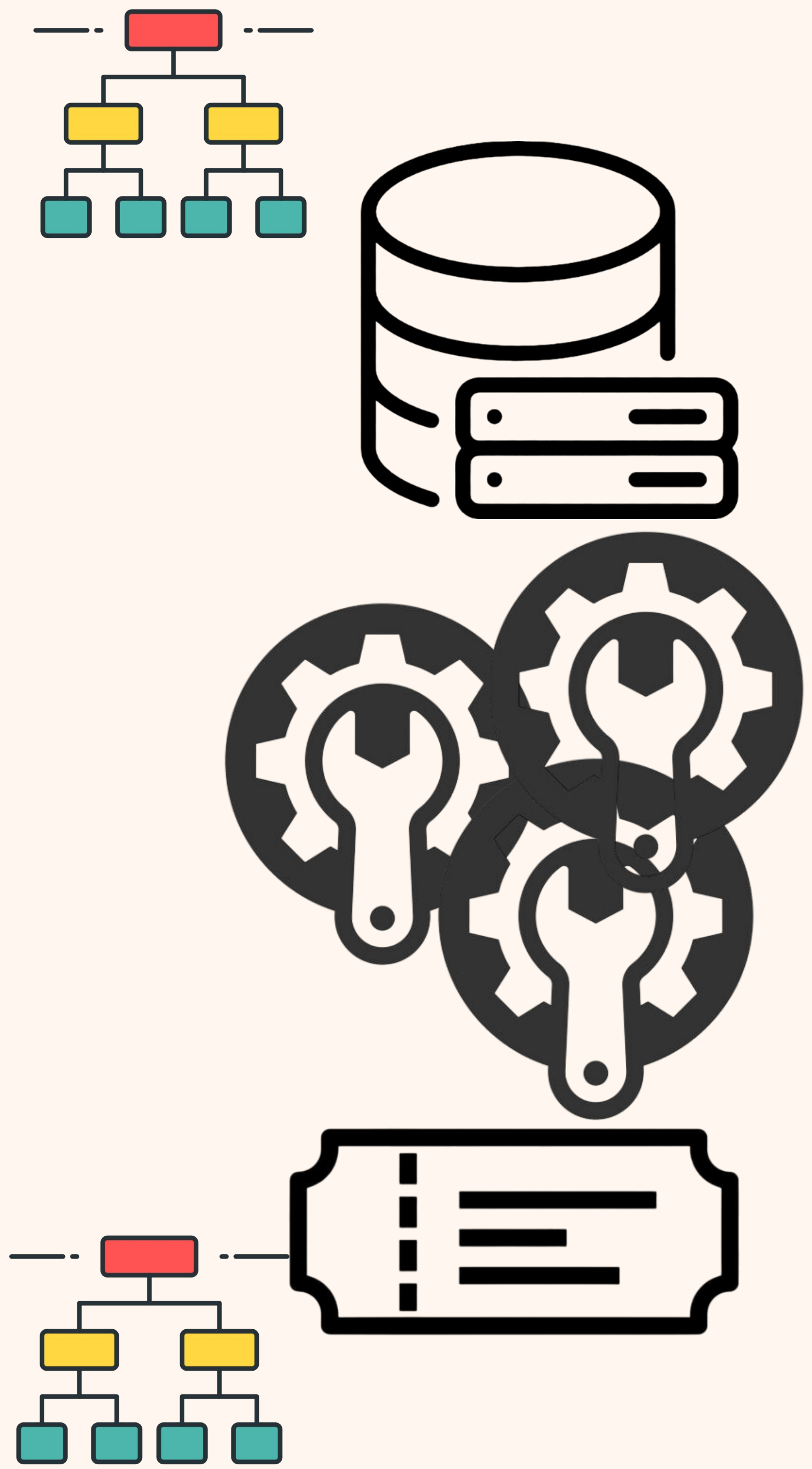
O

- Rate Limiting (leaky bucket)
- Complex high dimensional reputation
- Multiple service providers can access different parts of state

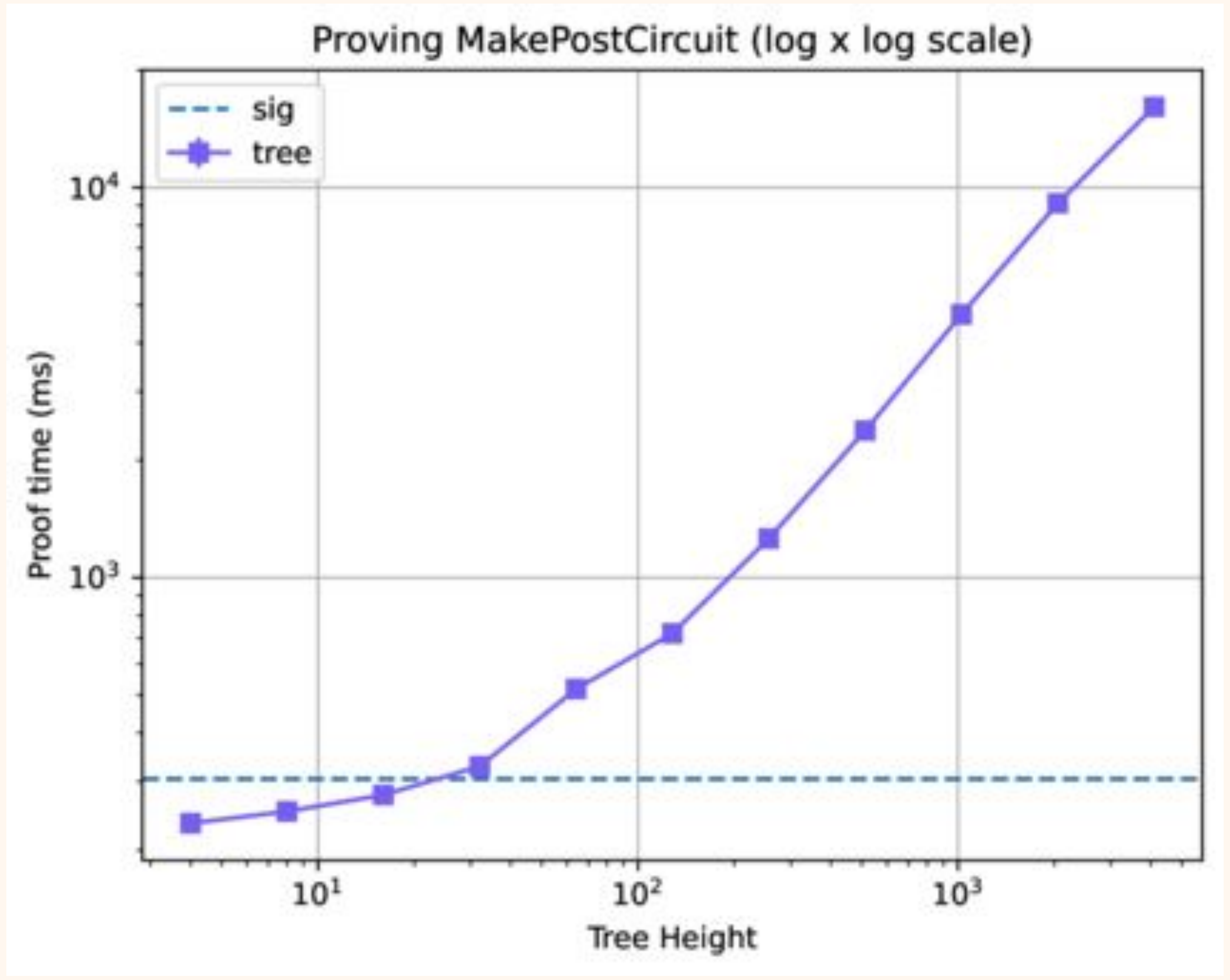
Application Specific

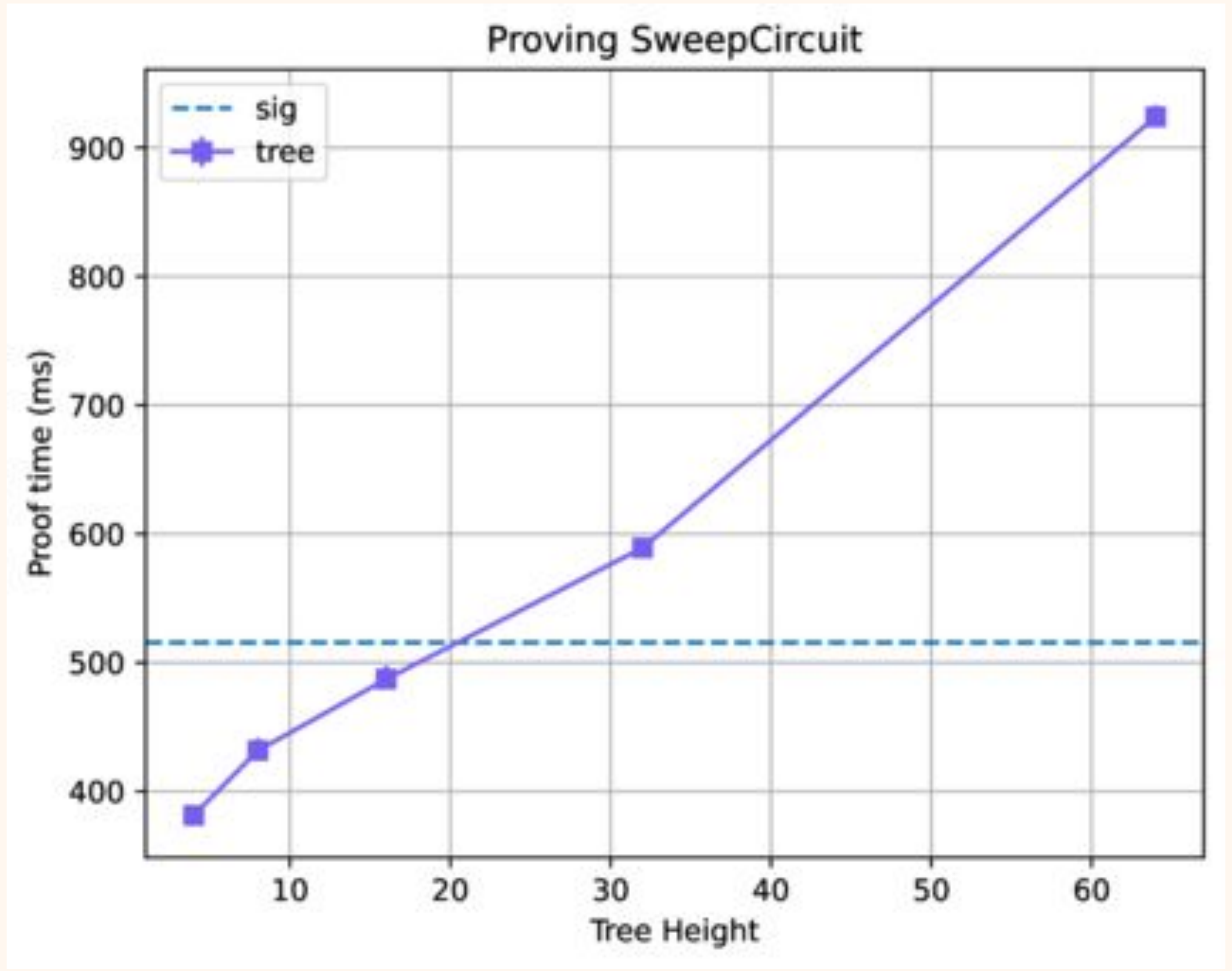


- Rate Limiting (leaky bucket)
- Complex high dimensional reputation
- Multiple service providers can access different parts of state
- Finite call retention with lockout (delete old calls)



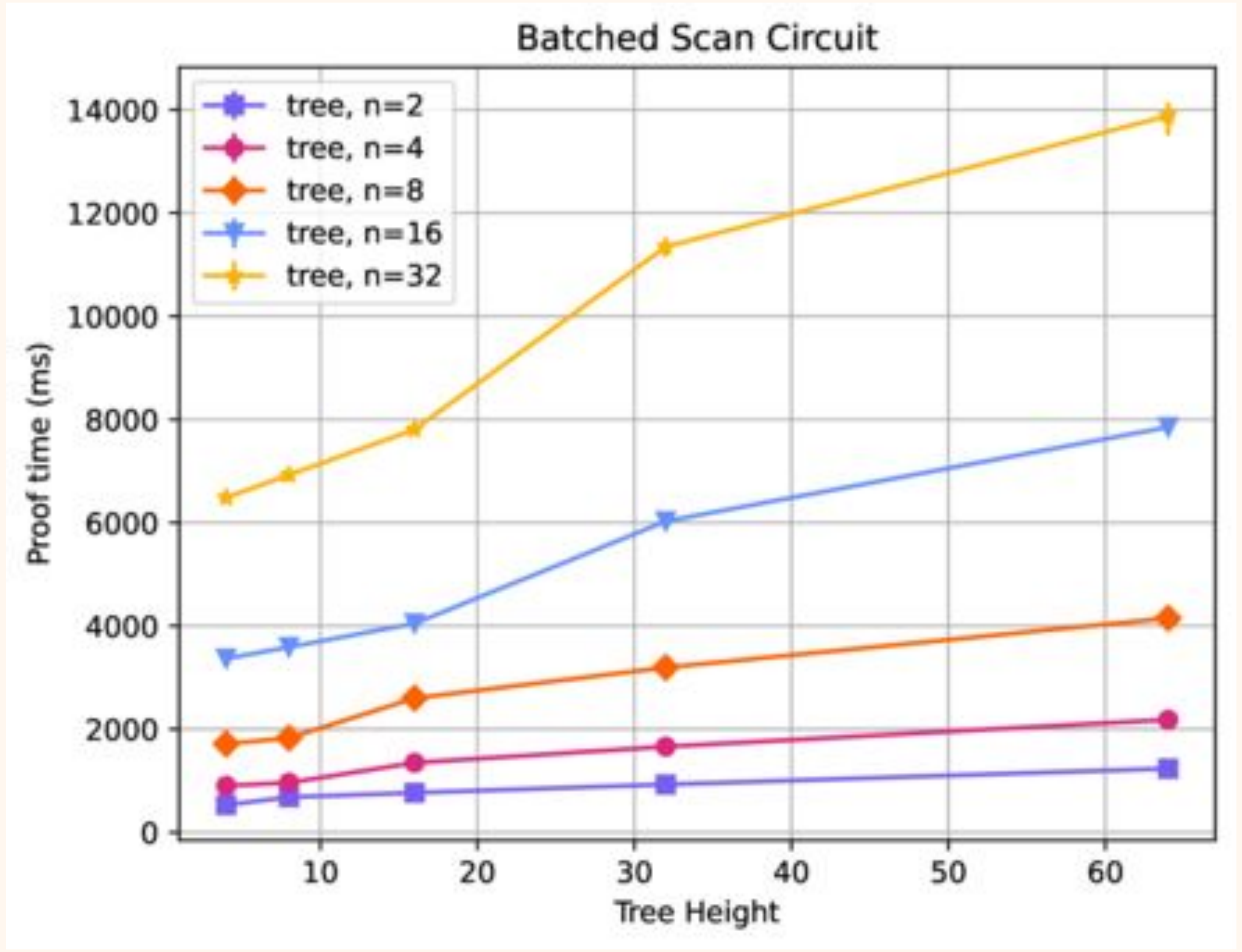
Performance: Microbenchmarks





Performance: Microbenchmarks

- **Making Posts (previous slide)**
  - 328 ms for depth 32 Merkle tree, scales linearly with height
  - 10x faster with signature, constant
- **Settling one callback (previous slide)**
  - 510 ms for depth 32 Merkle tree, scales linearly with height
  - 10x faster with signature, constant
- **Chunked settle**
  - Linear scaling



zk-promises

- Provides a generic framework extending zk-objects with callbacks
- Implement an anonymous reputation system through this framework

See the eprint for more details!
https://ia.cr/2024/1260

● Programmable ● Asynchronous ● Scalable