Fearless Cooperation

giving `eval()` to your worst enemy for fun and profit

Brian Warner, Agoric

Decentralized Web Summit 2018

What is `eval()`?

```
eval('1+2') === 3;


const f = eval('(function(a) { return a+1 })');
f(5) === 6;
```

A function that turns strings into behavior

A Brief History of Web Browser(-like thing)s

- 1978: VT100 terminal, ANSI X3.64

image: Jason Scott, wikipedia

Code	Name
CSI *n* A	CUU – Cursor Up
CSI *n* B	CUD – Cursor Down
CSI *n* C	CUF – Cursor Forward
CSI *n* D	CUB – Cursor Back
CSI *n* E	CNL – Cursor Next Line

- 1990: HTML

```html
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```

- 1995: Javascript

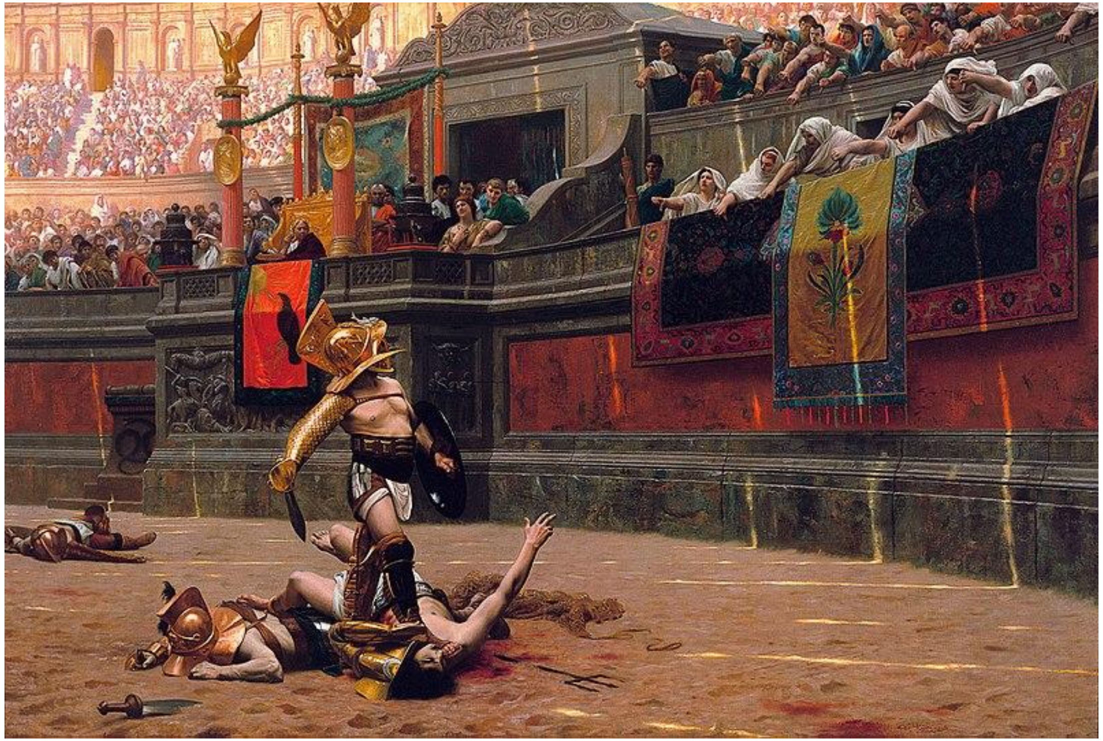
```html
<button id="hellobutton">Hello</button>
<script>
    document.getElementById('hellobutton').onclick = function() {
        alert('Hello world!');                     // Show a dialog
        var myTextNode = document.createTextNode('Some new words.');
        document.body.appendChild(myTextNode);     // Append "Some new words" to the page
    };
</script>
```

Interaction vs Vulnerability

Ulysses* and the *Sirens, 1891, by John William Waterhouse

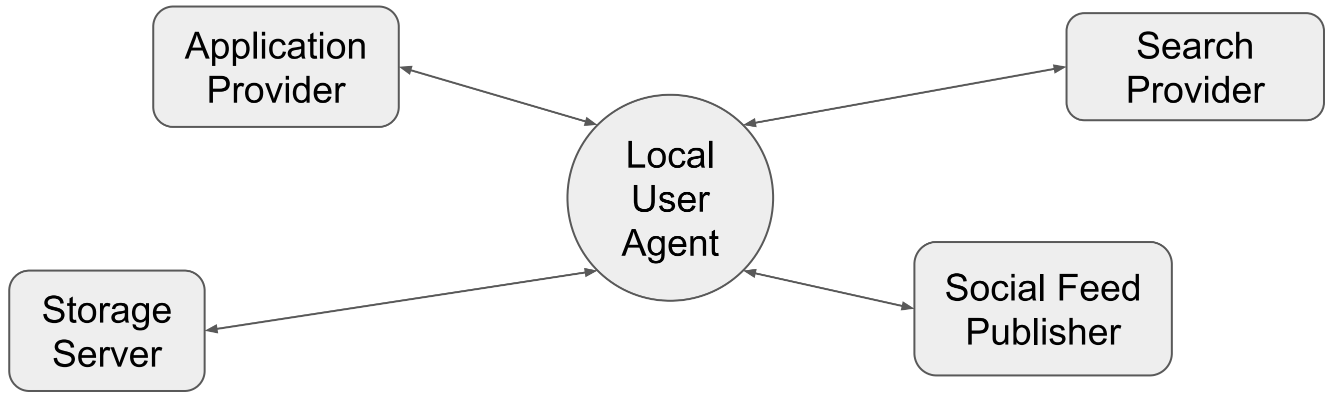
User Agent mediates interaction

- Browser is an arena in which programs do battle
- Server sends a program to the client
- Client runs the program with limited access to local resources
- Browser manages the interaction


Pollice Verso, 1872, by Jean-Léon Gérôme,

Three's a Party

- Client/server is only 2 parties
- Things become more interesting with 3 or more
- We need richer forms of interaction: not just us-vs-them
- Safe cooperation between mutually-suspicious programs



`eval()` turns strings into behavior

```
function server() {
    const items = database.loadItems();
    const queriesRemaining = [1, 2, 3, 4];

    const search = function(searchCriteria) {
        if (queriesRemaining.length <= 0)
            throw new NoMoreQueriesError();
        let matches = [];
        for (let row of items) {
            if (eval(searchCriteria)) {
                matches.push(row);
            }
        }
        queriesRemaining.pop();
        return matches;
    };

    return search;
}
```

Great:

```
search('row.price < 10.25 && row.size == 8.5');
```

Not so great:

```
search('database.deleteAllItems(); false');
```

Two-Argument Safe `eval()`

```
function server() {
    const items = database.loadItems();
    const queriesRemaining = [1, 2, 3, 4];

    const search = function(searchCriteria) {
        if (queriesRemaining.length <= 0)
            throw new NoMoreQueriesError();
        let matches = [];
        for (let row of items) {
            if (safeEval(searchCriteria, { row: row })) {
                matches.push(row);
            }
        }
        queriesRemaining.pop();
        return matches;
    };

    return search;
}
```

endowments

Blocked! :

```
search('database.deleteAllItems(); false');
 -> ReferenceError
```

Shared Mutable Primordials are Vulnerable

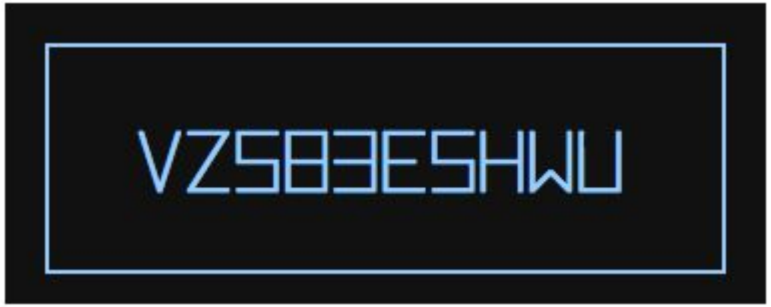
```
function server() {
    const items = database.loadItems();
    const queriesRemaining = [1, 2, 3, 4];

    const search = function(searchCriteria) {
        if (queriesRemaining.length <= 0)
            throw new NoMoreQueriesError();
        let matches = [];
        for (let row of items) {
            if (safeEval(searchCriteria, { row: row })) {
                matches.push(row);
            }
        }
        queriesRemaining.pop();
        return matches;
    };

    return search;
}
```

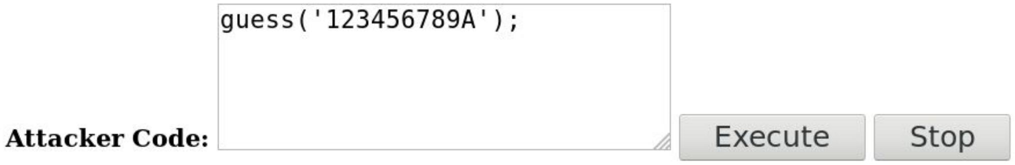
Modify what `Array` does

```
search('Array.prototype.pop = function() { }; false');
```

SES: Secure ECMAScript

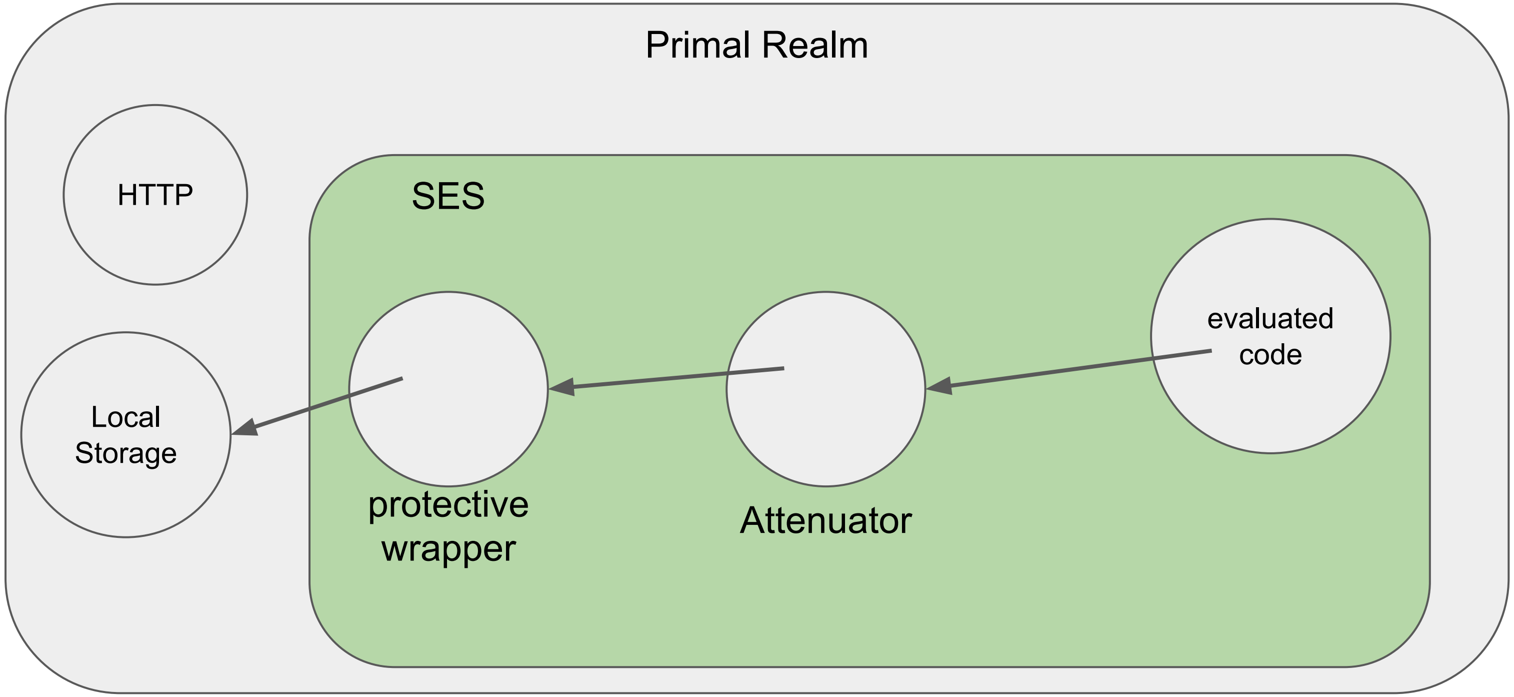
- https://github.com/Agoric/SES
- Works in web and Node.js
- Creates a "Realm" with frozen primordials and safe two-argument eval()
- Provides an Object-Capability -safe environment with Minimal overhead
- Still in development but go ahead and play with it today
- Online demo of confined execution




Attacker Code:

```
guess('123456789A');
```

Primal Realm



SES in action

create SES environment

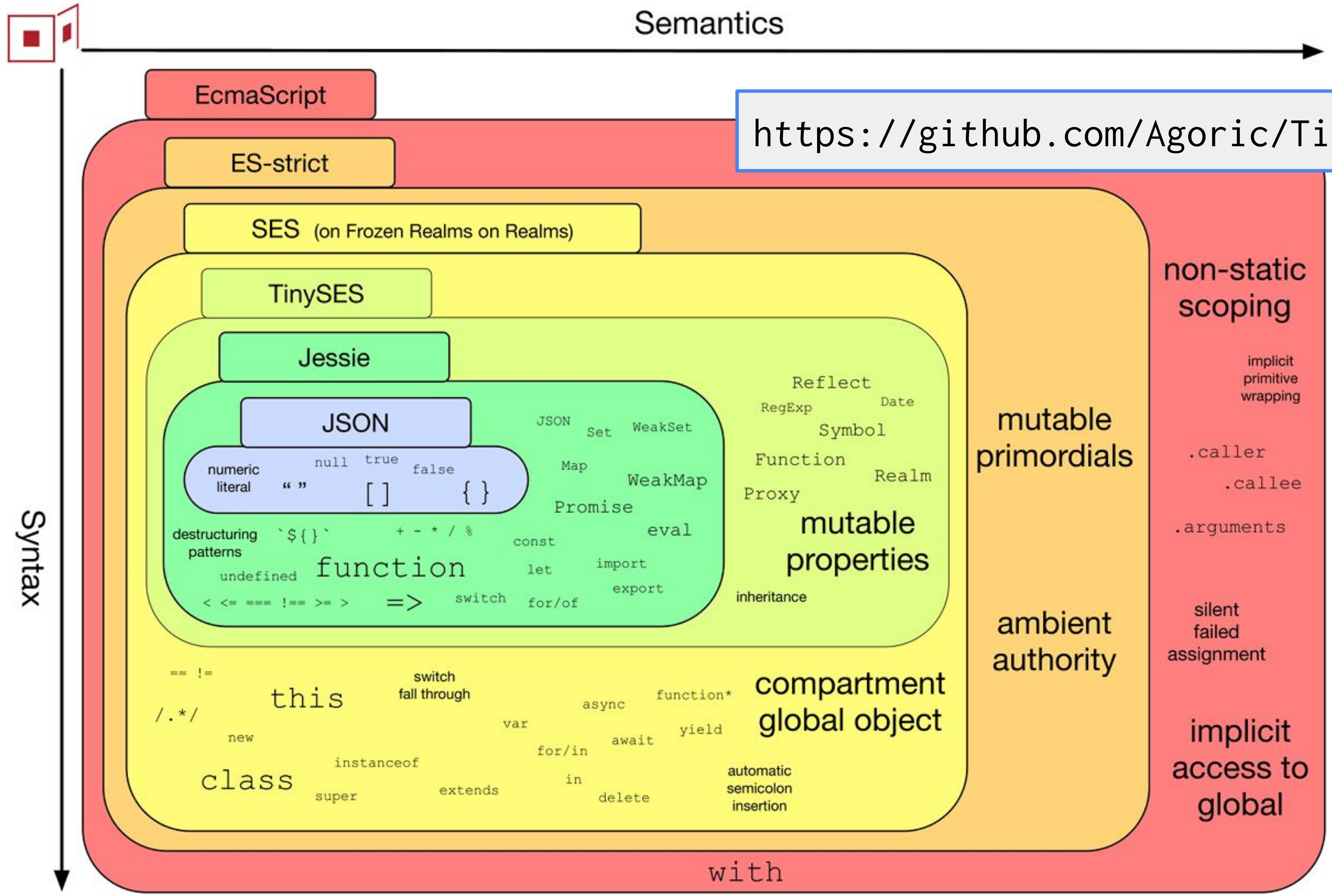
safe eval() from inside

freeze return values

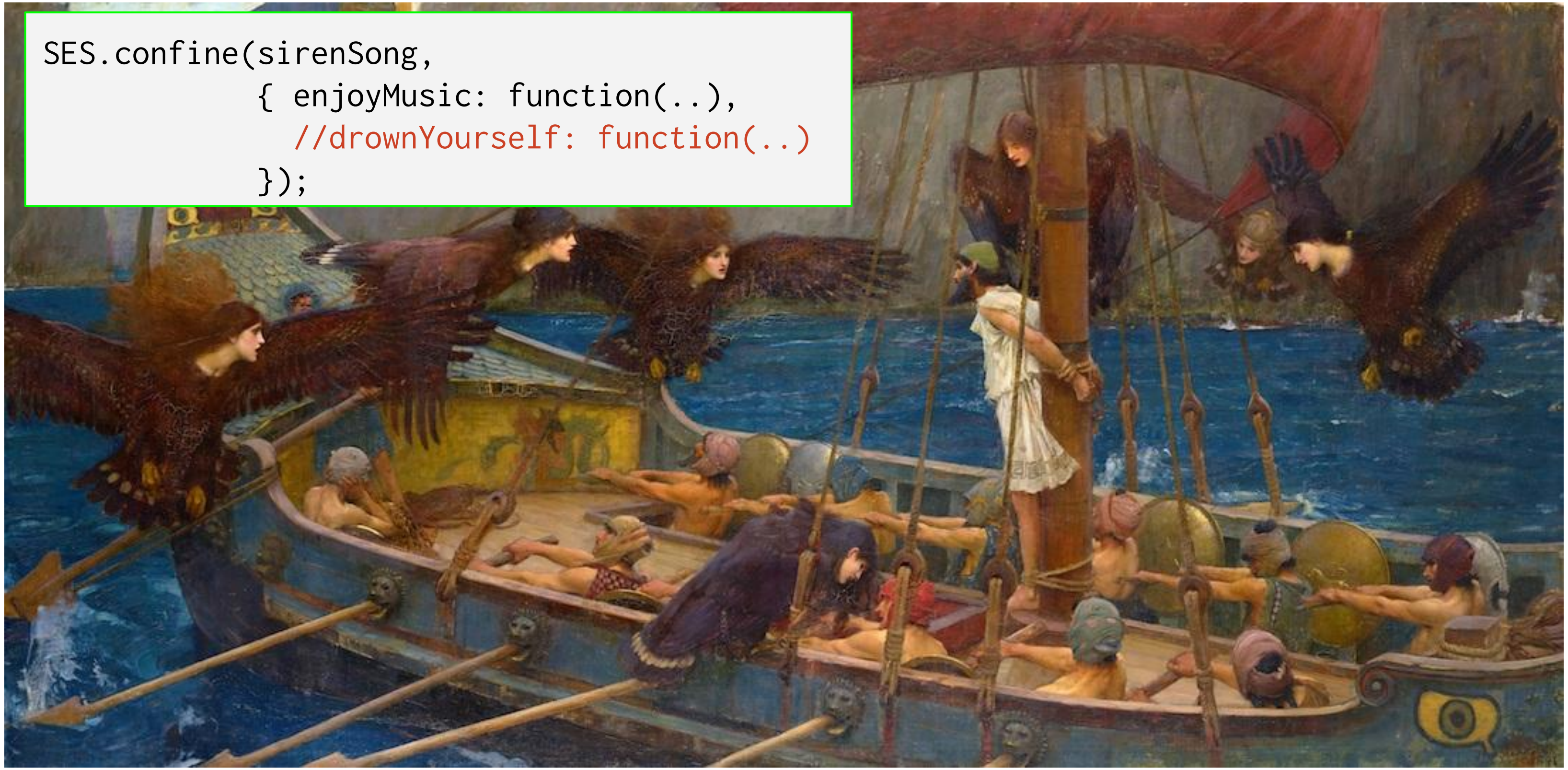
safe eval() from outside

```
const r = SES.makeSESRootRealm();
function server() {
    const items = database.loadItems();
    const queriesRemaining = [1, 2, 3, 4];

    const search = function(searchCriteria) {
        if (queriesRemaining.length <= 0)
            throw new NoMoreQueriesError();
        let matches = [];
        for (let row of items) {
            if (SES.confine(searchCriteria, { row })) {
                matches.push(row);
            }
        }
        queriesRemaining.pop();
        return def(matches);
    };

    return def(search);
}
const s = r.evaluate(`${server}; server()`, { database });
```

https://github.com/Agoric/TinySES



Conclusions



```
SES.confine(sirenSong,
            { enjoyMusic: function(..),
              //drownYourself: function(..)
            });
```

https://github.com/Agoric/SES