Accelerate Your Digital Transformation

About Michael Dawson

Loves the web and building software (with Node.js!)

Senior Software Developer @ IBM

IBM Runtime Technologies Node.js Technical Lead

Node.js collaborator and CTC/TSC member

Active in LTS, build, benchmarking, api and post-mortem working groups

Contact me:

michael_dawson@ca.ibm.com

Twitter: @mhdawson1

https://www.linkedin.com/in/michael-dawson-6051282

Agenda

- Common Adoption Patterns
- Locality of Data
- Journey to the Cloud (aka Digital Transformation)
- Community work to remove roadblocks
- Summary and Wrap up

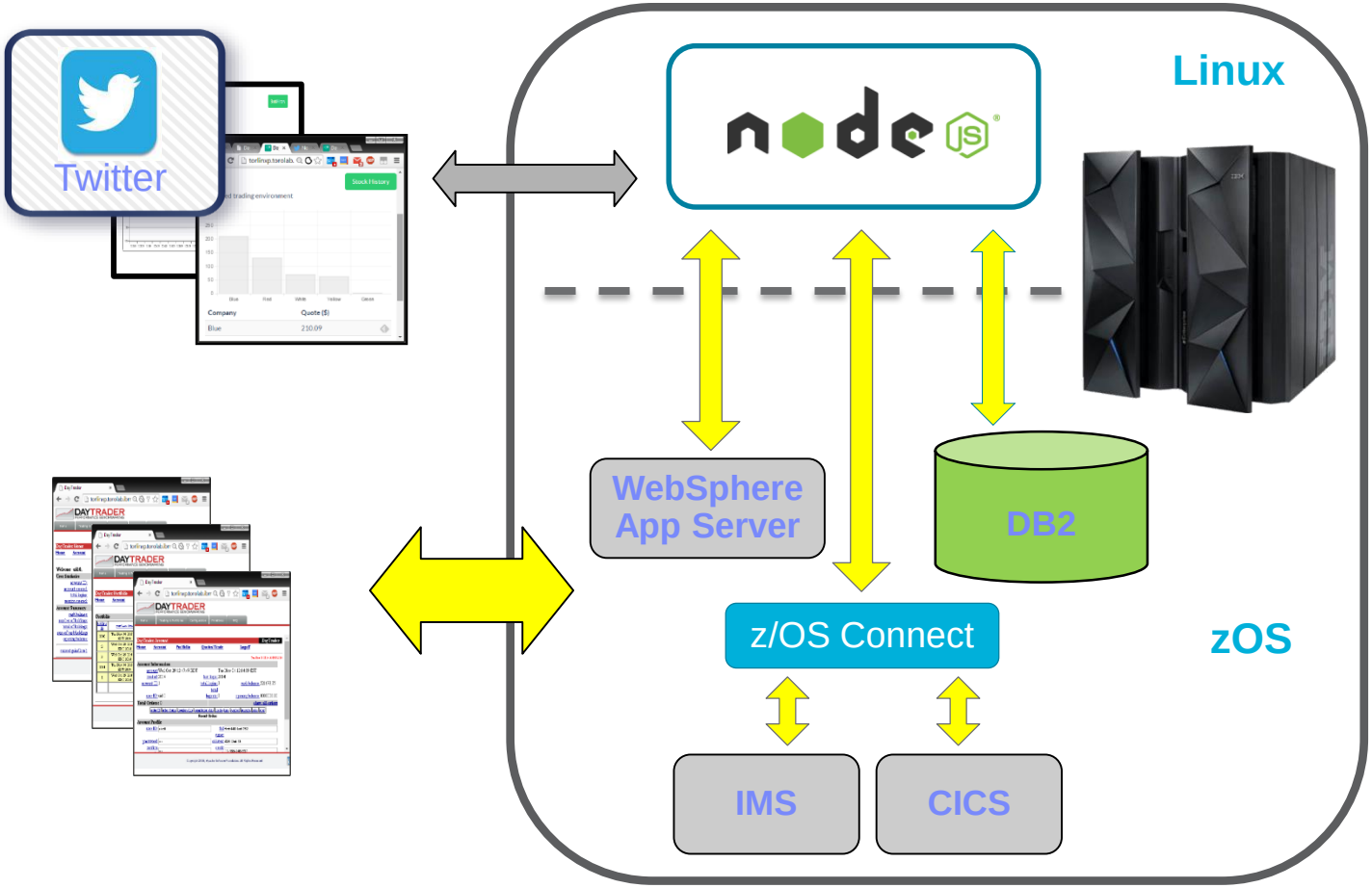
Common Adoption Patterns

- Skunk works project
- Internal APIs or sites
- Exposing existing data/systems
- Greenfields development

Locality of Data

- Moving data can be hard
  - Confidentiality
  - Regulation
  - Support for legacy data stores
  - Effort
- Locality of Data affects performance

Locality of Data – Example on z Systems



Co-locate **Node.js on z** vs x86

2.5x Better Throughput

60% Faster Response Time to **DB2 on z/OS**

Journey to Cloud Native - Potential Roadblocks

- Having to change Environment
  - Platforms/Operating systems
  - Data
- Support
- Monitoring and Problem Investigation
- Baseline Enterprise Requirements
  - Internationalization
  - Security
  - Quality/Stability
  - Performance

IBM in the Node.js Community

- 9 Collaborators, 2 CTC/TSC
- Active in many WGs
- Platinum Foundation Sponsor


Ben Noordhuis


Gibson Fahnestock


Michael Dawson


Sam Roberts


Michael Tunnicliffe


Steven Loomis


Ryan Graham


Bert Belder


Richard Lau

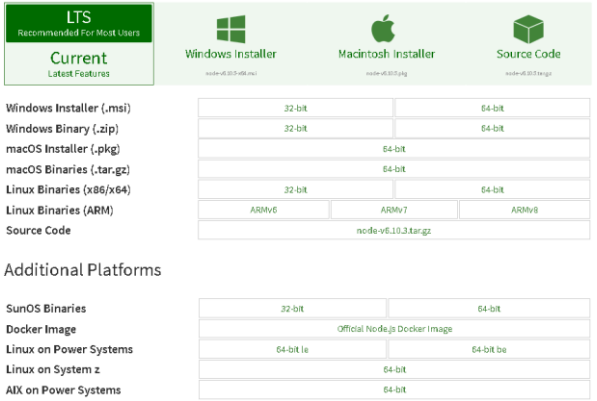
Environment: Platform Choice

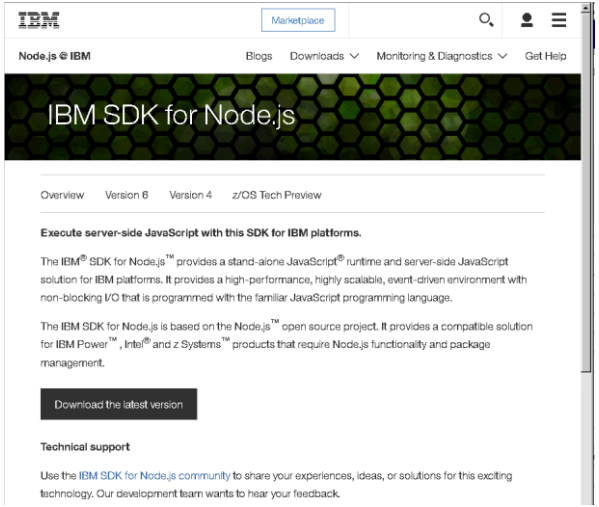
- Community
  - Binaries for **Linux on P** and **Z** and **AIX**

- IBM SDK for Node.js
  - Shipping Node.js releases since 2013
  - 0.10.x + 0.12.x + 4.x + 6.x
  - Linux on x / p / z, AIX, Windows, Mac

  - Working on support for z/OS

tech preview - https://developer.ibm.com/node/sdk/





Environment: Deployment Choice

- Develop for Deployment Independence
  - Plan to Leverage Infrastructure services
    - Load balancing
    - Scale in/out
    - Monitoring
  - 12 Factor Apps (https://12factor.net/)
- Bluemix
  - Cloud Foundry and Docker/Kubernetes deployments
    - Public
    - Dedicated
    - Local
  - OpenWhisk
- Tools

Deployment choice: Pattern Generators

Node.js @ IBM

Blogs Downloads Monitoring & Diagnostics Get Help

The code generators can be either invoked via the Bluemix CLI (Install steps) or via the Bluemix UI (Create Project). Both methods require the user to have a Bluemix account (Register).

1. Select a Pattern

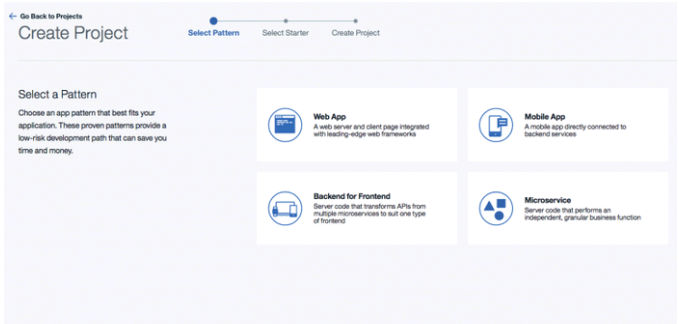
The code generators first ask a user to select their preferred design pattern out of the following:

1. Web App – A web server and client page integrated with leading-edge web frameworks
2. Mobile App – A mobile app directly connected to backend services
3. Backend for Frontend – Server code that transforms APIs from multiple microservices to suit one type of frontend
4. Microservice – Server code that performs an independent, granular business function

Bluemix CLI

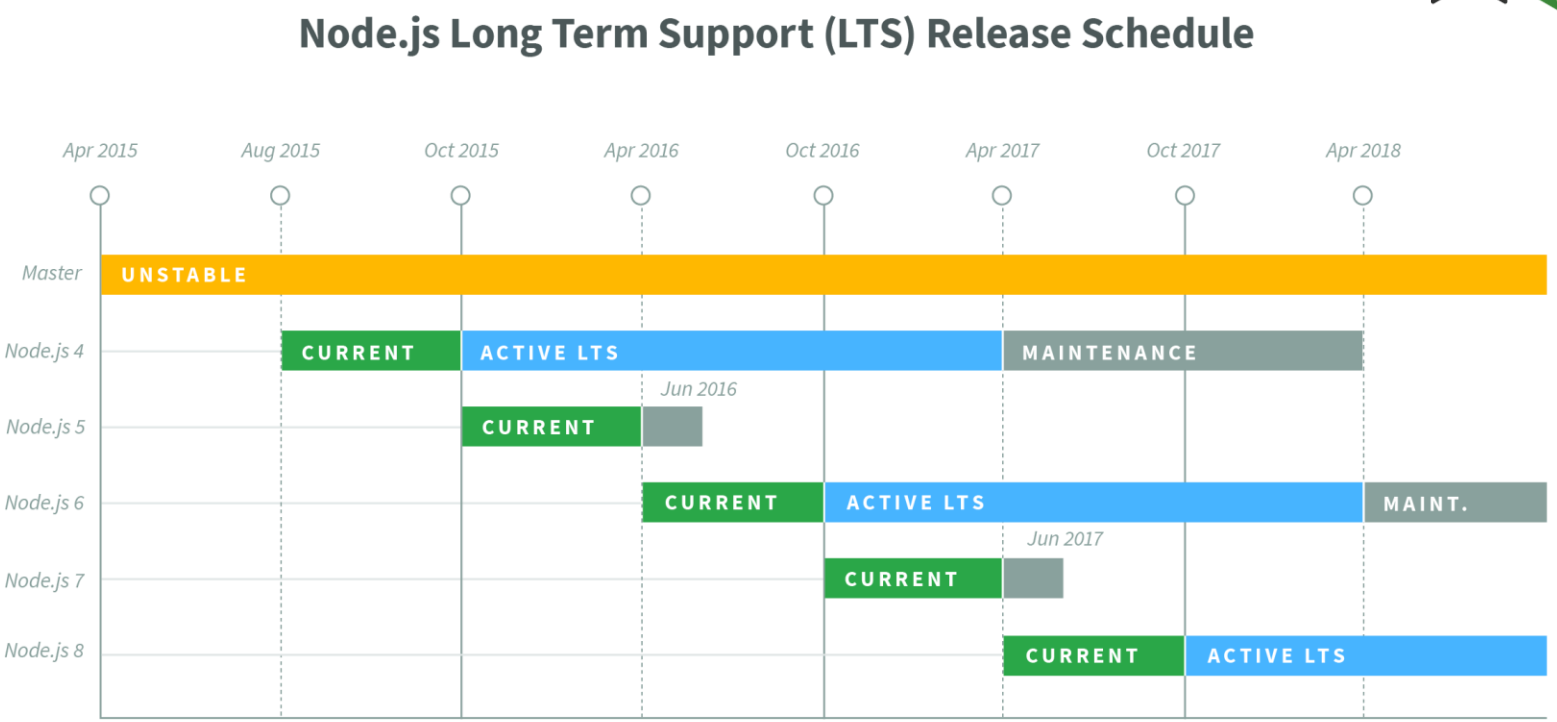
```
bethanys-mbp:~ beth$ bx dev create

dev version 0.0.6

See the documentation on Bluemix:
https://console.bluemix.net/docs/cloudnative/index.html

? Select a pattern:
1. Web App
2. Mobile App
3. Backend for Frontend
4. Microservice
Enter a number>
```

Bluemix UI



https://developer.ibm.com/node/2017/04/24/generate-node-js-apps-bluemix/

Support: LTS



https://github.com/nodejs/LTS

Problem Determination: Tooling

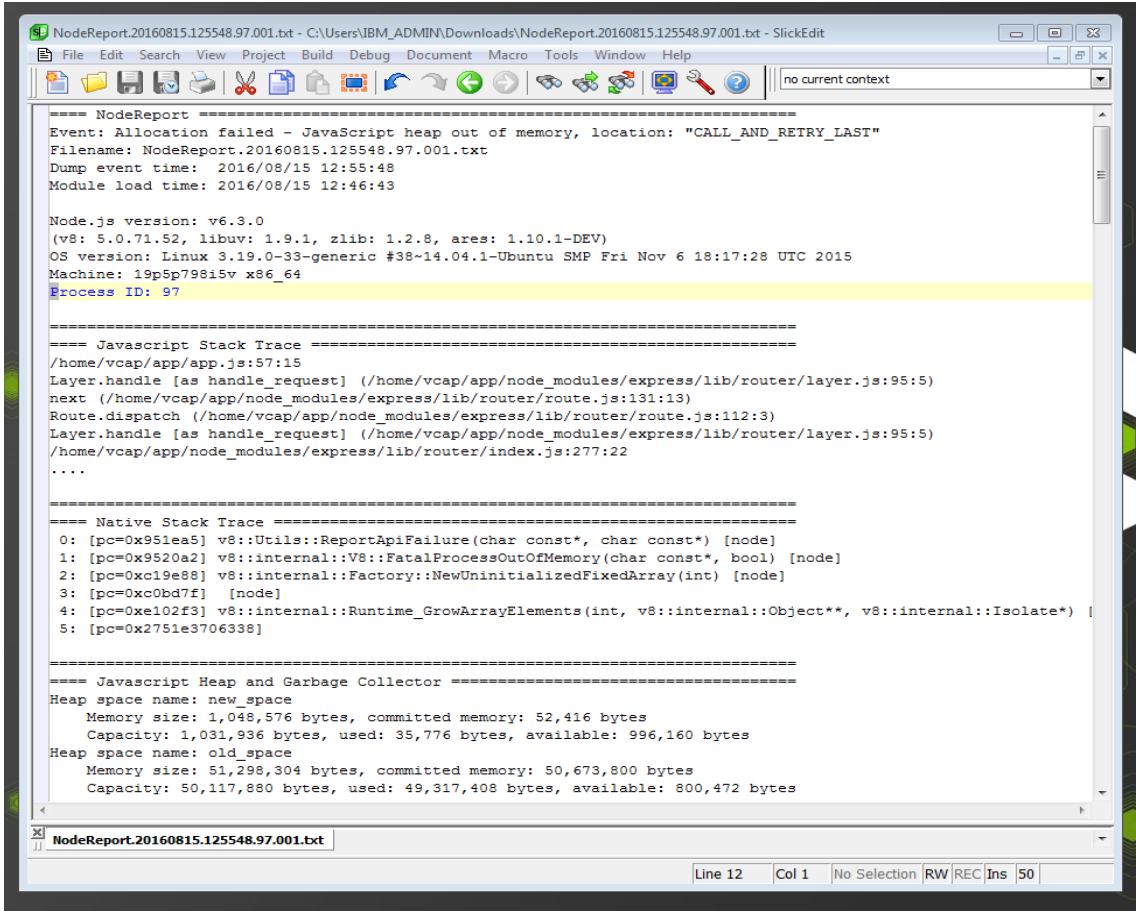
- NodeReport
- Heap Dump generation
- Core inspection - LLNODE

NodeReport

NodeReport content:

- Event summary
- Node.js and OS versions
- JavaScript stack trace
- Native stack trace
- Heap and GC statistics
- Resource usage
- libuv handle summary
- Environment variables
- OS ulimit settings

https://github.com/nodejs/nodereport

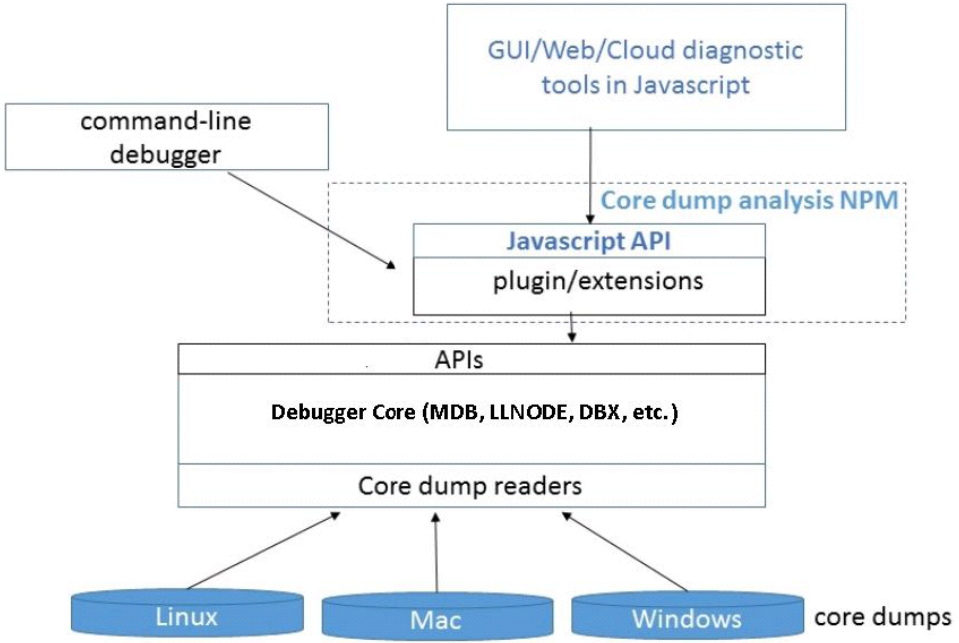


Heap Dump Generation

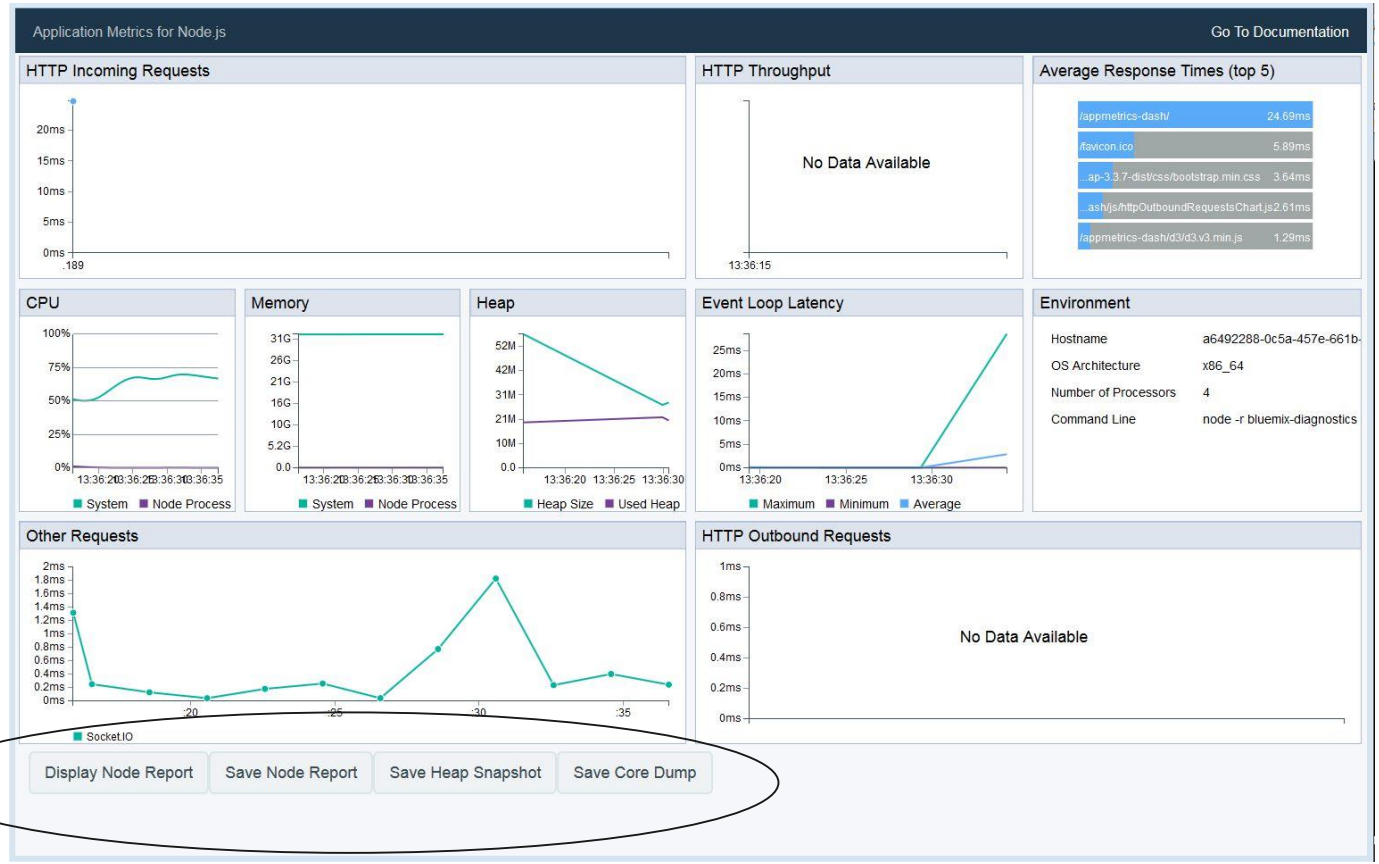
- Heapdump module – https://github.com/bnoordhuis/node-heapdump
- Chrome developer tools
- Limitations
  - Need to modify application
  - Slow to generate
  - O(N) memory usage
  - Limited content
  - Output is large

Core Inspection

- MDB
- IDDE
- Working in community to standardize on LLNODE



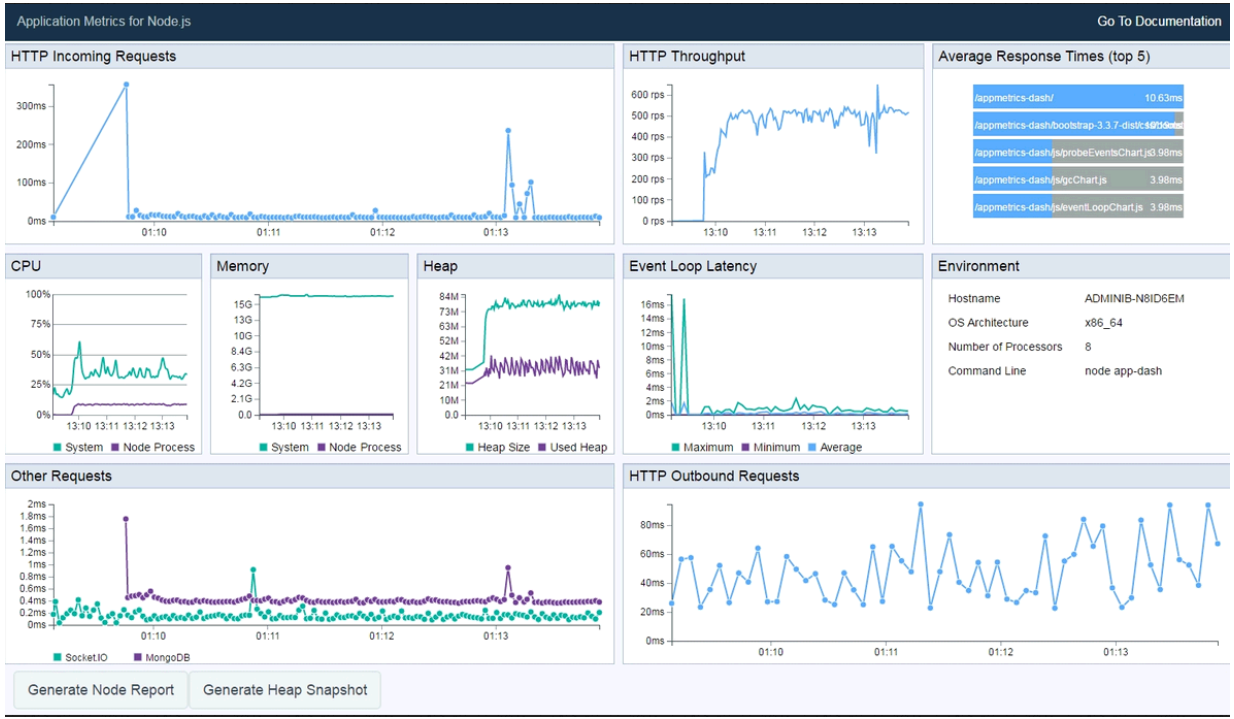
Problem Determination: In the Cloud



Monitoring: Tooling

- Appmetrics
- Appmetrics - Dash
- Health Center

Appmetrics-dash



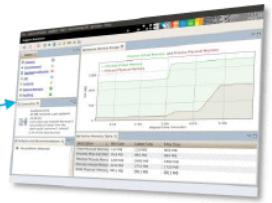

```
require('appmetrics-dash').monitor()
```

https://www.npmjs.com/package/appmetrics-dash

Node.js IBM – Appmetrics

https://www.npmjs.com/package/appmetrics

Node.js application monitored by appmetrics



```
monitor.on('gc', function(data) {
    gcMetrics.used += data.used;
    gcMetrics.count++;
});
```

appmetrics

NPM, Inc. (US) https://www.npmjs.com/package/appmetrics

Normally Palatable Mango

npm

find packages

appmetrics public

Node Application Metrics

Node Application Metrics monitoring and profiling agent

codacy A

Node Application Metrics instruments the Node.js runtime for performance monitoring, providing the monitoring data via an API. Additionally the data can be visualized in an Eclipse IDE using the IBM Monitoring and Diagnostics Tools - Health Center client.

See https://www.ibm.com/developerworks/java/jdk/tools/healthcenter/ for more details.

Node Application Metrics provides the following built-in data collection sources:

source	description
Environment	Machine and runtime environment information
CPU	Process and system CPU
Memory	Process and system memory usage
GC	Node/V8 garbage collection statistics
Event Loop	Event loop latency information
Express	Express 4.x Web Framework application request monitoring

Baseline ER:Quality and Stability

- Different release types
- Change flow processes
- Enhancement Proposal process
- Automation and Testing
  - Functional Tests
  - Module Testing (Citgm - https://github.com/nodejs/citgm/)
  - Stress Testing (Future)
  - Platform/OS coverage
  - Development Workflows (Future)
- Performance Benchmarks (http://benchmarking.nodejs.org)
- Tools (https://coverage.nodejs.org, ci jobs)

https://nodejs.org/en/blog/community/quality-with-speed/

Baseline ER: Application Stability – N-API

- Simplify native module maintenance
- Break dependency on v8
- Modules work across Node.js releases without re-compilation

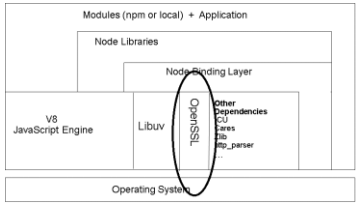
https://github.com/nodejs/abi-stable-node

Baseline ER: Internationalization

- Internationalization
  - Integration of ICU
  - Internal Messages

Baseline ER: Security

- Security
  - Security WG - https://github.com/nodejs/security-wg
- FIPs



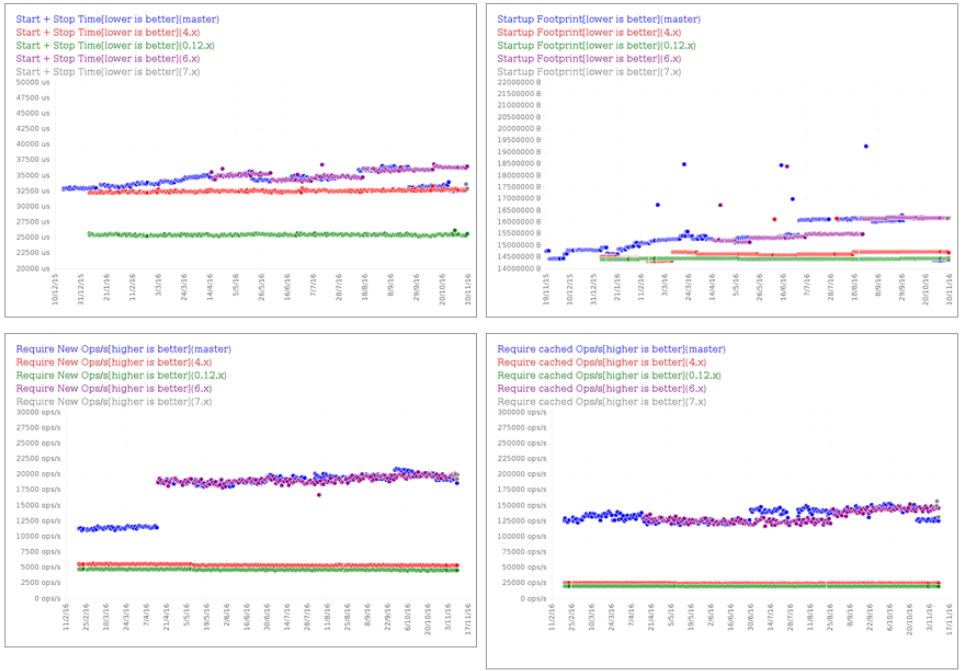
1747	**OpenSSL Software Foundation** 1829 Mount Ephraim Road Adamstown, MD 21710 USA Steve Marquess TEL: 877-673-6775 **CST Lab:** NVLAP 100432-0	**OpenSSL FIPS Object Module** (Software Versions: 2.0, 2.0.1, 2.0.2, 2.0.3, 2.0.4, 2.0.5, 2.0.6, 2.0.7, 2.0.8, 2.0.9 or 2.0.10) *(When built, installed, protected and initialized as assumed by the Crypto Officer role and as specified in the provided Security Policy. Appendix A of the provided Security Policy specifies the actual distribution tar file containing the source code of this module. There shall be no additions, deletions or alterations to the tar file contents as used during module build. The distribution tar file shall be verified as specified in Appendix A of the provided Security Policy. Installation and protection shall be completed as specified in Appendix A of the provided Security Policy. Initialization shall be invoked as per Section 4 of the provided Security Policy. Any deviation from specified verification, protection, installation and initialization procedures will result in a non FIPS 140-2 compliant module.)* **Validated to FIPS 140-2** **Consolidated Validation Certificate** **Security Policy**	Software	06/27/2012 07/09/2012 07/18/2012 10/24/2012 01/22/2013 02/06/2013 02/22/2013 02/28/2013 03/28/2013 05/16/2013 06/14/2013 08/16/2013 08/23/2013 11/08/2013 12/20/2013 06/27/2014 07/03/2014 09/02/2014 09/12/2014 10/16/2014 12/31/2014 06/15/2015 09/04/2015 01/25/2016	***Overall Level:* 1** -Roles, Services, and Authentication: Level 2 Design Assurance: Level 3 -Operational Environment: Tested as meeting Level 1 with Android 2.2 running on Qualcomm QSD8250 (ARMv7) without NEON (gcc Compiler Version 4.4.0) Android 2.2 running on Qualcomm QSD8250 (ARMv7) with NEON (gcc Compiler Version 4.4.0) Microsoft Windows 7 (32 bit) running on Intel Celeron (Microsoft 32 bit C/C++ Optimizing Compiler Version 16.00) uCLinux 0.9.29 running on ARM 922T (ARMv4) (gcc Compiler Version 4.2.1) Fedora 14 running on Intel Core i5 with PAA (gcc Compiler Version 4.5.1) HP-UX 11i (32 bit) running on Intel Itanium 2 (HP C/aC++ B3910B) HP-UX 11i (64 bit) running on Intel Itanium 2 (HP C/aC++ B3910B) Ubuntu 10.04 running on Intel Pentium T4200 (gcc Compiler Version 4.1.3)

node-test-commit-linux-fips	14 min - #3084	5 hr 48 min - #3071	14 min	

Baseline ER: Performance

https://benchmarking.nodejs.org/

- Benchmarking WG
  - Define Use Cases
  - Identify/Build Benchmarks
  - Run/Capture results

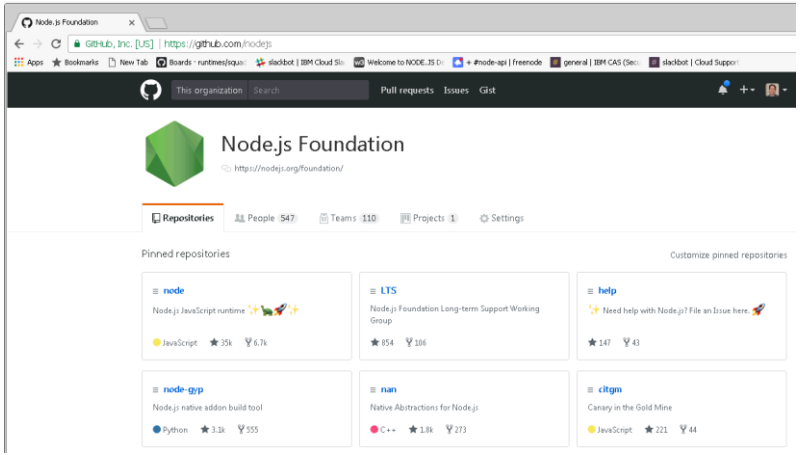


https://github.com/nodejs/benchmarking

Summary

- Node.js is a key runtime for Polyglot deployments
- Intro to typical journey to Cloud Native
- Support for existing environments is an enabler for Cloud Native
- Community work to reduce potential roadblocks
- Meet us in github to get involved

https://github.com/nodejs



Questions ?

Copyrights and Trademarks