[Bug]: Destructuring null with rest #14982

According to the ECMAScript specification, running a JavaScript program `var { ... x } = null ;` is expected to result in `TypeError` but using Babel terminates without error.

Below is a detailed explanation using ECMAScript specification. Evaluation of `var { ... x } = null ;` is done by following algorithms.

1. Evaluation of *VariableStatement* : **var** *VariableDeclarationList* of the ECMAScript specification

The algorithms calls the evaluation of *VariableDeclarationList* in step 1, where *VariableDeclarationList* represents `{ ... x } = null ;`.

VariableStatement[0,0].Evaluation (this)

1. Let *next* be the result of evaluating *VariableDeclarationList*.
2. ReturnIfAbrupt(*next*).
3. Return empty.

2. Evaluation of *VariableDeclaration* : *BindingPattern Initializer* of the ECMAScript specification

The algorithm calls BindingInitialization of *BindingPattern* with arguments *rval* and **undefined**, where rval is `null`.

VariableDeclaration[1,0].Evaluation (this)

1. Let *rhs* be the result of evaluating *Initializer*.
2. Let *rval* be ? GetValue(*rhs*).
3. Return ? BindingInitialization of *BindingPattern* with arguments *rval* and **undefined**.

3. Operation BindingInitialization of the ECMAScript specification

The algorithm calls RequireObjectCoercible, where *value* is `null`.

BindingPattern[0,0].BindingInitialization (this, value, environment)

1. Perform ? RequireObjectCoercible(*value*).
2. Return ? BindingInitialization of *ObjectBindingPattern* with arguments *value* and *environment*.

4. Operation RequireObjectCoercible of the ECMAScript specification

RequireObjectCoercible throw `TypeError` exception, when argument type is Null.

Table 17: RequireObjectCoercible Results

Argument Type	Result
Undefined	Throw a **TypeError** exception.
Null	Throw a **TypeError** exception.

Therefore the program should throw `TypeError` exception.