Node.js Apps Import Integration

Treasure Data provides td-agent to collect server-side logs and events and to seamlessly import the data from Node.js applications.

The fluent-logger-node library enables Node.js applications to post records to their local td-agent.

td-agent, in turn, uploads the data to the cloud every 5 minutes. Because the daemon runs on a local node, the logging latency is negligible.

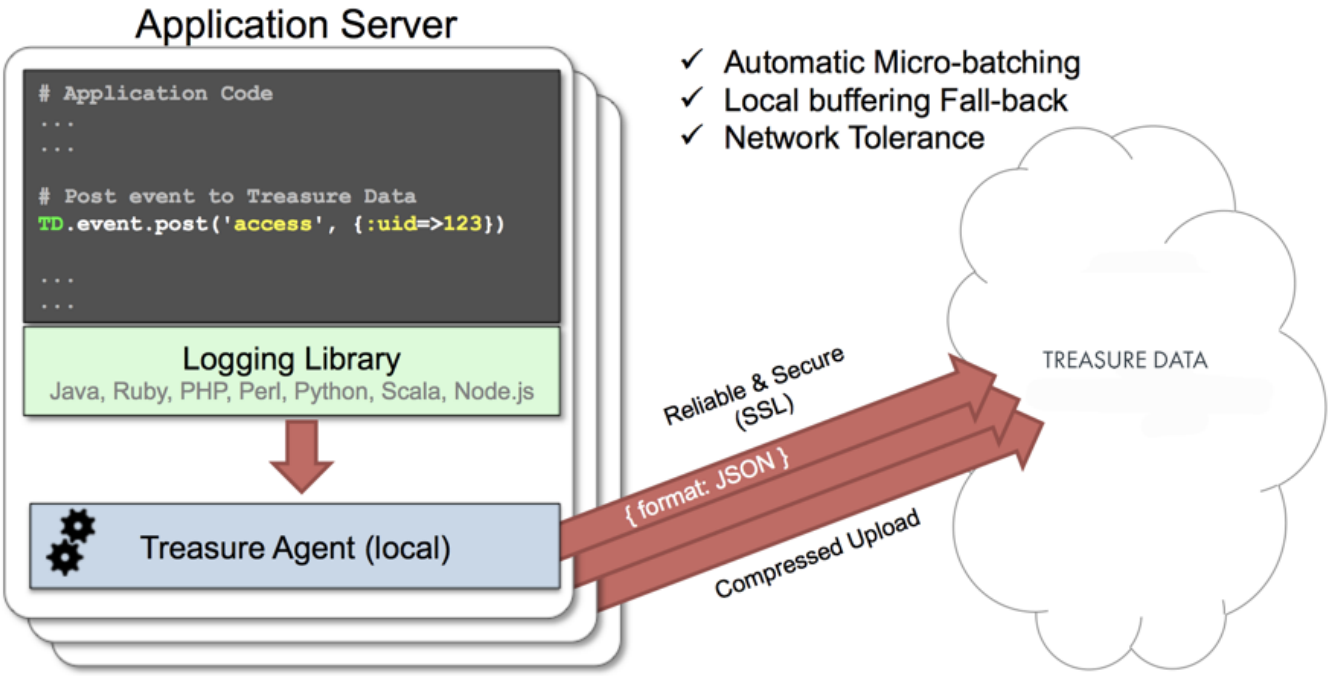
Continue to the following topics:

Prerequisites

- Basic knowledge of Node.js and NPM.
- Basic knowledge of Treasure Data, including the TD Toolbelt.
- Node.js 0.6 or higher (for local testing).

Installing td-agent

Install `td-agent`on your application servers. td-agent sits within your application servers, focusing on uploading application logs to the cloud.



The fluent-logger-node library enables Node.js applications to post records to their local td-agent. td-agent, in turn, uploads the data to the cloud every 5 minutes. Because the daemon runs on a local node, the logging latency is negligible.

td-agent Install Options

To install `td-agent`, run one of the following commands based on your environment. The agent program is installed automatically by using the package management software for each platform like rpm/deb/dmg.

RHEL/CentOS 5,6,7

```
$ curl -L https://toolbelt.treasuredata.com/sh/install-redhat-td-agent3.sh | sh
```

Ubuntu and Debian

```
# 18.04 Bionic
$ curl -L https://toolbelt.treasuredata.com/sh/install-ubuntu-bionic-td-agent3.sh | sh
# 16.04 Xenial (64bit only)
$ curl -L https://toolbelt.treasuredata.com/sh/install-ubuntu-xenial-td-agent3.sh | sh
```

Legacy support for EOL versions is still available

```
# 14.04 Trusty
$ curl -L https://toolbelt.treasuredata.com/sh/install-ubuntu-trusty-td-agent3.sh | sh
# 12.04 Precise
$ curl -L https://toolbelt.treasuredata.com/sh/install-ubuntu-precise-td-agent3.sh | sh
# Debian Stretch (64-bit only) $ curl -L https://toolbelt.treasuredata.com/sh/install-debian-stretch-td-agent3.
sh | sh
# Debian Jessie (64-bit only)
$ curl -L https://toolbelt.treasuredata.com/sh/install-debian-jessie-td-agent3.sh | sh
# Debian Squeeze (64-bit only)
$ curl -L https://toolbelt.treasuredata.com/sh/install-debian-squeeze-td-agent2.sh | sh
```

Amazon Linux

You can choose Amazon Linux 1 or Amazon Linux 2. Refer to Installing td-agent on Amazon Linux.

MacOS X 10.11+

```
$ open 'https://td-agent-package-browser.herokuapp.com/3/macosx/td-agent-3.1.1-0.dmg'
```

MacOS X 10.11.1 (El Capitan) introduced some security changes. After the td-agent is installed, edit the **/Library/LaunchDaemons/td-agent.plist** file to change **/usr/sbin/td-agent** to **/opt/td-agent/usr/sbin/td-agent**.

Windows Server 2012+

The Windows installation requires the steps detailed in:

- Installing Treasure Agent using .msi Installer (Windows)

Opscode Chef Repository

```
$ echo 'cookbook "td-agent"' >> Berksfile
$ berks install
```

AWS Elastic Beanstalk is also supported. Windows is currently NOT supported.

Modifying /etc/td-agent/td-agent.conf

Specify your API key by setting the `apikey` option. You can view your API key from your profile in TD Console. Set the `apikey`option in your td-agent.conf file.

```
# Treasure Data Input and Output
<source>
  type forward
  port 24224
</source>
<match td.*.*>
  type tdlog
  endpoint api.treasuredata.com
  apikey YOUR_API_KEY
  auto_create_table
  buffer_type file
  buffer_path /var/log/td-agent/buffer/td
  use_ssl true
</match>
```

YOUR_API_KEY should be your actual API key string. Using a [write-only API key](access-control#rest-apis-access) is recommended.

Restart your agent when the following lines are in place.

```
# Linux
$ sudo /etc/init.d/td-agent restart

# MacOS X
$ sudo launchctl unload /Library/LaunchDaemons/td-agent.plist
$ sudo launchctl load /Library/LaunchDaemons/td-agent.plist
```

td-agent now accepts data via port 24224, buffers the data (*var/log/td-agent/buffer/td*), and automatically uploads the data into the cloud.

Using fluent-logger-node

Obtaining the Most Recent Version

Obtain the most recent version of fluent-logger-node.

A Sample Application

A sample Express app using fluent-logger-node is as follows:

package.json

```
{
  "name": "node-example",
  "version": "0.0.1",
  "dependencies": {
    "express": "2.5.9",
    "fluent-logger": "0.1.0"
  }
}
```

Now use *npm* to install your dependencies locally:

```
$ npm install
fluent-logger@0.1.0 ./node_modules/fluent-logger
express@2.5.9 ./node_modules/express
|-- qs@0.4.2
|-- mime@1.2.4
|-- mkdirp@0.3.0
|-- connect@1.8.6 (formidable@1.0.9)
```

web.js

This is the simplest web app.

```
var express = require('express');
var app = express.createServer(express.logger());

var logger = require('fluent-logger');
logger.configure('td.test_db', {host: 'localhost', port: 24224});

app.get('/', function(request, response) {
  logger.emit('follow', {from: 'userA', to: 'userB'});
  response.send('Hello World!');
});
var port = process.env.PORT || 3000;
app.listen(port, function() {
  console.log("Listening on " + port);
});
```

Confirming Data Import

Execute the app and go to `http://localhost:3000/` in your browser.

```
$ node web.js
```

A SIGUSR1 signal is sent and flushes td-agent's buffer. Upload starts immediately.

```
# Linux
$ kill -USR1 `cat /var/run/td-agent/td-agent.pid`

# MacOS X
$ sudo kill -USR1 `sudo launchctl list | grep td-agent | cut -f 1`
```

To confirm that your data has been uploaded successfully, issue the *td tables* command as follows:

```
$ td tables
+------------+------------+------+-----------+
| Database   | Table      | Type | Count     |
+------------+------------+------+-----------+
| test_db    | follow     | log  | 1         |
+------------+------------+------+-----------+
```

The first argument of post() determines the database name and table name. If you specify `td.test_db.test_table`, the data is imported into the table *test_table* within the database *test_db*. They are automatically created at upload time.

Production Deployments

High-Availability Configurations of td-agent

For high-traffic websites (more than 5 application nodes), use a high availability configuration of td-agent to improve data transfer reliability and query performance.

- High-Availability Configurations of td-agent

Monitoring td-agent

Monitoring td-agent itself is also important. Refer to the following document for general monitoring methods for td-agent.

- Monitoring td-agent

td-agent is fully open-sourced under the Fluentd project.

Next Steps

We offer a schema mechanism that is more flexible than that of traditional RDBMSs. For queries, we leverage the Hive and Presto Query Languages.

- Schema Management
- Presto Query Language
- Hive Query Language
- Programmatic Access with REST API and its Bindings