RChain Namespaces

Mike Stay
Kyle Butt

Validators

A **validator** is a participant in the casper protocol. Casper requires each validator to know what the entire set of validators is. Let V be the set of validators.

Regions

Assume a nonempty finite set R of **regions** and a function $v:R \to \wp^{+}(V)$ that picks out a nonempty subset of validators for each region. We say that a validator V inhabits a region X if $V \in v(X)$. Regions get to set slashing criteria on the validators that inhabit them.

Namespaces

The set S of **namespaces** is the powerset of the set of regions: $S = \wp(R)$. The validators for the namespace $S \vee T$ are $v(S) \cup v(T)$.

Namespaces

Deploying code in a namespace with many Vs will be more expensive, since more validators have to validate the messages. The execution may also be slower or otherwise constrained, since the criteria for inclusion of a validator in other regions may be less strict.

Names

All names are quoted processes:
If `P` is a process, `@{P}` is a name.

Names created with `new`

Names created with `new` have the form

`@{private(uid, namespace)}`

Quoted ground terms

Ground terms live in the namespace $\bot$:

NS(`@{"hello"}`) = $\bot$

There are no validators in $\bot$.

Quoted processes

Quoted processes live in the namespace that is the join of the namespaces of the names in the term.

Diffie-Hellman-like construction

If two processes exchange names `a`, `b` in the namespaces A, B, then they can construct a new name `@{*a | *b}` in the namespace $A \vee B$ that they both share. If `a`, `b` are created with new, no other processes can send or listen on that name.

Mobility

If `x` is in the namespace X, then both of these processes run on the validators of X:

```
x!(Q)
for (y <- x) { Q }
```

Joins

Joins between names that are not in the same namespace is a type error.

Bundling

You can't pattern match on the interior of a bundled process:

```
bundle{P | Q}
```

Bundling

Processes maintain two bits for whether you can send or receive on the corresponding name. Bundling lets you mask off bits:

```
bundle+{P} // write-only + eq
bundle-{P} // read-only + eq
bundle0{P} // equality-only
```

Registry

Ground terms of the form `rho:...` may be “registered” if certain proofs are supplied:
rho:iana:<domain> // DNS record
rho:pubkey:secp256:<pubkey> // signature
rho:hash:sha256:<hash> // preimage

Registry

Registering such a URN allows others to look it up and receive a write-only unforgeable name to send on (or a read-only name for a data feed). This name can be updated, e.g. when ownership of a domain changes.

Blessed contracts

The Rev contract, the registry, and the Casper contracts for bonding, unbonding, slashing, and distribution of funds will be “blessed” contracts, i.e. will run without needing funds.

Validator rotation

When a validator bonds, unbonds, or is slashed, every other validator needs to know about it. Doesn't necessarily happen at $\top$, but happens high enough that a coalition in a single namespace can't prevent it.