ATMOSPHERE
CONFERENCE

scaling & securing node.js apps

Maciej Lasyk

AtmosphereConf 2014

Warsaw, 2014-05-19

$ whoami

- not only sysadmin ;)

- 14+ years of exp software dev / sysop

- ops lead

- contributing to Fedora Project (and couple more)

- and...

$ whoami

- not only sysadmin ;)

- 14+ years of exp software dev / sysop

- ops lead

- contributing to Fedora Project (and couple more)

- and...

- love AtmosphereConf – been to Velocity


fedora

So what do you think about JS?

- JS is for children!

- JS is slow!

- JS is not scalable!

- JS is insecure!

node.js: history

- 2008: Google V8 release

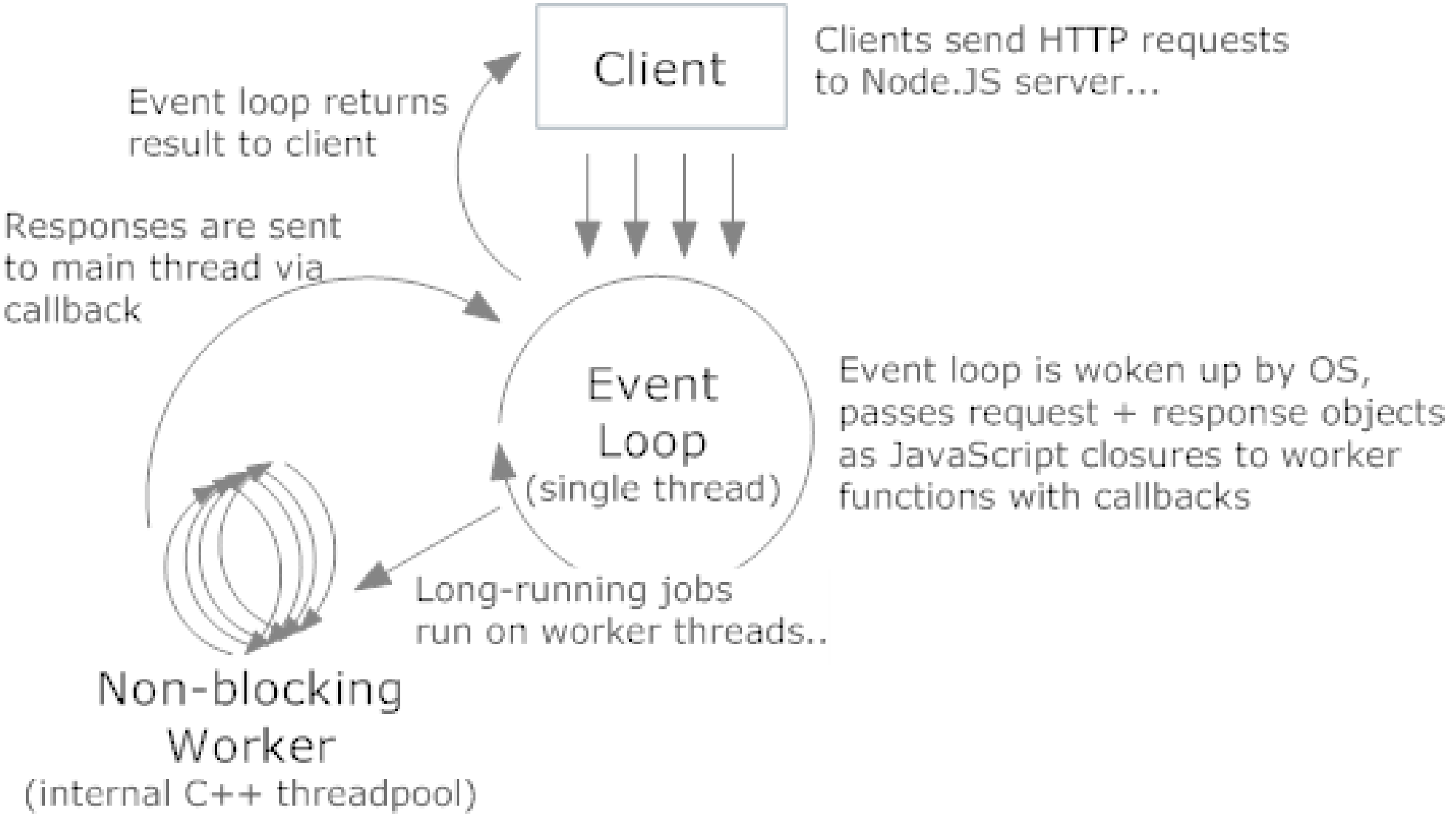
- 2009: Ryan Dahl & node.js

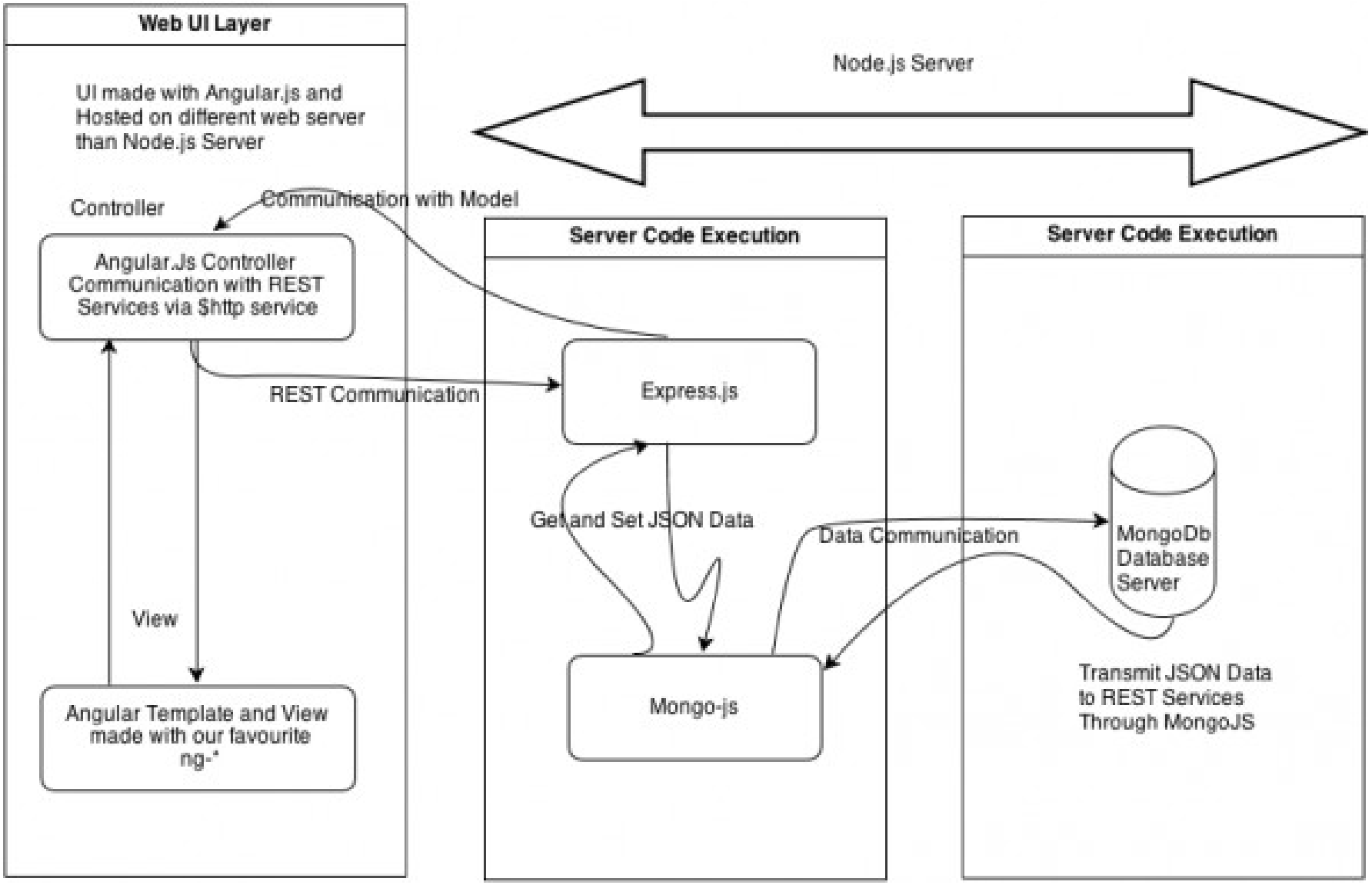
- 2011: node.js release

- later on – Joyent till today

- and ^liftsecurity / nodesecurity.io

Node.JS Processing Model





(http://www.phloxblog.in)

node.js: developing ur code

raw node.js coding srsly?

node.js: developing ur code

maybe some frameworks?

- webserver: express
- client-server sync: backbone.js
- push: socket.io
- templates: swig
- i18n: babelfish
- client – side: jquery
- or...
- kraken.js does the all (almost)

node.js: developing ur code

Biggest win here?

node.js: developing ur code

Biggest win here?

One Language to Rule them all!

security: JS issues

eval() like fncs takes string argument and

evalute those as source code

security: JS issues

eval() like fncs takes string argument and

evalute those as source code

srsly – who does that?

security: JS issues

not only evals:

```
setInterval(code,2)

setTimeout(code,2)

str = new Function(code)
```

Content-Security-Policy knows about those but we're talking about server side...

security: JS issues

Global nameSpace Pollution

- node.js is single threaded

- all variable values are common

- one could thrtically change bhv of others reqs

- watch out for globals then!

security: JS issues

```
var auth = false;

app.get('/auth', function(req, res) {
    if(legit) { auth = true; res.send("success");
});

app.get('/payments-db', function(req, res) {

   if (auth) res.send("legit to see all payments data");

   else res.send("not logged in");
})

app.listen(8080);
```

security: JS issues

So now imagine..

global namespace pollution + evals & co

security: JS issues

So now imagine..

global namespace pollution + evals & co

security: JS issues

object properties:

- writable: RO/RW

- enumerable: no loops enumeration

- configurable: deletion prohibited

- all default set to True so watch out

security: JS issues

```
var obj = {}; obj.prop = "LOL";

// OR:

Object.defineProperty(obj, "prop", {
    writable: true,
    enumerable: true,
    configurable: true,
    value: "LOL"
})
```

security: JS issues - prevention

strict mode:

- let's throw all errors!

- declare variables!

- global namespaces help

security: JS issues - prevention

```
"use strict";

function do_smt() {
   do_smt.caller; // no way :)
   do_smt.arguments; // no way :)
}
```

security: JS issues - prevention

```
"use strict";
eval("var smt = 123");
console.log(smt); // sorry – ReferenceError
```

security: JS issues - prevention

```
"use strict";
eval("var smt = 123");
console.log(smt); // sorry – ReferenceError
```

But watch out:

```
"use strict";
var smt = 0;
eval("smt = 123");
console.log(smt); // outputs “123” properly
```

security: JS issues - prevention

strict mode:

- evals & co are not that insecure now

- no access to caller and args props

- enable globally or for some scope

- what about strict mode in 3rd party mods?

security: JS issues - prevention

Static code analysis

- If not doing it already – just do

- Commit hooks in (D)VCSes

- JSHint / JSLint

- Create policy for static code analysis

- Update & check this policy regularly

node.js – exploits anyone?

- http://seclists.org/bugtraq – ? hits

- http://osvdb.org – ? hits

- http://1337day.com, http://www.exploitdb.com – ? hit

- http://nodesecurity.io/advisories – ? hits

node.js – exploits anyone?

- http://seclists.org/bugtraq – 0 hits

- http://osvdb.org – 2 hits

- http://1337day.com, http://www.exploitdb.com – 1 hit

- http://nodesecurity.io/advisories – 4 hits

node.js – exploits anyone?

- http://seclists.org/bugtraq – 0 hits

- http://osvdb.org – 2 hits

- http://1337day.com, http://www.exploitdb.com – 1 hit

- http://nodesecurity.io/advisories – 4 hits

Such security big?

node.js – exploits anyone?

- http://seclists.org/bugtraq – 0 hits

- http://osvdb.org – 2 hits

- http://1337day.com, http://www.exploitdb.com – 1 hit

- http://nodesecurity.io/advisories – 4 hits

Such security big?

not exactly

node.js – what's wrong than?

node.js security is a blank page

Sessions	NO
Permanent Data Storage	NO
Caching	NO
Database Access	NO
Logging	NO
Default Error Handling	NO
...	Most likely NO

http://www.slideshare.net/ASF-WS/asfws-2012-nodejs-security-old-vulnerabilities-in-new-dresses-par-sven-vetsch

node.js – exceptions / callbacks

callbacks Error object – remember to handle those

```
var fs = require("fs");

fs.readFile("/some/file", "utf8", function (err, contents) {
    // err will be null if no error occured

    // ... otherwise there will be info about error
});
```

forget about handling and die debugging

node.js – eventemitter

EventEmitter: emitting events 4 async actions

```
var http = require("http");

http.get("http://nodejs.org/", function (res) {
    res.on("data", function (chunk) {
        do_something_with_chunk;
    });
    res.on("error", function (err) {
        // listener handling error
    });
});
```

Attach listeners to errors events or welcome unhandled exception!

node.js – uncaught exceptions

- by default node.js will print stack trace and terminate thread

- EventEmitter / process / uncaughtException

```
// it looks like this by default:

process.on("uncaughtException", function (err) {
    console.error(err);
    console.trace();
    process.exit();
});
```

node.js – uncaught exceptions

- by default node.js will print stack trace and terminate thread
- EventEmitter / process / uncaughtException

```
// it looks like this by default:

process.on("uncaughtException", function (err) {
    console.error(err);
    console.trace();
    process.exit();
});
```

So do you <u>really</u> want to comment out the 'process.exit()' line?

node.js – domains

- error handling mechanism

- group I/O operations

- when err event -> domain is notified not process

- context clarity

node.js – domains

Using Express take look at that:

https://github.com/brianc/node-domain-middleware

Assigning each Express request to a separate domain?

node.js – npm modules

- npm install (-g)

- who creates modules?

- who verifies those?

- how to update?

  - semantic versioning in package.json

  - "connect":"~1.8.7" -> 1.8.7 - 1.9

node.js – npm modules

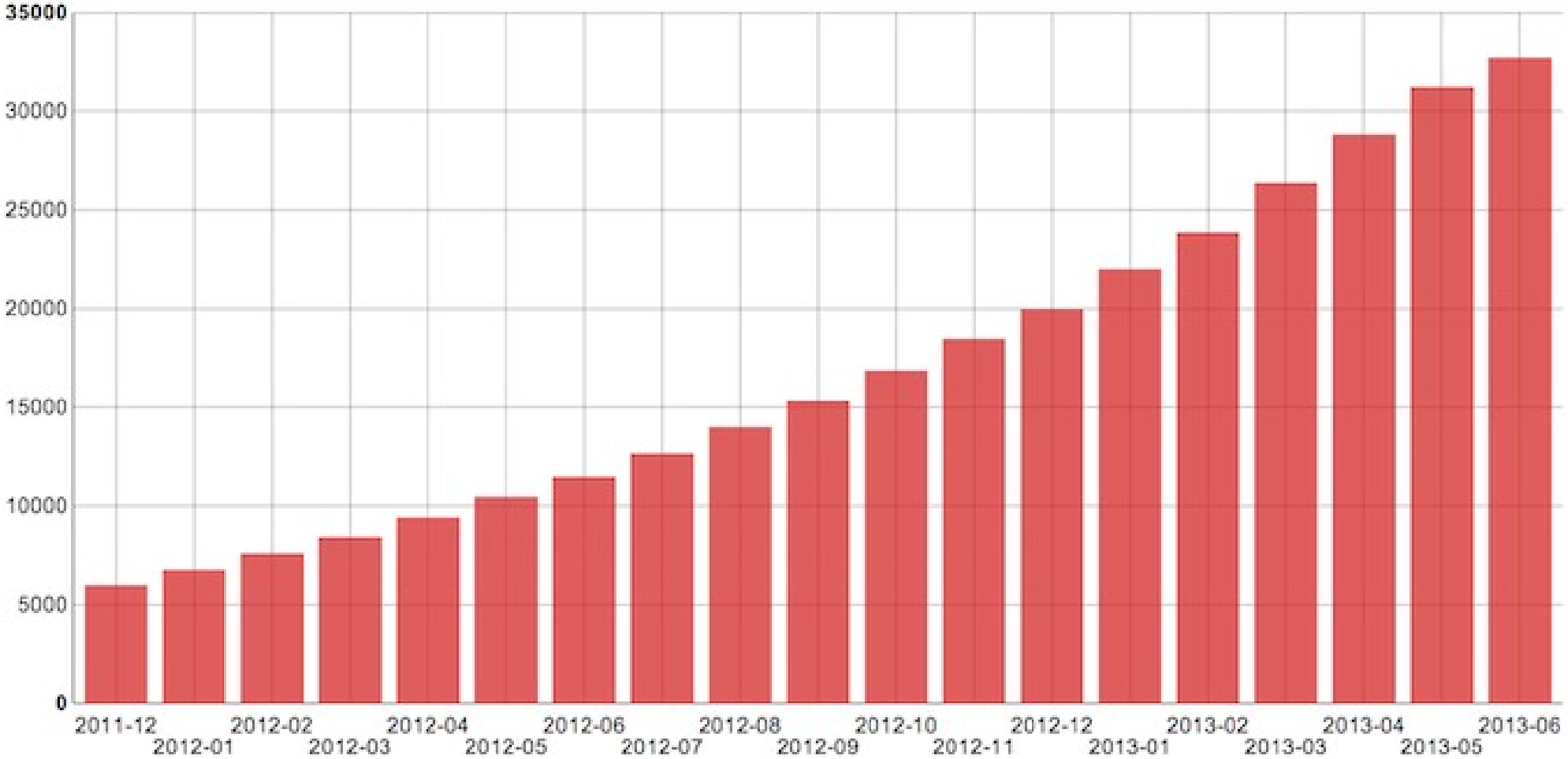
--ignore-scripts

stop preinstall/prepublish scripts

- mods auditing: https://nodesecurity.io/

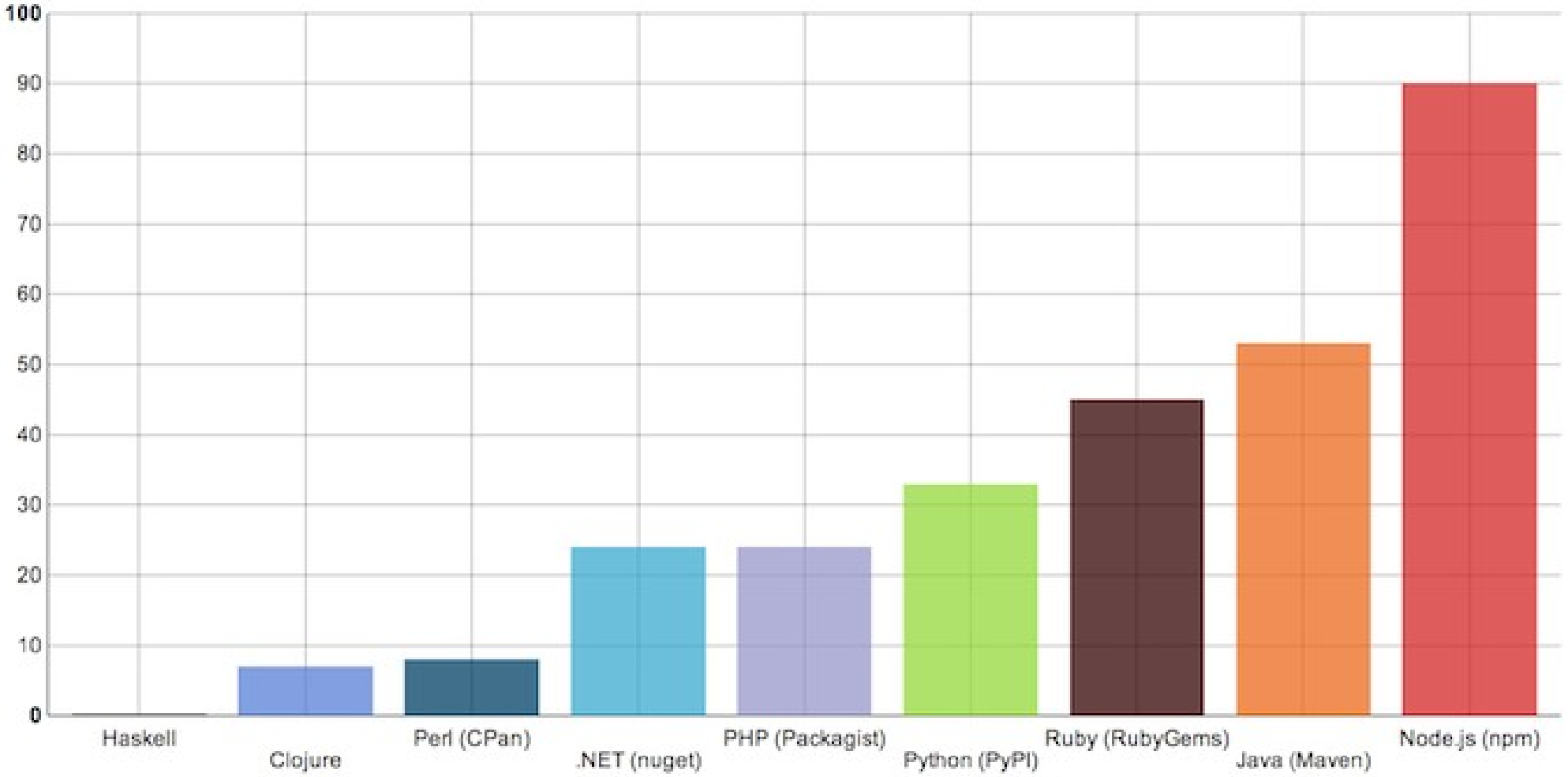
node.js – npm modules

The scale of npm modules



node.js – npm modules

Comparison to other langs (mods/day):



node.js – npm modules

Remember:

- use strict?

- static analysis?

- does include some test suite?

- what is the dependency tree?

node.js – express

Express – web dev framework

Built on top of connect

node.js – express – basic auth

```
var express = require('express'),
app = express();
app.use(express.basicAuth("user", "pwd"));
app.get("/", function (req, res) {
    res.send('Hello World');
});
app.listen(8080);
```

Plain text and simple auth issues

node.js – express – SSL auth

```
var express = require('express'), routes = require('./routes'), fs = require('fs')
var opts = {
        key: fs.readFileSync('ssl/server/keys/server.key'),
        cert: fs.readFileSync('ssl/server/certificates/server.crt'),
        ca: fs.readFileSync('ssl/ca/ca.crt'),
        crl: fs.readFileSync('ssl/ca/ca.crl'),
        requestCert: true,
        rejectUnauthorized: true
        passphrase: "pwd" // <<<< really here?
};

var app = module.exports = express.createServer(opts);

app.configure(function(){
    app.set('views', __dirname + '/views');
    ...
});

app.get('/', routes.index);
app.listen(8443);
```

node.js – express – passport.js

- provides API for authentication and authorization
- authentication:
    - LocalStrategy
    - OpenIDStrategy
    - OAuth / FacebookStrategy

node.js – express – authorization

```
var users = [
    { id: 1, name: "user1", role: "admin" },
    { id: 2, name: "user2", role: "common" },
];

function loadUser(req, res, next) {
    req.userData = users[req.params.user];
    return next();
}

function requireRole(role) {
    return function (req, res, next) {
        if (req.user.role === role) {
        return next();
    } else {
        return next(new Error("Unauthorized"));
    }
};}
```

node.js – express – authorization

```
app.get("/users/:user", loadUser, function (req, res) {
    res.send(req.user.name);
});

app.del("/users/:user", requireRole("admin"), loadUser,
function (req,res) {
    res.send("User deleted");
});
```

node.js – express – logging

OWASP will tell you what should be logged :)

https://www.owasp.org/index.php/Logging_Cheat_Sheet

- authentication & authorisation
- session management
- errors & weirdo events
- events (startups, shutdowns, slowdowns etc)
- high risk functionalities (payments, privileges, admins)

node.js – express – logging

Try Winston module (Github -> flatiron/winston)

- logging to console
- logging to file
- sending logs over HTTP
- CouchDB, Redis, MongoDB, Riak etc

node.js – express – sessions

```
var express = require('express');
var app = express();
var RedisStore = require('connect-redis')(express);

app.use(express.cookieParser());
app.use(express.session({
  store: new RedisStore({
    host: '127.0.0.2',
    port: 6379,
    db: 3,
    pass: 'pwd'
  }),
  secret: 'this-is-very-secret'
}));

app.get('/somewhere', function(req, res) {
  res.send('In the middle of nowhere');
});

app.listen(process.env.PORT || 8080);
```

node.js – common threats

- CSRF
- input validation
- XSS
- DoS
- ReDoS
- HPP
- request size

node.js – monitoring anyone?

- is app functional? :)

- is app overloaded?

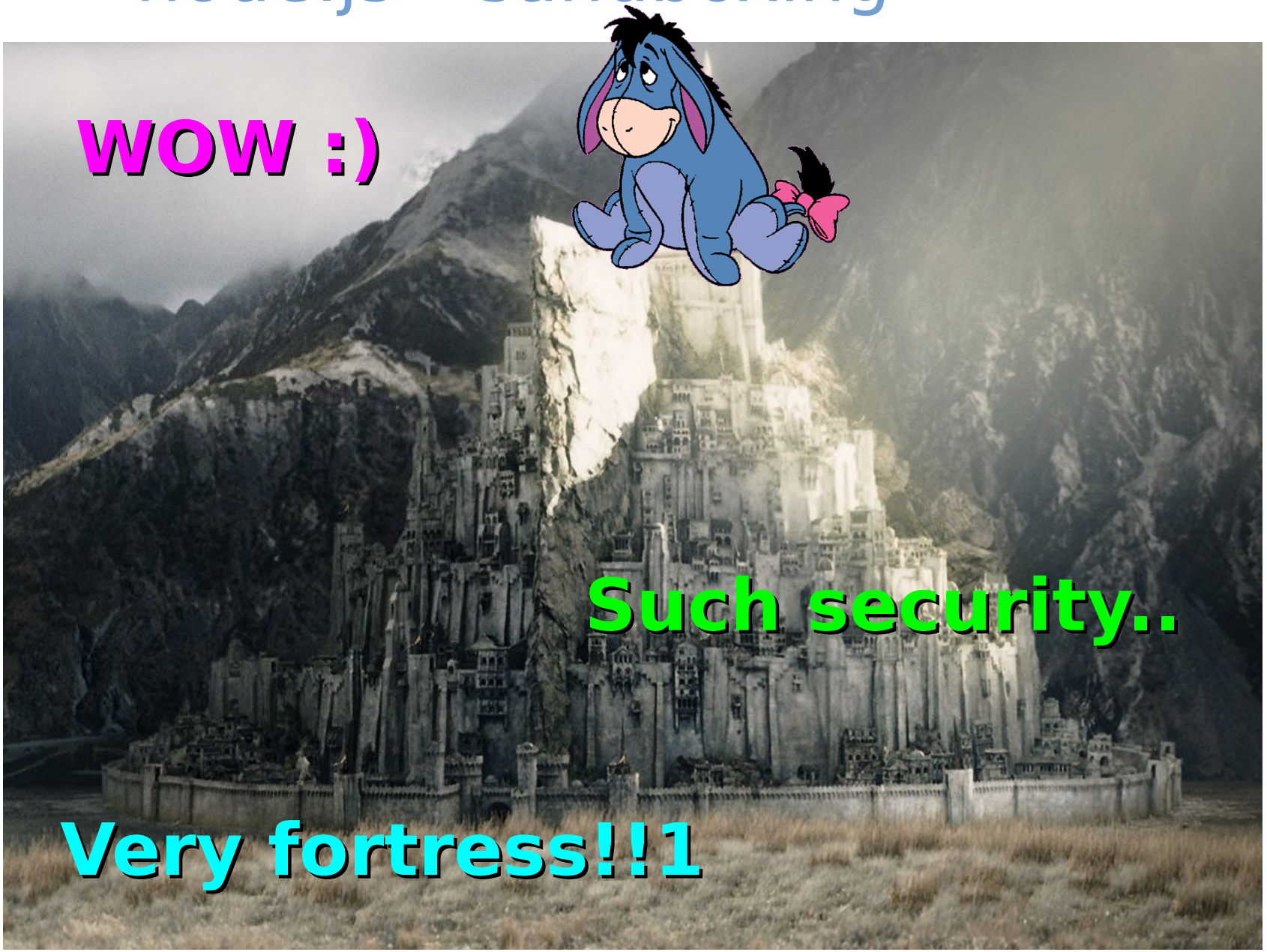
- app should provide monitoring interface

- how many errors caught?

- are forks alive and OK?

node.js – sandboxing

node.js – sandboxing



node.js – sandboxing

SElinux sandbox:

- legit r/w from stdin/out + only define FDs
- no network access
- no access to any other processes files
- cgroups friendly :)
- lightweight!

node.js – sandboxing

libvirtd sandbox:

- use LXC, Qemu or KVM

- provides high level API

- don't need to know virt internals

- integrates with systemd inside the sandbox

- virt-sandbox -c lxc:/// /bin/sh

node.js – sandboxing

Docker:

- very easy learning curve – just run & go

- it just works

- big community

- growing rapidly

- almost stable ;)

node.js – one more thing

Just...

node.js – one more thing

Just...

Don't run as `root`!!!

node.js – tracing execution

- SmartOS / Joyent: debugging

- Bunyan / Dtrace

- strace of course...

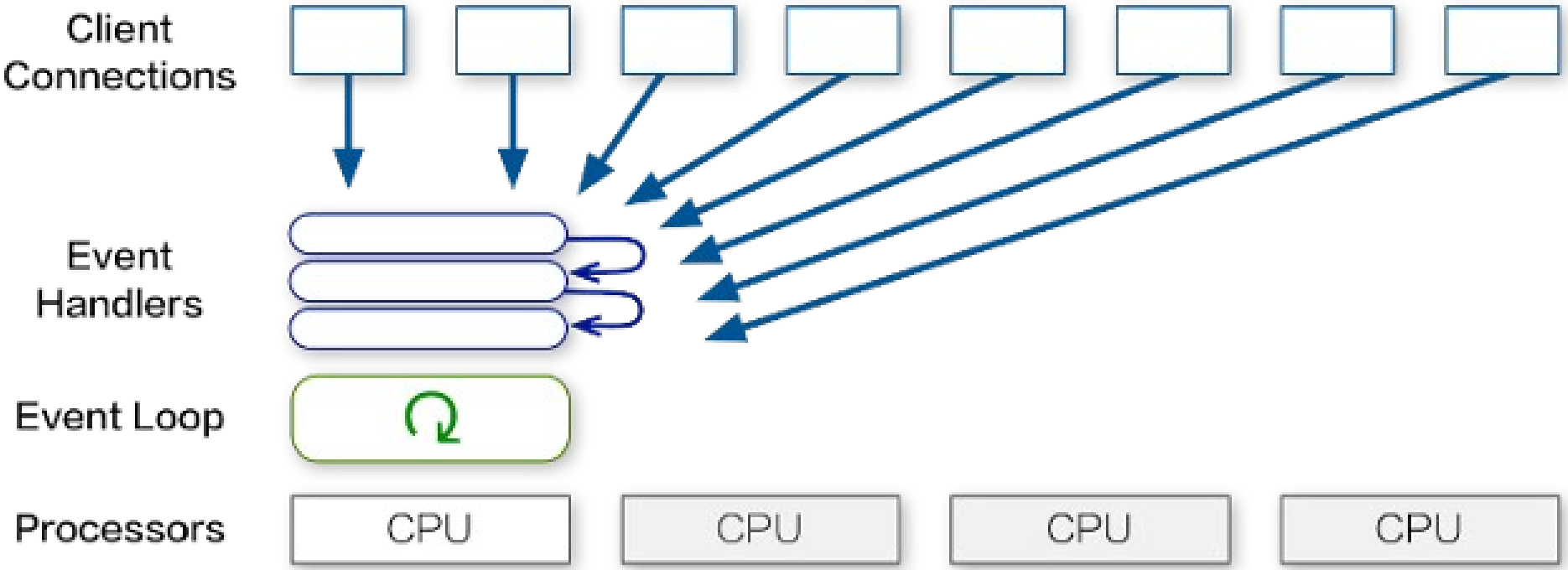
node.js – testing

- maybe some interface for white-box pentests?

- unit-testing 4 the sake! (Mocha, supertest, should.js)

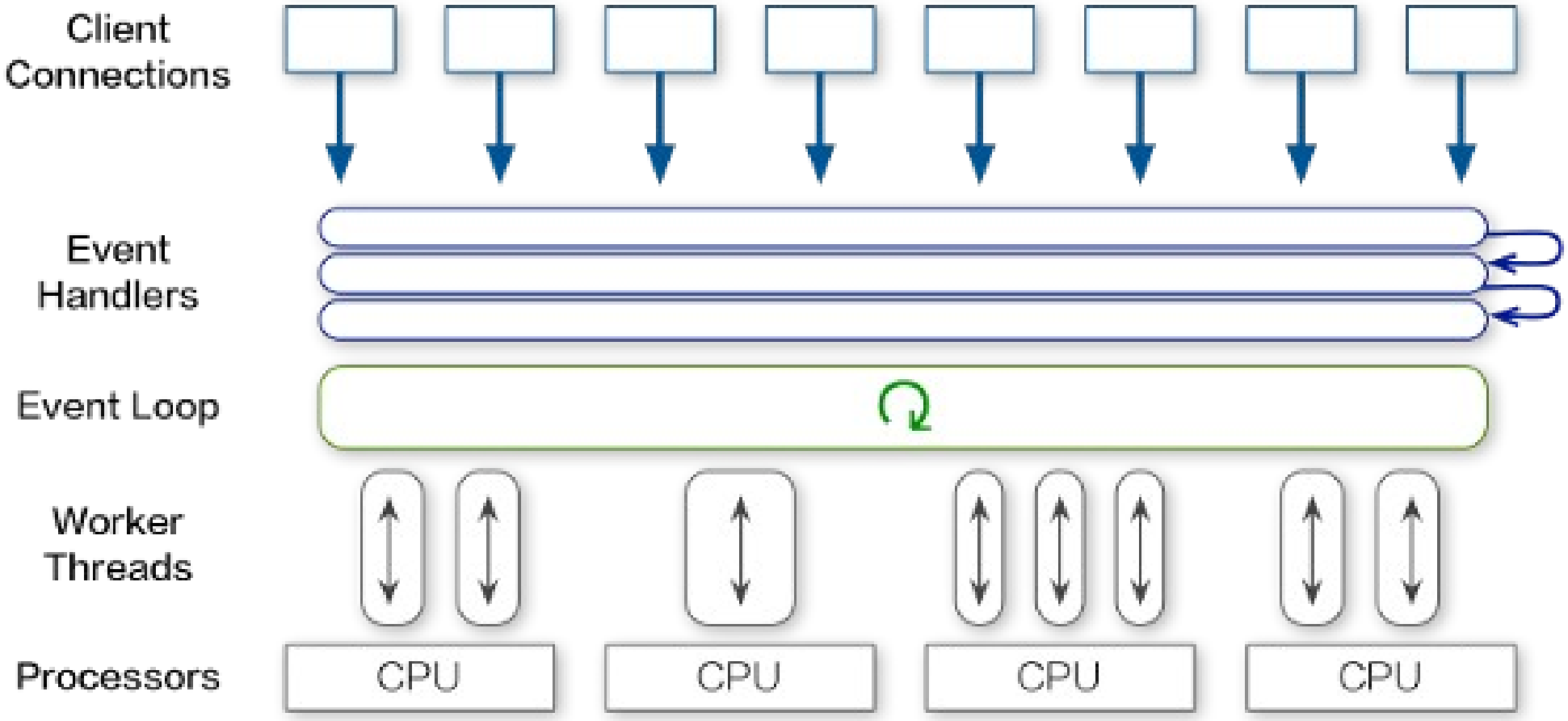
- OWASP Zed Attack Proxy

scaling node.js – cluster module



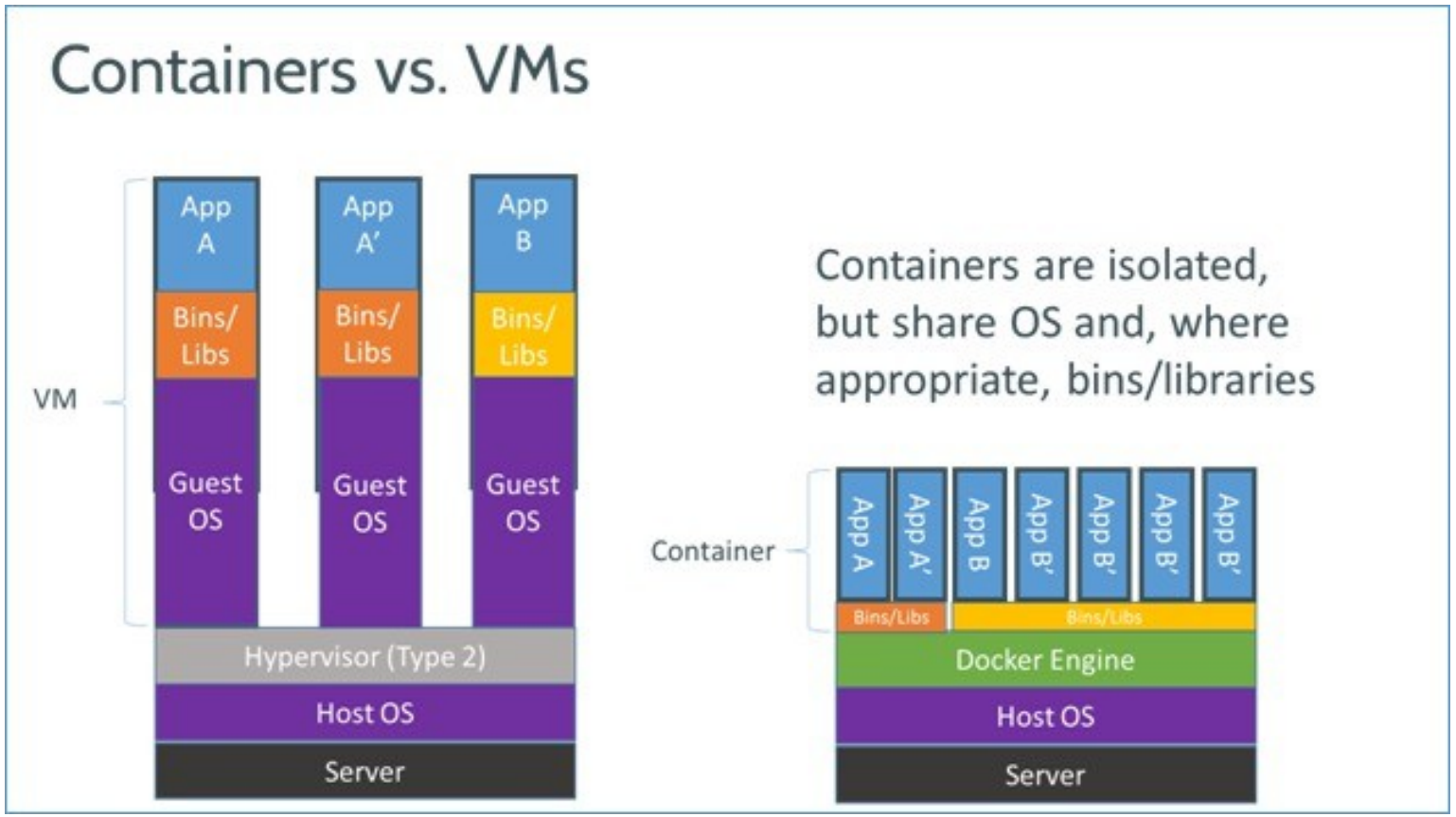
http://aosabook.org

scaling node.js – cluster module



http://aosabook.org

scaling node.js – containers



scaling node.js – resources

Just use cgroups

node.js performance

- c10k problem!

- paypal – release the Kraken & stories

So what do you think about JS?

- JS is for children? wrong, children aren't async ;)

- JS is slow? wrong – V8!

- JS is not scalable? wrong – we'll JS the world!

- JS is insecure? wrong – people do

node.js.learning

- Node Security Book

- OWASP Node Goat (top10)

- nodesecurity.io (Twitter, RSS)

Infosec & meet.js meetups @krakow
meetup.com

Docker workshops with node.js!
#dockerkrk #nodekrk

Thank you :)

Any Qs?

http://maciek.lasyk.info/sysop

maciek@lasyk.info

@docent-net