BayNode
The Bay Area Node.js Meetup

March 2015

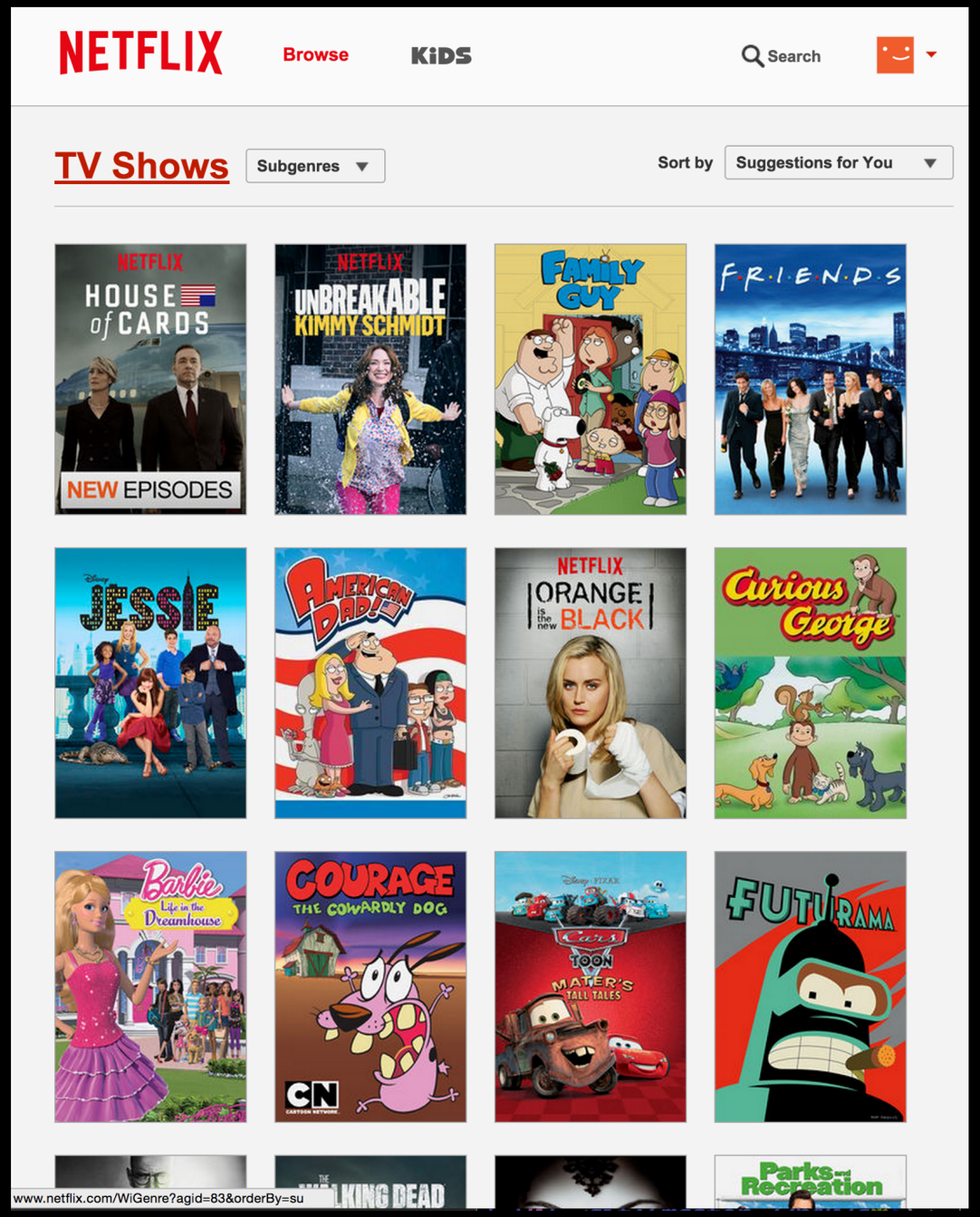
Observable Applications w/ Node.js

Yunong Xiao, Senior Node.js Engineer,
UI Platform, Netflix
@yunongx, yunong@netflix.com

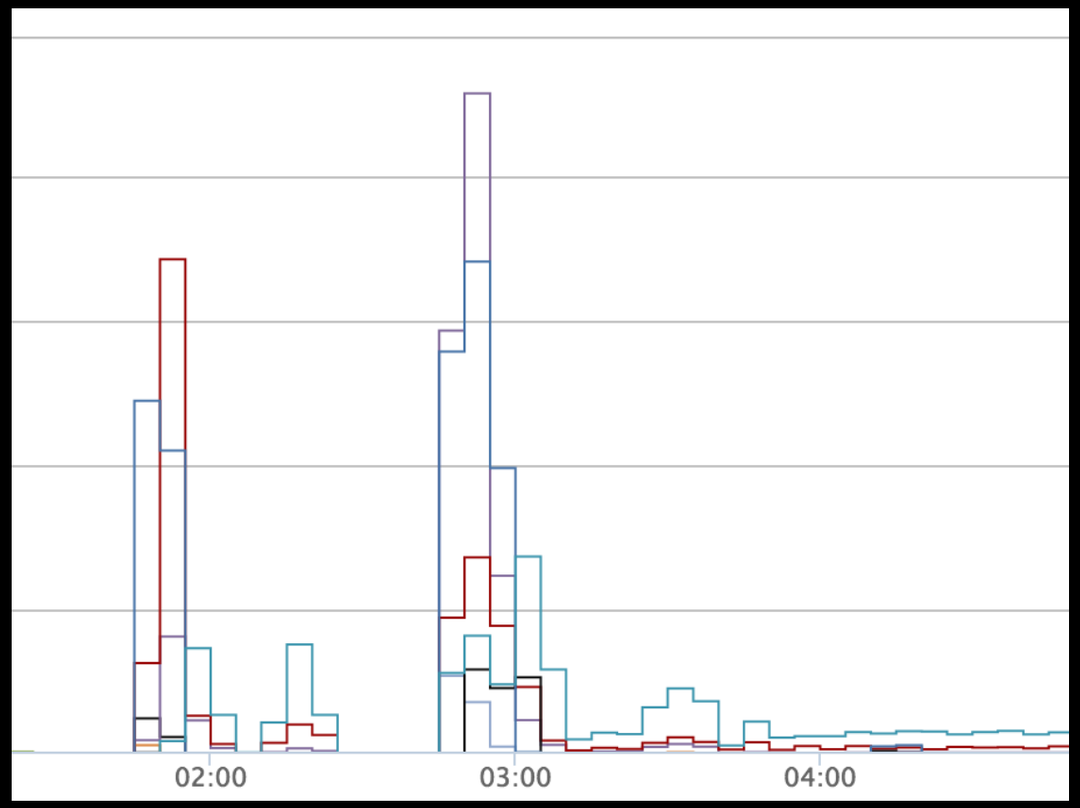
NETFLIX

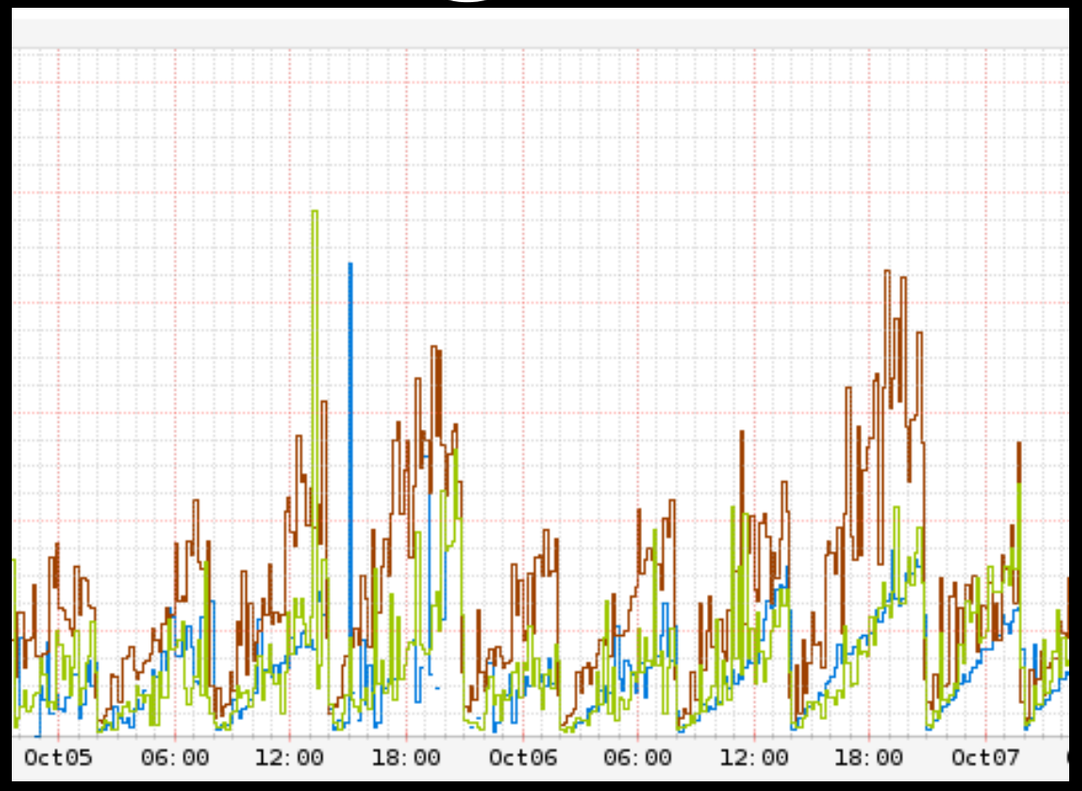
Node.js @ Netflix

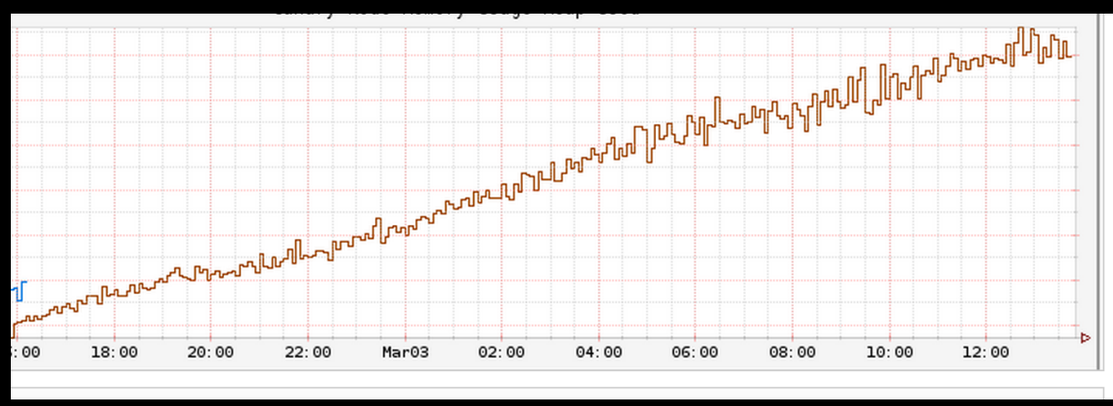
- Website www.netflix.com
  - Discovery
  - Playback
  - Acquisition
  - Account Management
- Internal Services
- TV/Mobile/Devices

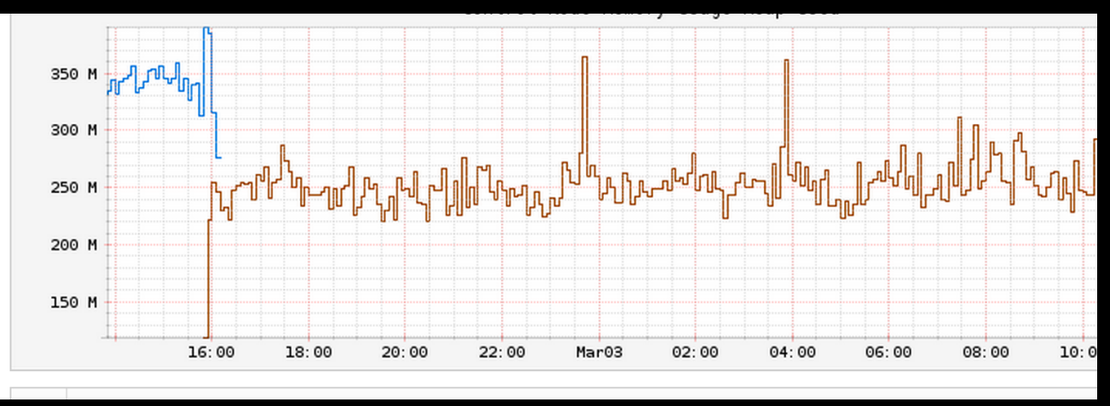


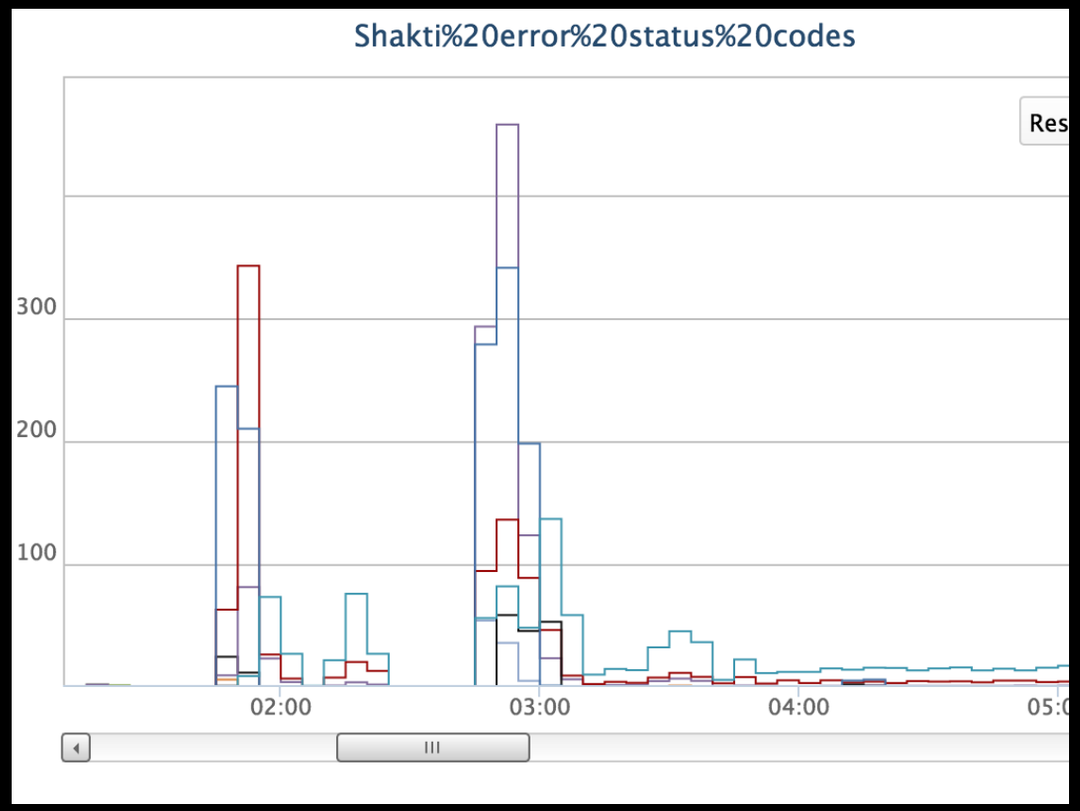
What's Wrong?


02:00
03:00
04:00

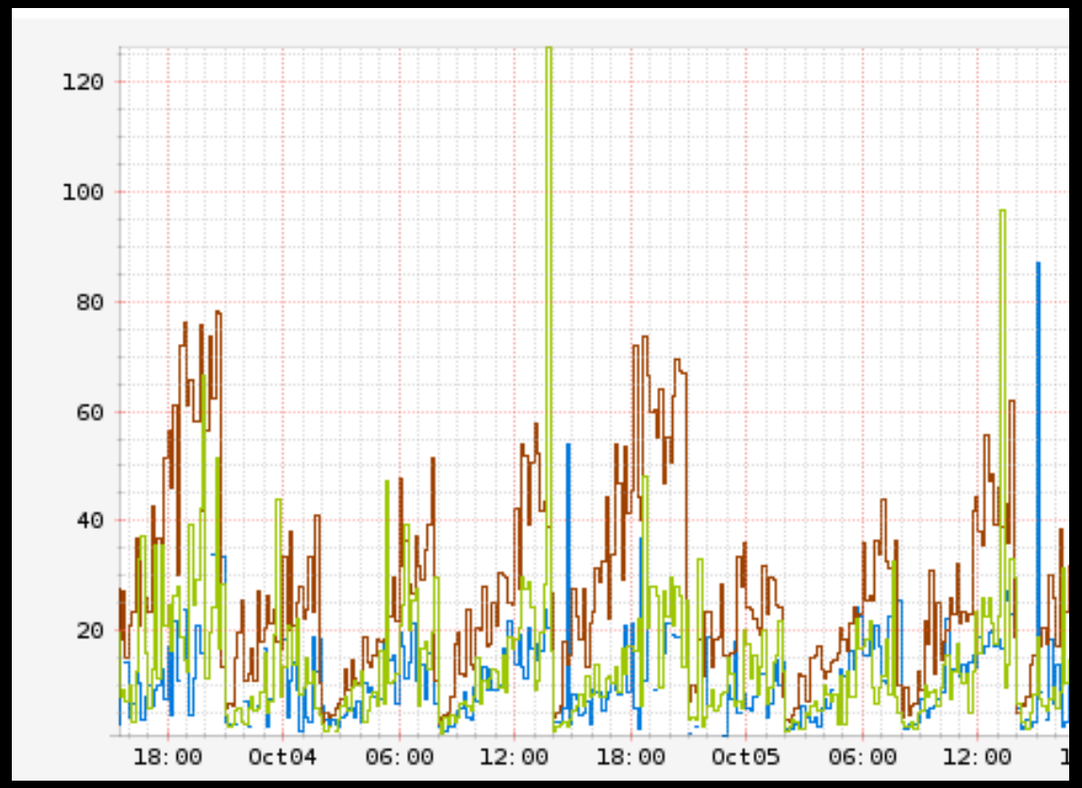

Oct05
06:00
12:00
18:00
Oct06
06:00
12:00
18:00
Oct07


18:00
20:00
22:00
Mar03
02:00
04:00
06:00
08:00
10:00
12:00

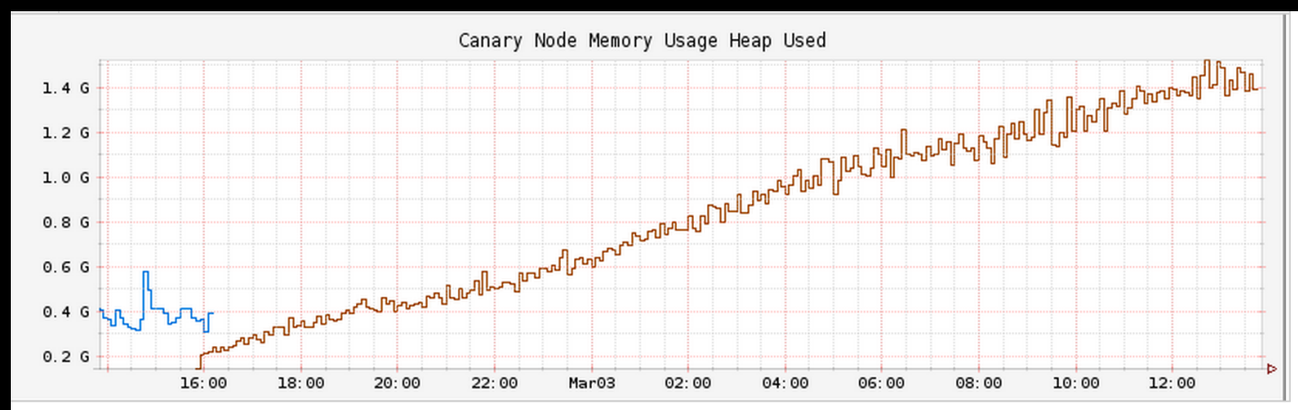

350 M
300 M
250 M
200 M
150 M
16:00
18:00
20:00
22:00
Mar03
02:00
04:00
06:00
08:00

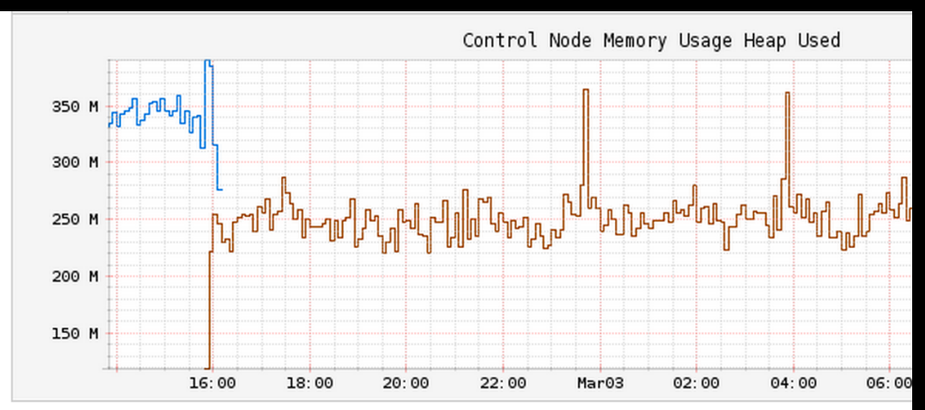


Increased Errors



Increased Latency





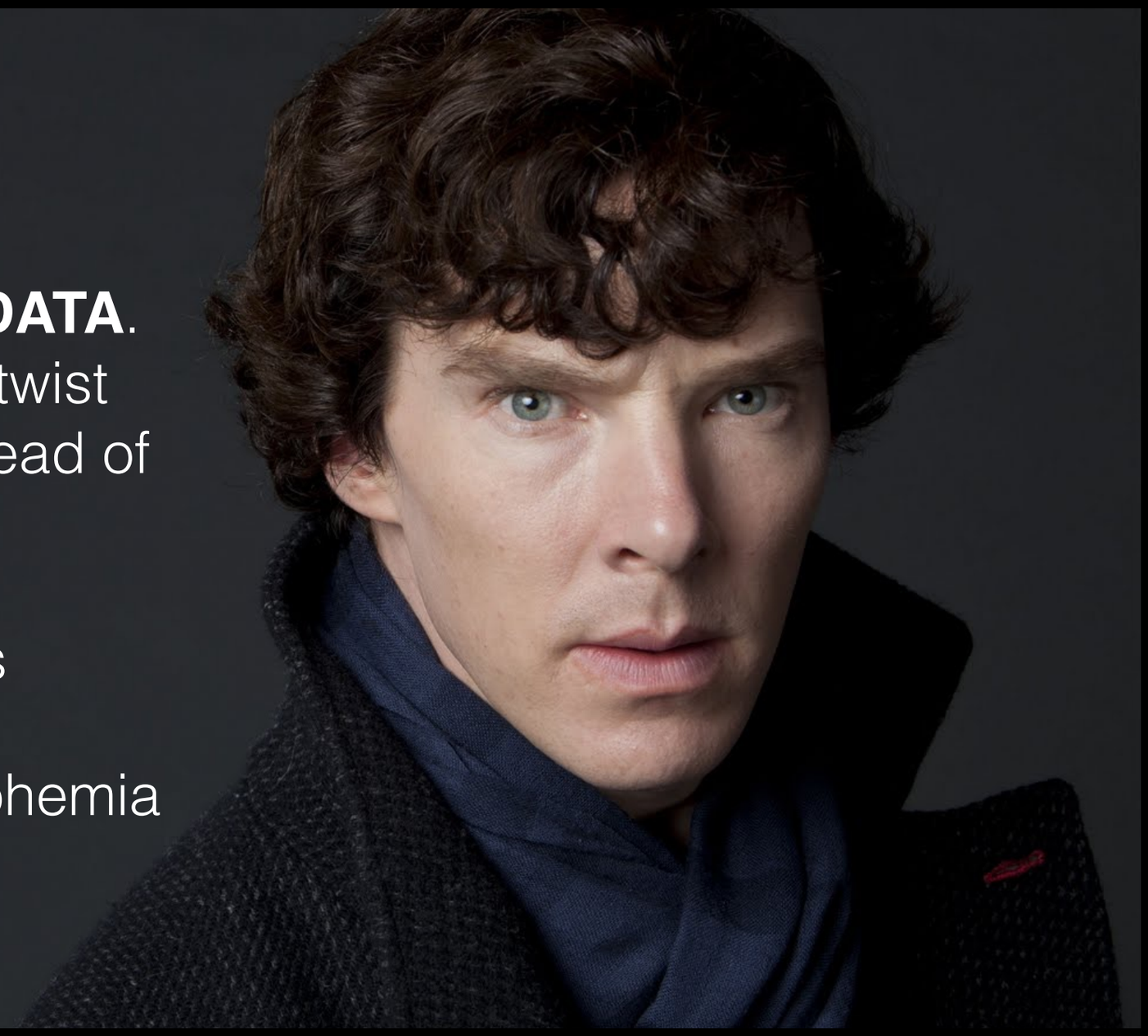
Memory Leak

What Now?

“It is a capital mistake to theorize before one has **DATA**. Insensibly one begins to twist facts to suit theories, instead of theories to suit facts.”

Sherlock Holmes

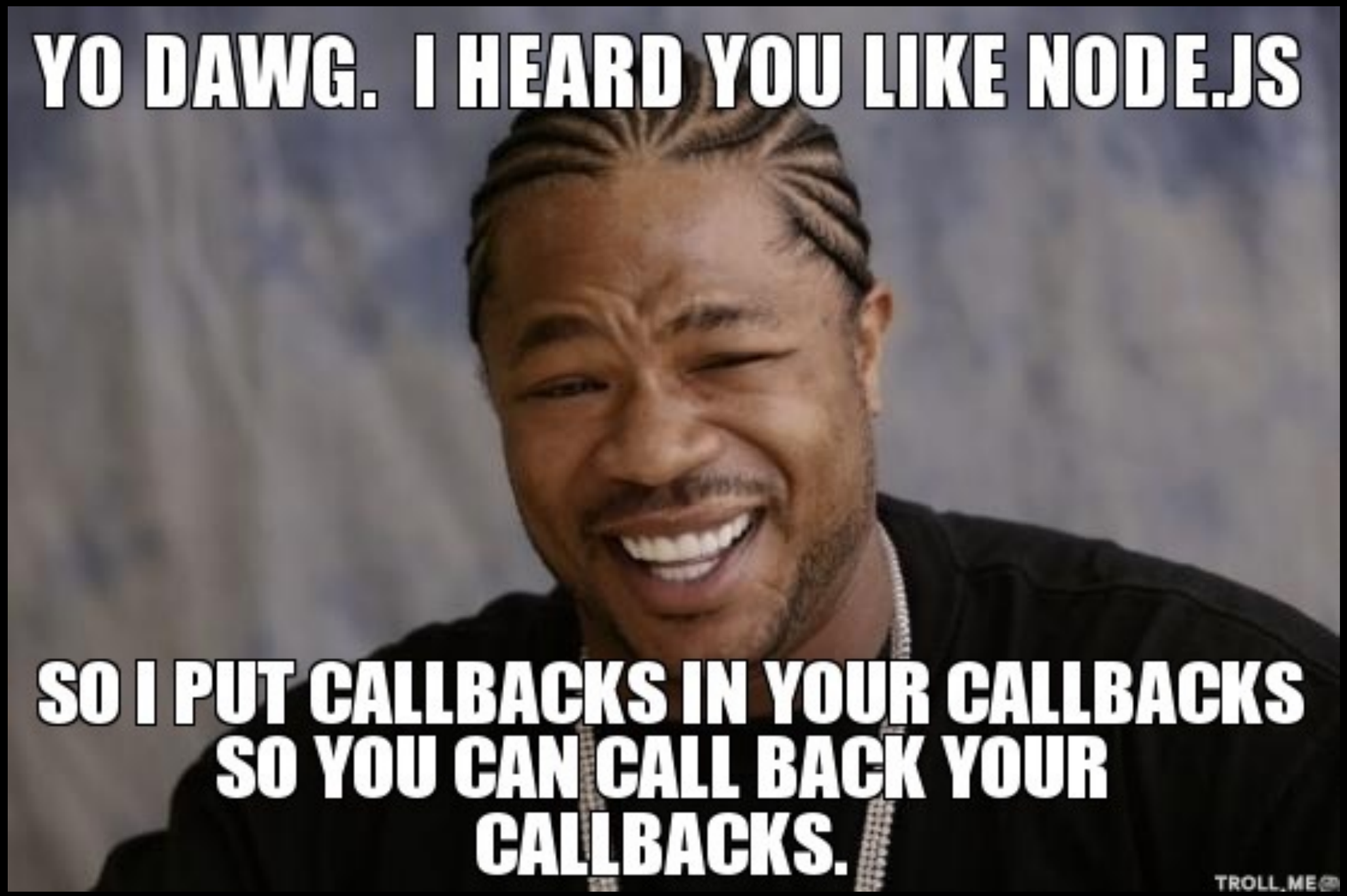
-A Scandal in Bohemia



Data Data Data

Node.js Modules

```
var db = require('./db/callbackDb');

db.set('key1', 'value1', function(err) {
    if(err) throw err;

    db.set('key2', 'value2', function(err) {
        if(err) throw err;

        db.set('key3', 'value3', function(err) {
            if(err) throw err;

            var str = '';
            db.get('key1', function(err, value) {
                if(err) throw err;

                str += value + ' - ';

                console.log(str);
            });
        });
    });
});
```


YO DAWG. I HEARD YOU LIKE NODE.JS
SO I PUT CALLBACKS IN YOUR CALLBACKS
SO YOU CAN CALL BACK YOUR
CALLBACKS.
TROLL.ME

```
var vasync = require('vasync');

var results = vasync.pipeline({funcs: [
    function one(_, cb) {
        // some async function one
        return cb();
    },
    function two(_, cb) {
        // some async function two
        return cb();
    },
    function three(_, cb) {
        // some async function three
        return cb();
    }
], args: {}}, function cb(err, res) {
    console.log('finished pipeline', err, res);
});
```

vasync

- https://github.com/davepacheco/node-vasync
- Similar to caolan/async.
- Added observability.

```
{ operations:
   [ { func: [Function: one],
       funcname: 'one',
       status: 'fail',
       err: [Error],
       result: undefined },
     { func: [Function: two],
       funcname: 'two',
       status: 'ok',
       err: undefined,
       result: undefined },
     { func: [Function: three], funcname: 'three', status: 'pending' } ],
  successes: [ undefined ],
  ndone: 2,
  nerrors: 1 }
```

async f() state

failed f()

pending f()

successful f()

finished f()

# of errors

vasync

Async workflow management is error prone

- Make API requests.
- Persist state to file system.
- Query database.
- Update caches.

Example

```
1 var vasync = require('vasync');
2
3 var results = vasync.pipeline({funcs: [
4     function one(_, cb) {
5         console.log('some async function one');
6         return cb();
7     },
8     function two(_, cb) {
9         console.log('some async function two');
10        return cb();
11    },
12    function three(_, cb) {
13        console.log('some async function three');
14        // No cb due to bug
15        // return cb();
16    }
17 ], args: {}}, function cb(err, res) {
18    console.log('finished pipeline', err, res);
19 });
20
21 setInterval(function() { console.log('pipeline status', results); }, 1000);
```

cb() not invoked

Results

```
{ operations:
   [ { func: [Function: one],
       funcname: 'one',
       status: 'ok',
       err: undefined,
       result: undefined },
     { func: [Function: two],
       funcname: 'two',
       status: 'ok',
       err: undefined,
       result: undefined },
     { func: [Function: three], funcname: 'three', status: 'pending' } ],
  successes: [ undefined, undefined ],
  ndone: 2,
  nerrors: 0 }
```

function three pending

How Do I See this in Prod?

- Logs
- Core Dumps
- REPL
- HTTP API

node-bunyan

- https://github.com/trentm/node-bunyan

- Streaming JSON logging library for JS. (node and browser)

Streaming JSON

- One JSON object per line. e.g. Twitter's streaming API.
- Perfect for machine processing. i.e. works with Unix tools such as grep(1), cut(1), …
- Illegible for Humans

Streaming JSON


```
{"name":"shakti","hostname":"shakti-prod-i-232a3529","pid":7567,"component":"restify",
"req_id":"9d92d710-b961-11e4-a9bd-a188e49c3b0c","level":30,"method":"GET","fullResourc
eUrl":"/shakti/90409bc7/memberContext","statusCode":200,"elapsedTime":305,"msg":"[nf-h
ttp]: memberContext request success","time":"2015-02-21T00:36:03.763Z","v":0}
{"name":"shakti","hostname":"shakti-prod-i-232a3529","pid":7567,"component":"restify",
"req_id":"9d92d710-b961-11e4-a9bd-a188e49c3b0c","level":30,"msg":"Resolving action sha
rd: search","time":"2015-02-21T00:36:03.763Z","v":0}
{"name":"shakti","hostname":"shakti-prod-i-232a3529","pid":7567,"component":"restify",
"req_id":"9d8f0680-b961-11e4-a9bd-a188e49c3b0c","level":30,"method":"GET","fullResourc
eUrl":"/shakti/90409bc7/memberContext","statusCode":200,"elapsedTime":325,"msg":"[nf-h
ttp]: memberContext request success","time":"2015-02-21T00:36:03.767Z","v":0}
{"name":"shakti","hostname":"shakti-prod-i-232a3529","pid":7567,"component":"restify",
"req_id":"9d3aa4f0-b961-11e4-a9bd-a188e49c3b0c","level":30,"method":"GET","fullResourc
eUrl":"/shakti/90409bc7/usermessages?authURL=1424478962983.IuqDa5%2BZsu5Iq%2FF4oaWe%2B
NJyaOs%3D","statusCode":200,"elapsedTime":360,"msg":"[nf-http]: umaModel request succe
ss","time":"2015-02-21T00:36:03.769Z","v":0}
```

bunyan(1) CLI

```
└─[0] <> cat /tmp/foo| bunyan
[2015-02-21T00:36:03.763Z]  INFO: shakti/restify/7567 on shakti-prod-i-232a3529: [nf-h
ttp]: memberContext request success (req_id=9d92d710-b961-11e4-a9bd-a188e49c3b0c, meth
od=GET, fullResourceUrl=/shakti/90409bc7/memberContext, statusCode=200, elapsedTime=30
5)
[2015-02-21T00:36:03.763Z] DEBUG: shakti/restify/7567 on shakti-prod-i-232a3529: Resol
ving action shard: search (req_id=9d92d710-b961-11e4-a9bd-a188e49c3b0c)
[2015-02-21T00:36:03.767Z] TRACE: shakti/restify/7567 on shakti-prod-i-232a3529: [nf-h
ttp]: memberContext request success (req_id=9d8f0680-b961-11e4-a9bd-a188e49c3b0c, meth
od=GET, fullResourceUrl=/shakti/90409bc7/memberContext, statusCode=200, elapsedTime=32
5)
[2015-02-21T00:36:03.769Z]  WARN: shakti/restify/7567 on shakti-prod-i-232a3529: [nf-h
ttp]: umaModel request success (req_id=9d3aa4f0-b961-11e4-a9bd-a188e49c3b0c, method=GE
T, statusCode=200, elapsedTime=360)
    fullResourceUrl: /shakti/90409bc7/usermessages?authURL=1424478962983.IuqDa5%2BZsu5
Iq%2FF4oaWe%2BNJyaOs%3D
[2015-02-21T00:36:03.769Z] ERROR: shakti/restify/7567 on shakti-prod-i-232a3529: [nf-h
ttp]: umaModel request success (req_id=9d3aa4f0-b961-11e4-a9bd-a188e49c3b0c, method=GE
T, statusCode=200, elapsedTime=360)
    fullResourceUrl: /shakti/90409bc7/usermessages?authURL=1424478962983.IuqDa5%2BZsu5
Iq%2FF4oaWe%2BNJyaOs%3D
```

Features

- Lightweight API
- Log levels: trace, debug, info, warn, error, fatal
- Extensible Streams interface.
- Custom object rendering with serializers.
- DTrace support.

Demo

DTrace

- Dynamic tracing framework.
- Available on Mac OS X (dev env). `man dtrace`
- Node.js DTrace USDT provider: https://github.com/chrisa/node-dtrace-provider
- Pure JS probes.

bunyan -p

Demo

Production Tracing

- Production environment is Ubuntu Linux on EC2.
- Future work: get probe access. Possibilities:
  - SystemTap
  - LTTng (Linux Tracing Toolkit Next Gen)
  - perf_events markers (currently a proposed patch)

restify

- http://restifyjs.com
- Production tested REST framework.
- Observability.
- Metrics.
- First class Bunyan integration.
- DTrace.

Vanilla App

```
1 var restify = require('restify');
2
3 var server = restify.createServer({});
4
5 server.use(restify.queryParser());
6
7 server.get(
8     '/hello',
9     function hello(req, res, next) {
10         res.send({hi: req.query.name});
11         return next();
12     }
13 );
14
15 server.listen(1337, function () {
16     console.log('server started', server.url);
17 });
```

Vanilla App

```
└─[1] <> node vanilla.js&
[1] 5683
┌─[yunong@lgml-yunong] - [~/workspace/restify-demo] - [2015-03-17 12:25:56]
└─[0] <> server started

┌─[yunong@lgml-yunong] - [~/workspace/restify-demo] - [2015-03-17 12:25:58]
└─[0] <> curl 'localhost:1337/hello?name=yunong'
{"hi":"yunong"}%
```

Logging

Request Capture Stream

- Tradeoffs, Perf vs Verbosity.
- Captures all log statements including trace in memory.
- Dump all logs only on error.

Audit Logs

```
[2014-12-10T21:04:18.251Z]  INFO: shakti/restify-audit/21152 on shakti-prod-i-97f6627b: handled: 200 (req_id=16c66e20-80b0-11e4-9f6b-cf625d17b873, audit=true
, latency=7003, secure=true, _audit=true, req.version=*)
    GET /?locale=en-US&nmrs=1418245445135 HTTP/1.1
    host: www.netflix.com
    x-netflix.appinfo.name: SHAKTI
    accept-language: pt-BR,pt;q=0.8,en-US;q=0.6,en;q=0.4
    x-forwarded-for: 129.170.194.148
    x-forwarded-port: 443
    x-netflix.edge.server.timestamp: 1418245446126
    accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
    x-netflix.client-host: www.netflix.com
    user-agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.71 Safari/537.36
    x-forwarded-proto: https
    x-forwarded-host: www.netflix.com
    netflix.nfhttpclient.version: 1.0
    x-netflix-httpclientname: shakti-prod
    --
    HTTP/1.1 200 OK
    x-netflix.client.instance: i-97f6627b
    x-frame-options: DENY
    content-type: text/html
    --
    req.timers: {
      "bunyan": 3539,
      "reqResTracker": 351,
      "resLocals": 55,
      "parseQueryString": 4015,
      "instanceHeaders": 1929,
      "xForwardedProto": 98,
      "httpsRedirector": 3353,
      "readBody": 225,
      "parseBody": 43,
      "xframe": 36,
      "restifyCookieParser": 442,
      "springbank": 46,
      "requestContextFp": 3262457,
      "membershipStatus": 211,
      "fetchModels": 111865,
      "htmlContentType": 205,
      "redirectInAppToEcWeb": 1250,
      "globalPageRedirect": 51,
      "openSiteRedirectOrDisplay": 54,
      "moneyballModeRedirects": 141,
      "nonMemberContext": 106,
      "setContextData": 3266,
      "evidonSetup": 96,
      "continuumImgUrls": 110,
      "controlTemplateImgUrls": 190,
      "ethnioSetup": 6234,
      "addCspHeader": 43,
      "redirectCurrentMember": 33,
      "interimEcWebInterceptor": 270,
      "render": 3575516
    }
```

URL

req latency

status code

request ID (UUID)

request headers

response headers

individual handler timers

```
req.timers: {
  "bunyan": 3539,
  "reqResTracker": 351,
  "resLocals": 55,
  "parseQueryString": 4015,
  "instanceHeaders": 1929,
  "xForwardedProto": 98,
  "httpsRedirector": 3353,
  "readBody": 225,
  "parseBody": 43,
  "xframe": 36,
  "restifyCookieParser": 442,
  "springbank": 46,
  "requestContextFp": 3262457,
  "membershipStatus": 211,
  "fetchModels": 111865,
  "htmlContentType": 205,
  "redirectInAppToEcWeb": 1250,
  "globalPageRedirect": 51,
  "openSiteRedirectOrDisplay": 54,
  "moneyballModeRedirects": 141,
  "nonMemberContext": 106,
  "setContextData": 3266,
  "evidonSetup": 96,
  "continuumImgUrls": 110,
  "controlTemplateImgUrls": 190,
  "ethnioSetup": 6234,
  "addCspHeader": 43,
  "redirectCurrentMember": 33,
  "interimEcWebInterceptor": 270,
  "render": 3575516
```

Scoped Child Loggers

Logging

- Native Bunyan integration.
- Request capture stream.
- Audit logs with each req.
- Scoped child loggers with each req.

restify + Bunyan

- Streaming JSON.
- Processing using Unix tools is easy.
- Helpful tools:
  - Unix: cut(1), wc(1), grep(1), awk(1), perl(1), …
  - JSON: https://github.com/trentm/json (npm install -g json)
  - daggr: https://github.com/joyent/daggr (npm install -g daggr)

Examples

- Find all logs for a specific request.
- Count the # of non-200 responses.
- Show all requests that took longer than 200ms.

Advanced Example

Request latency distribution by URL

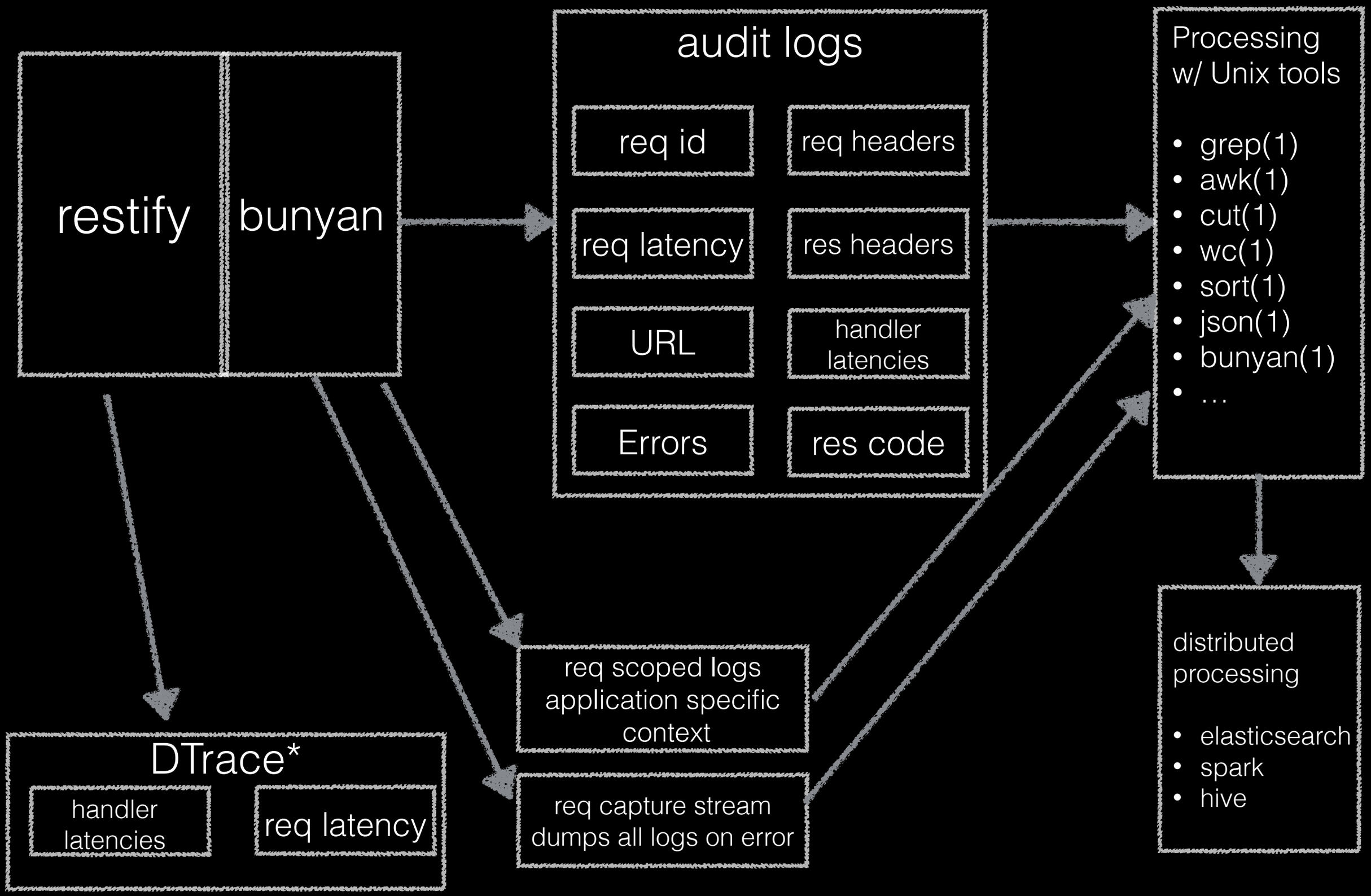
```
└─[0] <> cat 2015-02-21-shakti-prod-v017-i-232a3529_shakti_shakti.log_0236.log |  grep restify-audit| bunyan -0 --stri
ct| json -ga latency req.url  | awk '{print $1 " " substr($2,0,6)}' | daggr -k 2 -v 1 quantize
/WiVie
           value  ------------- Distribution ------------- count
               1 |                                         0
               2 |                                         1
               4 |                                         0
               8 |                                         0
              16 |                                         0
              32 |                                         0
              64 |@@                                       43
             128 |@@@@@@@@@@@@@@@@                         351
             256 |@@@@@@@@@@@@@@@@@@@@                     399
             512 |@@@                                      58
            1024 |                                         5
            2048 |                                         1
            4096 |                                         0

/searc
           value  ------------- Distribution ------------- count
             0.5 |                                         0
               1 |                                         13
               2 |                                         18
               4 |                                         4
               8 |                                         1
              16 |                                         0
              32 |                                         0
              64 |                                         0
             128 |@@@@@@@@@@@@@@@@@@@@                     1285
             256 |@@@@@@@@@@@@@@@@@@@@                     1270
             512 |@@                                       167
            1024 |                                         15
```

DTrace

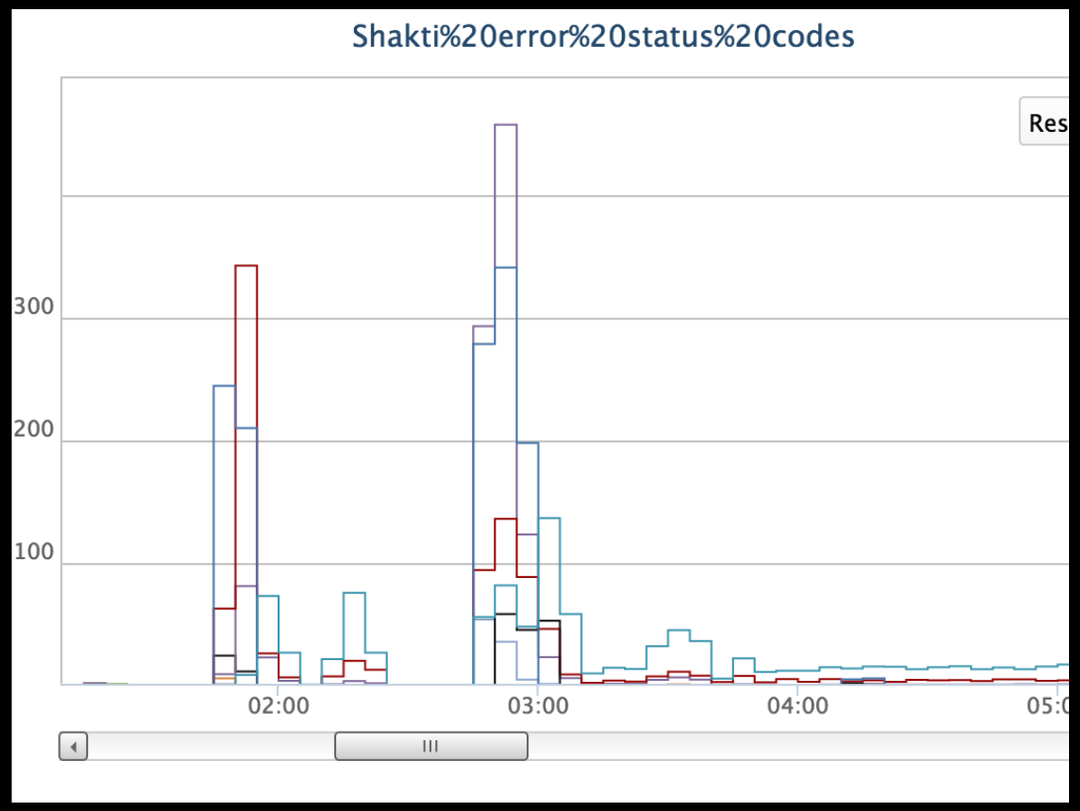
```
┌─[yunong@lgml-yunong] - [~] - [2015-03-17 08:15:36]
└─[0] <> sudo dtrace -l -P 'restify*'
   ID   PROVIDER            MODULE                          FUNCTION NAME
12088 restify11084   mod-0x100d05d60                route-start route-start
12089 restify11084   mod-0x100d05d60              handler-start handler-start
12090 restify11084   mod-0x100d05d60               handler-done handler-done
12091 restify11084   mod-0x100d05d60                route-done route-done
12095 restify11084   mod-0x100d05d60             client-request client-request
16505 restify11084   mod-0x100d05d60            client-response client-response
16506 restify11084   mod-0x100d05d60               client-error client-error
```

Demo

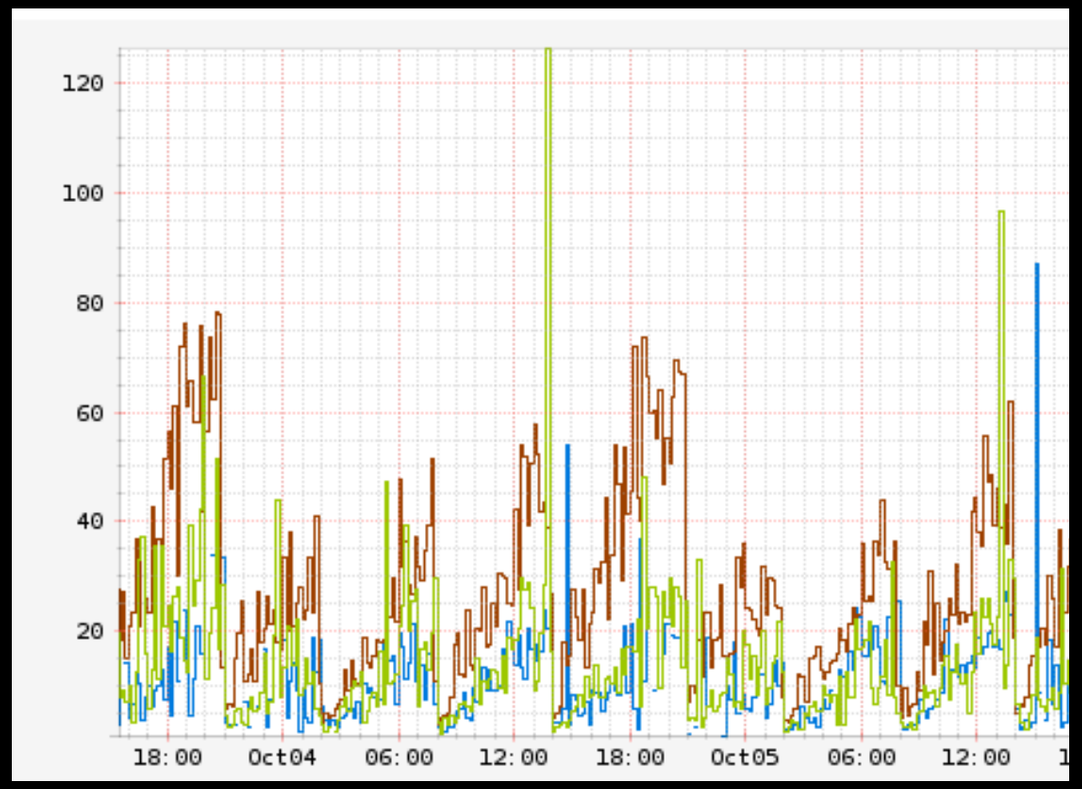
restify metrics



* Where available



Increased Errors



Increased Latency

restify

Interested? Call for contributors.

restifyjs.com

The Killer Combo

- vasync: https://github.com/davepacheco/node-vasync
  - Observable async operations.
- bunyan: https://github.com/trentm/node-bunyan
  - Streaming JSON logs.
- restify: restifyjs.com
  - Observable REST applications.
- Unix Tools
  - Stream of text model. Many tools to manipulate JSON logs.

Data Data Data

BayNode
The Bay Area Node.js Meetup

Thanks

March 2015

- Questions?

- yunong@netflix.com
- @yunongx

- http://restifyjs.com

NETFLIX