# JSONP

- “JSON with Padding”
- Pre-CORS cross-origin API exploiting script-src SOP exception
- Quite popular and fast
- Lots of legacy APIs of this sort that won’t go away or be converted to CORS for a long time
  - Pre-CORS browser compatability

- Client sends the name of a callback function
- Server returns JSON wrapped in a call to that function

# JSONP example

Client includes:

```
<script type=“text/javascript”
src=“http://api.example.com/api?callback=myFunc”>
</script>
```

Response from api.example.com:

```
myFunc({“Name”: “Foo”,  “Id” : “1234”, “Rank” : “Bar”});
```

# JSONP problems

- Data sharing by code injection

- Should be as safe as a CORS call

- Is actually as dangerous as a script injection

## Request: allow JSONP in CSP

- Potential CSP adopters want to be able to call JSONP APIs
  - But without having to accept the unlimited risk of putting those sites in script-src
  - And would like to make such calls close to "as safe as CORS" as possible

# Proposed CSP directives

“**jsonp-src**” : list of origins allowed for script-src but returned resources must parse as one of:

*function({JSON});*
*obj.function({JSON});*
*obj[“function”]({JSON});*

“**jsonp-sink**” : optional list of function names. If specified with jsonp-src, when script includes from origins allowed by jsonp-src are parsed, *function* must be in this list of function names.

# Why jsonp-sink?

- Prevent ROP-style exploitation:

http://api.example.com/jsonp?func=stage1

http://api.example.com/jsonp?func=stage2

http://api.example.com/jsonp?func=stage3