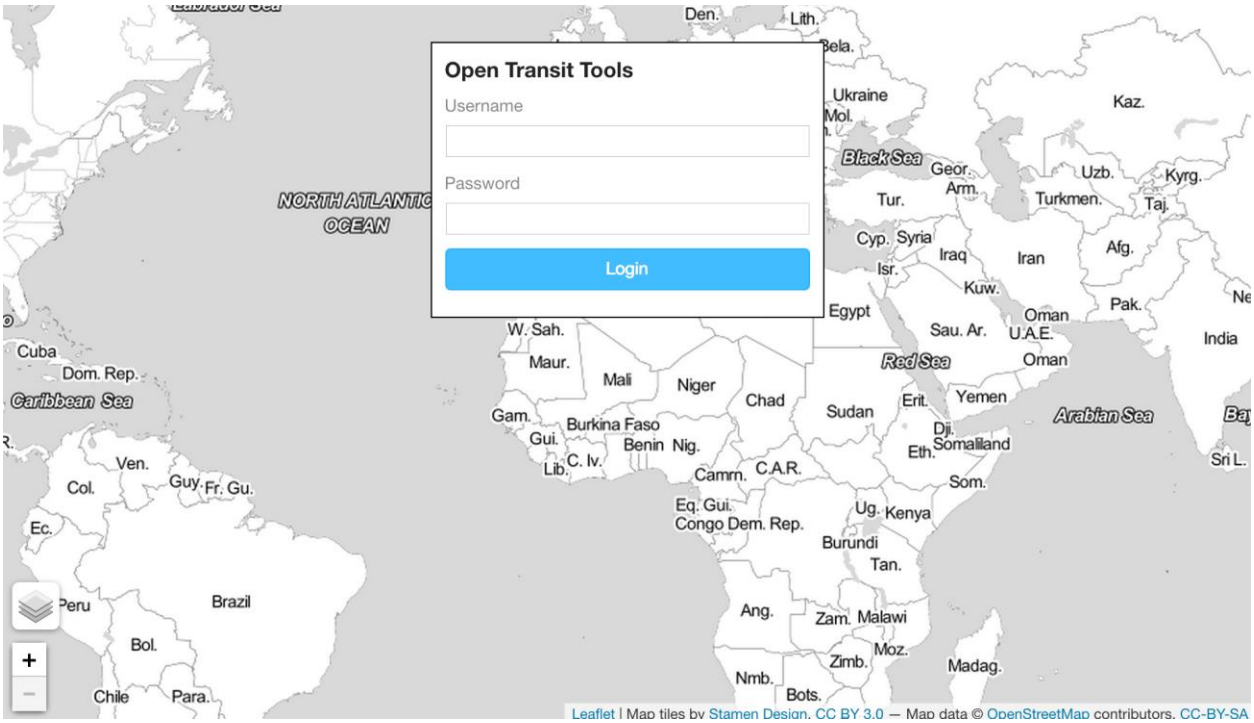
Open Transit Indicators

Server Administrators' Guide



Prepared for:

WORLD BANK GROUP

1818 H Street, NW
Washington, DC 20433

Funding support provided by:

Australian AID

Report developed by:

azavea

340 N 12th St, Suite 402
Philadelphia, PA 19107
(215) 925-2600
http://www.azavea.com

Table of Contents

Open Transit Indicators
System Administrators' Guide

Introduction

This manual is intended for the system administrator responsible for installing and maintaining the Open Transit Indicators application. Users of the application may refer to the End Users' Manual.

System Requirements

The Open Transit Indicators application runs on AMD 64 bit Ubuntu 12.04. The server should have the following minimum resources:

- 8GB RAM
- 2 core CPU
- 24GB Storage

Additional software may be required on the administrator's client machine. This guide assumes familiarity with SSH and administering Linux OS servers. Windows users may require PuTTy or similar software in order to SSH into the application Server.

Installation times will vary depending on local internet speed. There are several large code libraries upon which the application depends. With a high-speed connection, the installation process takes approximately 30 minutes.

Installation

There are multiple options for deploying the application.

For testing purposes, installing a virtual machine (VM) using Vagrant (https://www.vagrantup.com) is a fast and simple way to run the application. Note that the machine is configured to use 8GB of RAM, so the test VM should have at least that much allocated.

If installing on a webserver directly, a provisioning shell script is included that will gather and install all of the dependencies, set various configurations of the server software, and launch several processes so that the app is ready to run when provisioning is complete.

If deployment to Amazon EC2 is desirable, there is a configuration template that makes building and deploying an Amazon Machine Image (AMI) using Packer (http://www.packer.io/) very straightforward.

Standard Server Installation (Preferred)

Step 1: Server Base Configuration and Software Installation

SSH into the server that has had Ubuntu 12.04 (AMD64) freshly installed.

First, update the installation and install Git and your preferred text editor (Vim, Nano, etc.).

```
sudo apt-get update
sudo apt-get -y install git
```

Step 2: Clone Application Code

```
vagrant@precise32:~$ mkdir oti
vagrant@precise32:~$
vagrant@precise32:~$ ls
oti  postinstall.sh
vagrant@precise32:~$ cd oti
vagrant@precise32:~/oti$
vagrant@precise32:~/oti$ git clone -b master https://github.com/WorldBank-Transport/open-transit-indicators.git
```

Create a directory in which to install the application. This could be /projects, ~/app, or anything else – this location will be set in the deployment script later. In the example, the application will be installed to the ~/oti directory. Change directories into the install location and clone the git repository.

```
Cloning into 'open-transit-indicators'...
remote: Counting objects: 15336, done.
remote: Compressing objects: 100% (68/68), done.
remote: Total 15336 (delta 42), reused 18 (delta 7)
Receiving objects: 100% (15336/15336), 5.77 MiB | 2.14 MiB/s, done.
Resolving deltas: 100% (8553/8553), done.
vagrant@precise32:~/oti$
```

The project code will be pulled down from GitHub and cloned into the application directory in the server. In this case, it will be in ~/oti/open-transit-indicators as shown in the screenshot below.

```
vagrant@precise32:~/oti/open-transit-indicators$ pwd
/home/vagrant/oti/open-transit-indicators
vagrant@precise32:~/oti/open-transit-indicators$
```

Double-check the location of the installation using the command 'pwd'. This location will be used in the next step.

Step 3: Provision Script Customization

```
vagrant@precise32:~/oti/open-transit-indicators$ ls
deployment  js  LICENSE  python  README.md  scala  util  Vagrantfile
vagrant@precise32:~/oti/open-transit-indicators$ vim deployment/provision.sh
```

Before installing the software on the server, modify the deployment/provision.sh script to reflect the location of the codebase using your preferred text editor.

```
#!/bin/bash
PROJECTS_DIR="/projects"

# Set the path to the project directory; all other paths will be relative to this.
PROJECT_ROOT="$PROJECTS_DIR/open-transit-indicators"
```

The 2nd line of the script will look like the above screenshot initially.

```
#!/bin/bash
PROJECTS_DIR="/home/vagrant/oti"

# Set the path to the project directory; all other paths will be relative to this.
PROJECT_ROOT="$PROJECTS_DIR/open-transit-indicators"
```

Modify the PROJECTS_DIR value to match the directory that contains /open-transit-indicators.

Step 4: Provisioning the Server

```
vagrant@precise32:~/oti/open-transit-indicators$
vagrant@precise32:~/oti/open-transit-indicators$ sudo ./deployment/provision.sh production
```

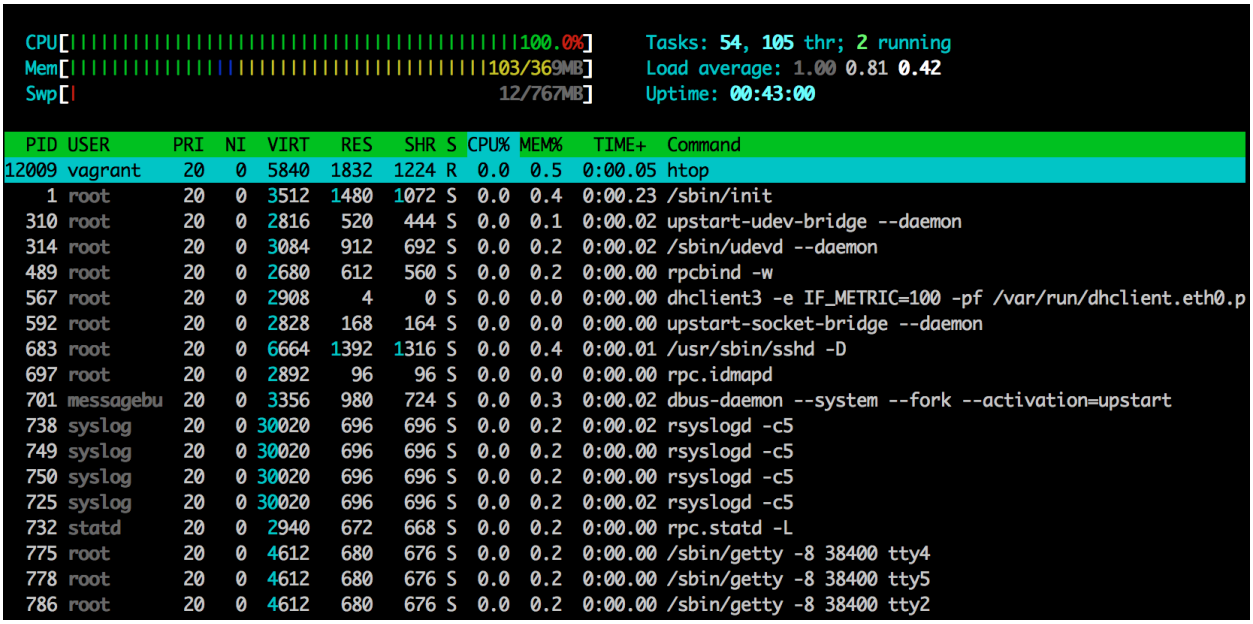
After modifying the provisioning script, run it using the above command with the production modifier.

The provisioning script will automatically configure the server with software and settings required to run the application. On a high-speed connection, this process will take approximately 30 minutes. If errors are encountered, it may be due to a repository being down or to a network connection issue. Starting over after a failure will skip the parts that were completed and continue from that point.

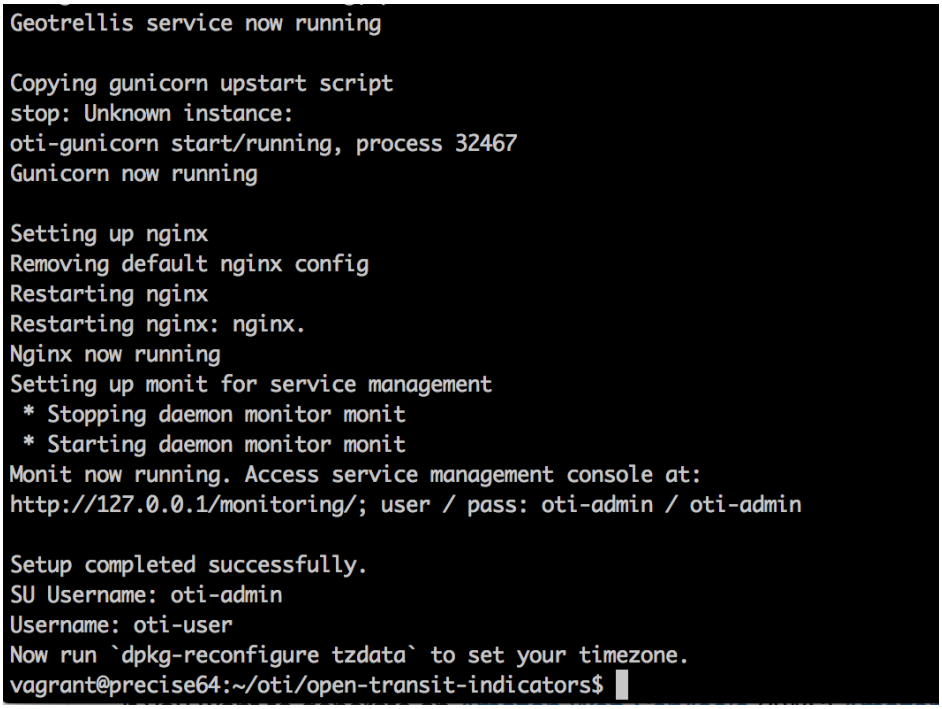
Make sure the system is configured with the appropriate memory and compute resources to ensure a successful installation.

```
Welcome to Ubuntu 12.04 LTS (GNU/Linux 3.2.0-23-generic-pae i686)

 * Documentation:  https://help.ubuntu.com/
New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Welcome to your Vagrant-built virtual machine.
Last login: Fri Dec 12 14:25:50 2014 from 10.0.2.2
vagrant@precise32:~$ htop
vagrant@precise32:~$
```

Watching the system processes with 'htop' will show that the installation is working. It should take nearly 100% of the CPU while installing, unless it is waiting for files to download.



HTOP displays resource usage during installation.



When the provisioning script is complete, it will display success messages as well as the default administrator user name and password. There is also a message recommending time zone configuration for the server. This is an important configuration to support scenario design.

Step 5: Setting the Server Timezone

```
vagrant@precise64:~/oti/open-transit-indicators$ sudo dpkg-reconfigure tzdata
```

With 'sudo', run the command noted at the end of the provisioning script.

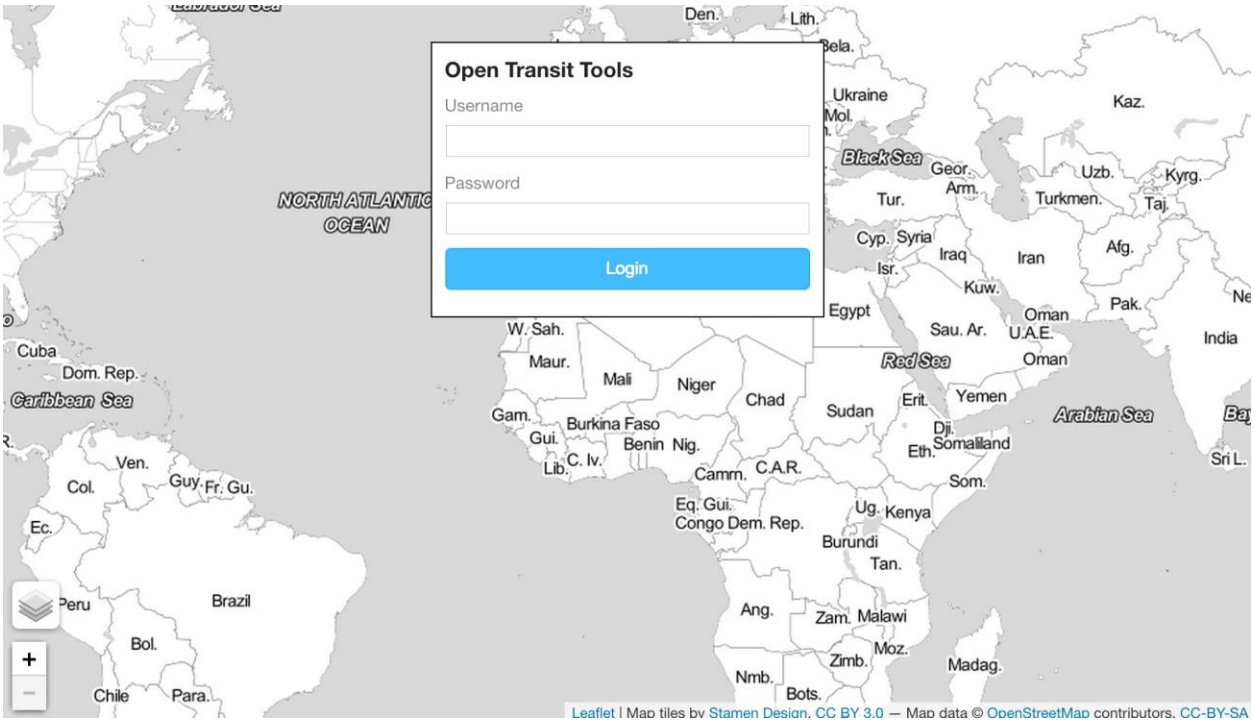
Choose the time zone to match the users' browser settings. For example, if the server is in Seattle, but the users are in New York, set the time zone to New York.

```
vagrant@precise64:~/oti/open-transit-indicators$ sudo dpkg-reconfigure tzdata

Current default time zone: 'America/New_York'
Local time is now:      Fri Dec 12 13:23:22 EST 2014.
Universal Time is now:  Fri Dec 12 18:23:22 UTC 2014.

vagrant@precise64:~/oti/open-transit-indicators$
```

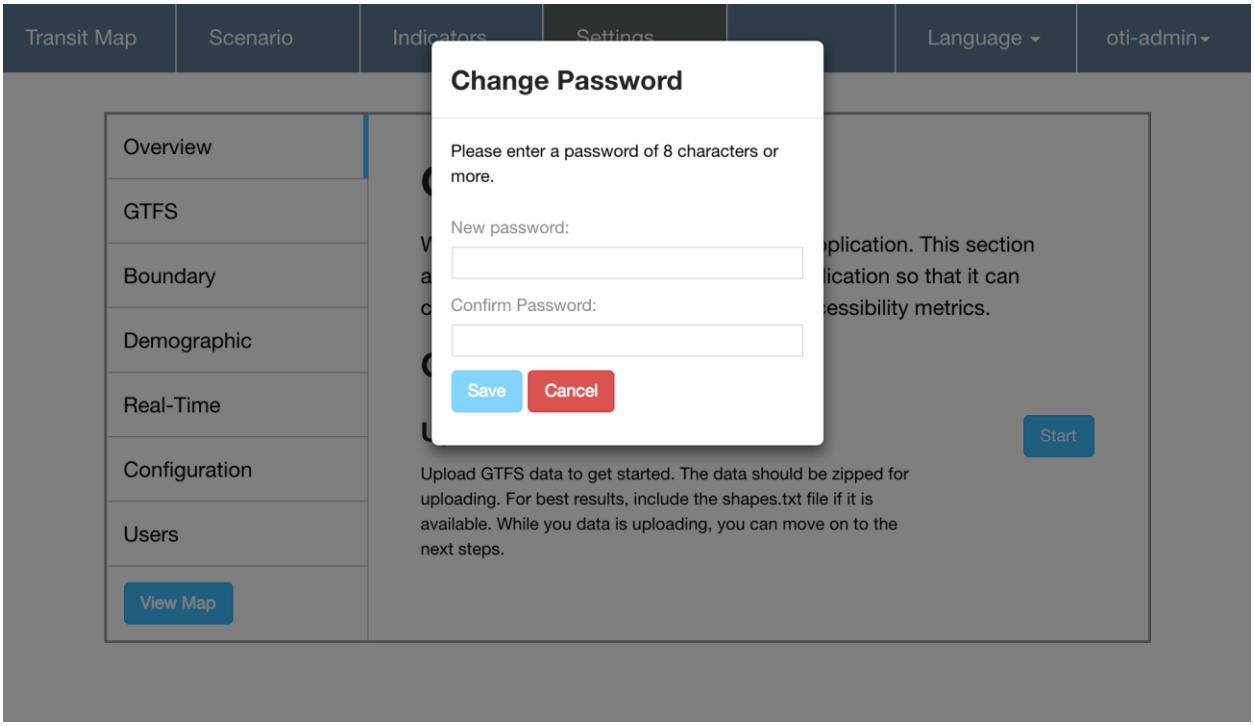
When complete, the server will confirm the time zone setting has been changed.

Step 6: Logging In

The installation process is complete. Navigate in the browser to the IP address or URL of the webserver. If the website does not show up, please check the security settings of the server to allow access from http port.



Log in with the default credentials. Click on the 'oti-admin' button at the top right after logging in, and change the password.



After changing the password, you are now ready to configure the software with your city's GTFS and GIS data.

Vagrant Test Machine

If you're using Vagrant, then installation is as simple as cloning the repo and then issuing vagrant up from the root of the repository directory.

Vagrant is a virtual machine integration tool that provides an integrated development environment on a local development machine. To install Vagrant with VirtualBox (both software packages are free), follow the instructions here: https://docs.vagrantup.com/v2/getting-started/index.html

After installing Vagrant, clone the repository from GitHub (https://github.com/WorldBank-Transport/open-transit-indicators) to a directory, navigate to it, and issue the command 'vagrant up'.

```
juniata:~ jbranigan$ cd git
juniata:git jbranigan$ mkdir oti
juniata:git jbranigan$ cd oti
juniata:oti jbranigan$ git clone https://github.com/WorldBank-Transport/open-transit-indicators.git
Cloning into 'open-transit-indicators'...
remote: Counting objects: 15668, done.
remote: Compressing objects: 100% (244/244), done.
remote: Total 15668 (delta 111), reused 0 (delta 0)
Receiving objects: 100% (15668/15668), 5.86 MiB | 9.55 MiB/s, done.
Resolving deltas: 100% (8693/8693), done.
Checking connectivity... done.
juniata:oti jbranigan$
juniata:oti jbranigan$
juniata:oti jbranigan$ vagrant up
```

Vagrant will download the appropriate Ubuntu machine image and then provision the machine with all necessary dependencies, just like the provision.sh script in the full server installation. The provision command will automatically configure the server with software and settings required to run the application. On a high-speed connection, this process will take approximately 30 minutes. If errors are encountered, it may be due to a repository being down or to a network connection issue. Starting over after a failure will skip the parts that were completed and continue from that point. Once the provisioning is complete, the application will be available at http://localhost:8067 in the browser.

The default amount of memory that Vagrant will allocate to the VM is 8GB (8192MB). To change the default amount of memory allocated to the vagrant machine, modify the Vagrantfile in the project root directory. Set the environment variable OTI_VAGRANT_MEMORY on line 9 to the preferred size, in MB.

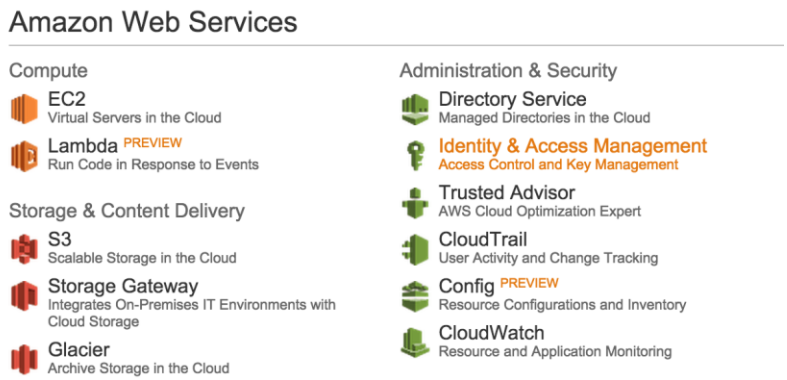
```
 7 if ENV['OTI_VAGRANT_MEMORY'].nil?
 8     # Project requirements require a machine w/ <= 8Gb
 9     MEMORY_MB = "8192"
10 else
11     MEMORY_MB = ENV['OTI_VAGRANT_MEMORY']
12 end
```

Deployment to Amazon EC2

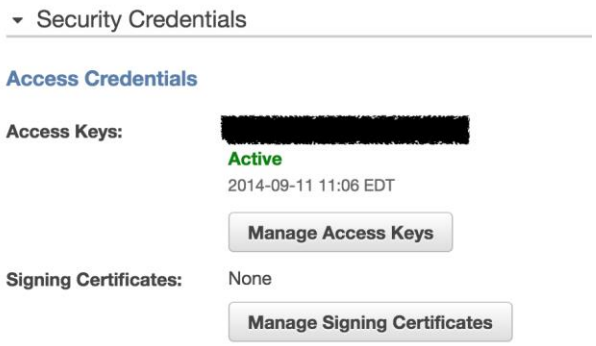
To generate an Amazon Web Services (AWS) Amazon Machine Image (AMI), it is possible to use Packer to handle the provisioning. AWS provides a variety of instance types that regularly get updated and enhanced. The instance type chosen to run this application should abide by the minimum server requirements mentioned above (8GB RAM, 2 core CPU, 24GB Storage). Packer is an open source tool that builds machine images from a source configuration file.

Step 1: Gather AWS Credentials

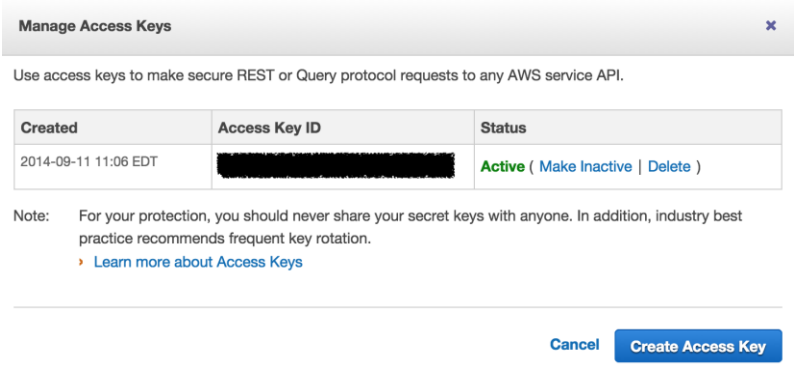
Create an AWS account if you do not have one already at http://aws.amazon.com/. Once logged in to the console, follow the Identity & Access Management link.



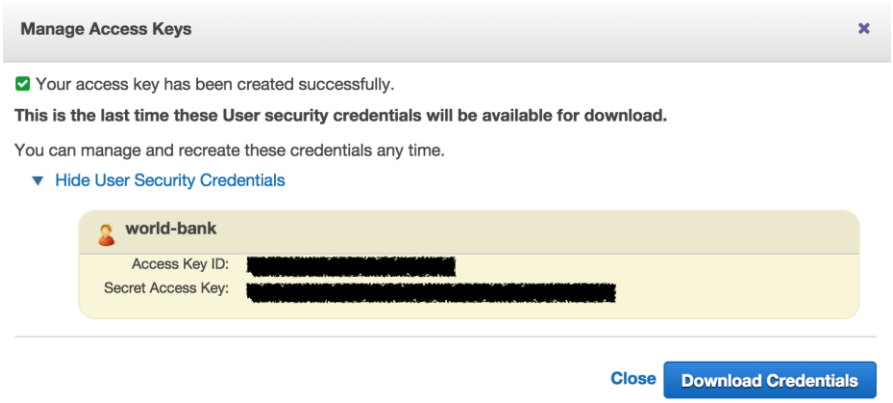
Scroll down to the Security Credentials section and click on Manage Access Keys.



Click on the Create Access Key button.



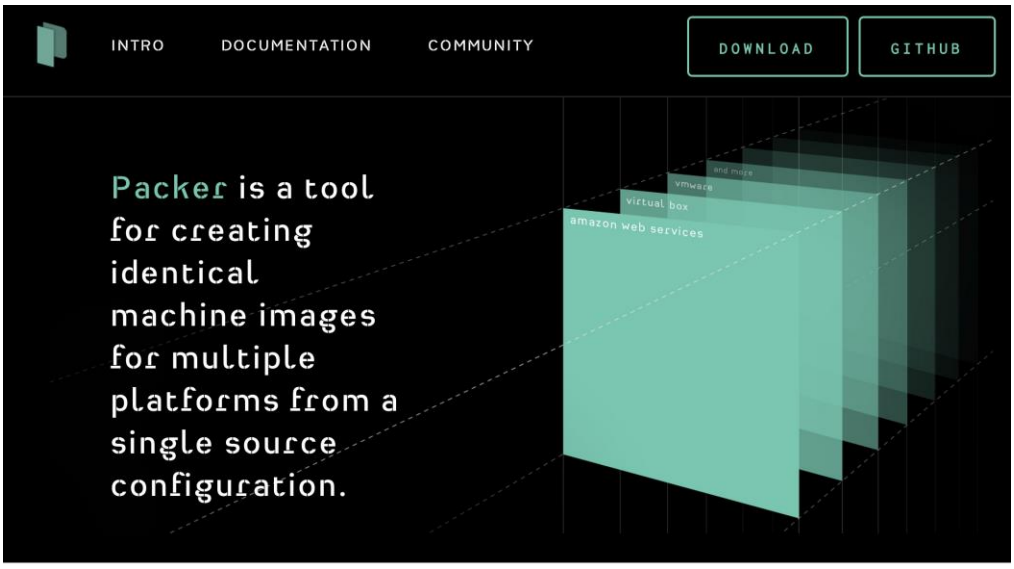
A new set of access keys will be generated. Download or copy the keys, as this will be the only time the secret key is exposed.



Step 2: Install Packer

Download and install Packer on your local machine (https://www.packer.io). The installation process requires several steps, and is different depending on your local operating system (Linux system is preferred, Mac system may cause access problems after building the AMI and AWS instance).

Detailed instructions can be found at https://www.packer.io/intro/getting-started/setup.html



Step 3: Configuration

Clone the repository to a local directory and navigate to the deployment/packer/ directory.

```
juniata:open-transit-indicators jbranigan$ cd deployment/packer/
juniata:packer jbranigan$ ls
open-transit-indicators.json    open-transit-vars.json.example
juniata:packer jbranigan$
```

Copy the open-transit-vars.json.example file and save it as open-transit-vars.json.

```
juniata:packer jbranigan$ cp open-transit-vars.json.example open-transit-vars.json
juniata:packer jbranigan$ ls
open-transit-indicators.json    open-transit-vars.json          open-transit-vars.json.example
juniata:packer jbranigan$
```

Edit open-transit-vars.json with the API keys downloaded in step 1.

```
1 {
2   "aws_access_key": "<FILL IN YOUR KEY HERE>",
3   "aws_secret_key": "<FILL IN YOUR KEY HERE>",
4 }
```

Step 4: Building the AMI

While in the deployment/packer/ directory, run the following command to generate a new AMI:

```
packer build -var-file=open-transit-vars.json open-transit-indicators.json
```

PLEASE NOTE: Running this command will cause resources to be created in AWS and will cost money. The installation process could take up to an hour, depending on internet connection speed. Once the process finishes installing, make note of the AMI ID. The completed script will display the AMI ID.

Troubleshooting 1: AMI ID. The above command is dependent upon an AMI resource provided by Ubuntu. As Ubuntu releases security updates, the current AMI may be replaced with one with a different ID. If an error is encountered where the software cannot find the AMI ID ami-daaed0b2, a new AMI ID can be found at http://cloud-images.ubuntu.com/locator/ec2/. In the search box on that page, enter "12.04 amd64 ebs-ssd" to narrow the results. Choose an AMI in your preferred AWS Availability Zone (us-east, cn-north, etc.) that has an instance type of ebs-ssd, and copy the AMI ID.

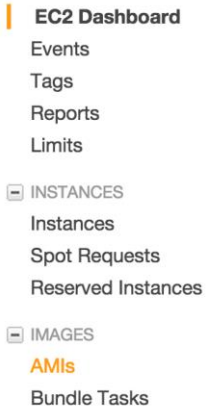
Edit the open-transit-indicators.json file and update the "ubuntu-ami" value on line 6 with the new Ubuntu AMI ID.

```
1 {
2   "variables": {
3     "aws_access_key": "",
4     "aws_secret_key": "",
5     "aws_region": "us-east-1",
6     "ubuntu_ami": "ami-daaed0b2",
7     "oti_branch": "master",
8     "instance_type": "m1.medium"
9   },
```

Troubleshooting 2: NPM error. When building the AMI, NPM error may occur during the process with npm error message indicating the process is not completed. This may due to a random failure of NPM. If an NPM error is encountered, try again the AMI building from the beginning.

Step 5: Launch an EC2 instance with the AMI

Log in to the AWS console and click on the AMIs link in the left sidebar.



Identify and select the AMI that was generated using Packer (it will have the same AMI ID displayed at the completion of the Packer process). Then click the Launch button.

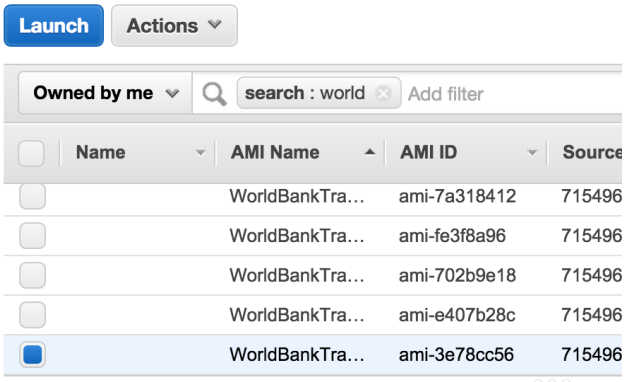
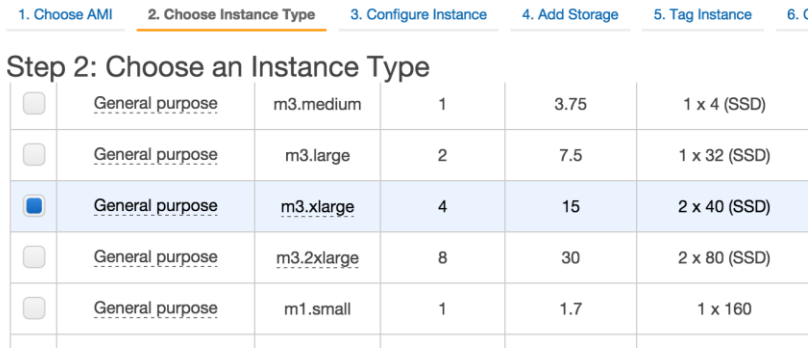


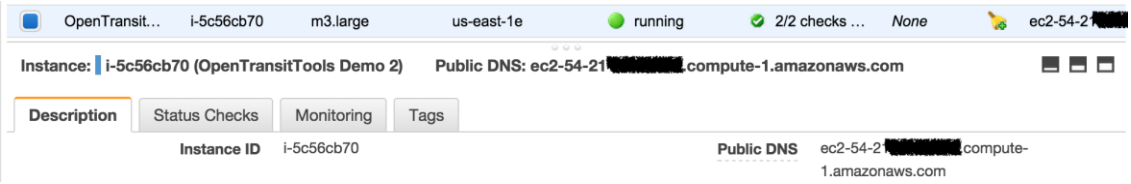
Image: ami-3e78cc56

The process of launching an AMI will walk through choosing the instance type, configuration, and other options. More information about this process is available on the AWS website: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instances-and-amis.html



Launch the AMI using the AWS EC2 management console and browse to the public DNS host name provided by Amazon. This process may be repeated to launch multiple instances from a single AMI.

Once the EC2 instance is configured and launched, the EC2 Management Console will list it as a running instance and will provide the public DNS host name.

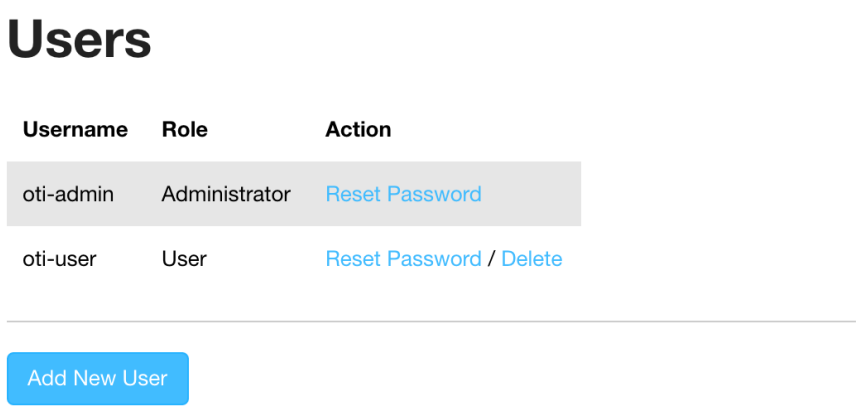


If the website does not show up by accessing the public DNS address, check the AWS instance security group rules to allow http access. More information about security group settings is available on the AWS website:
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-network-security.html#adding-security-group-rule

Application Configuration

Please refer to the End Users' Guide for details on application configuration.

User Management

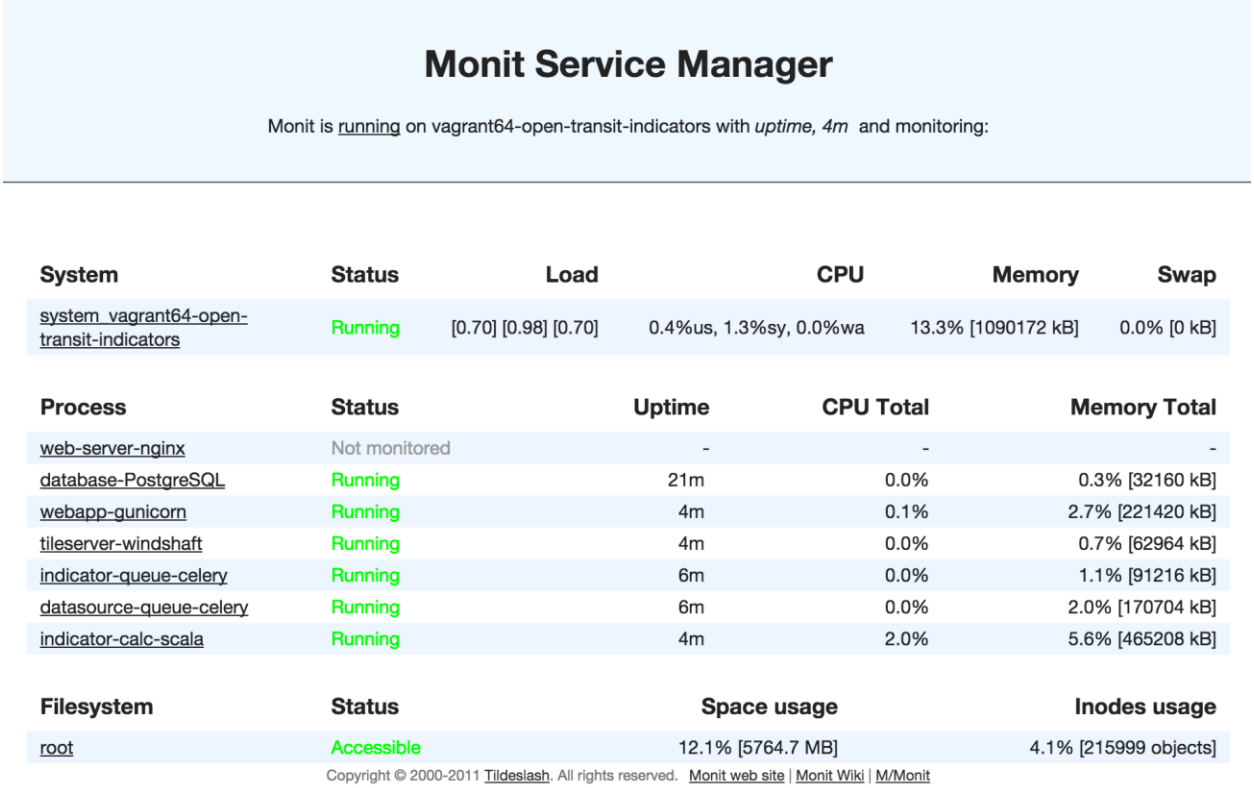


Admin users can add, delete, and manage users of the application. The system does not include email functionality, so an administrator would distribute temporary passwords to other users on account creation or password reset. There are two roles users can have:

- Administrator Capabilities
  - Configure the application datasets
  - Manage users
  - Create and view scenarios
- User Capabilities
  - View base indicator results
  - Create and view scenarios

System Monitoring

There is a tool available for administrators to monitor various services used by the application. It can be reached at http://<host-or-IP-or-machine>/monitoring.



Monit Service Manager

Monit is running on vagrant64-open-transit-indicators with *uptime, 4m* and monitoring:

System	Status	Load	CPU	Memory	Swap
system_vagrant64-open-transit-indicators	Running	[0.70] [0.98] [0.70]	0.4%us, 1.3%sy, 0.0%wa	13.3% [1090172 kB]	0.0% [0 kB]

Process	Status	Uptime	CPU Total	Memory Total
web-server-nginx	Not monitored	-	-	-
database-PostgreSQL	Running	21m	0.0%	0.3% [32160 kB]
webapp-gunicorn	Running	4m	0.1%	2.7% [221420 kB]
tileserver-windshaft	Running	4m	0.0%	0.7% [62964 kB]
indicator-queue-celery	Running	6m	0.0%	1.1% [91216 kB]
datasource-queue-celery	Running	6m	0.0%	2.0% [170704 kB]
indicator-calc-scala	Running	4m	2.0%	5.6% [465208 kB]

Filesystem	Status	Space usage	Inodes usage
root	Accessible	12.1% [5764.7 MB]	4.1% [215999 objects]

Copyright © 2000-2011 Tildeslash. All rights reserved. Monit web site | Monit Wiki | M/Monit

The application is designed to start all required services on server reboot, and to automatically restart any service that might fail during use. If an issue arises while the application is running, this page would be the first place to see if there are any errors. If one of the processes is marked as not running, click the name of the process, scroll to the bottom of the page, and click the "Start Service" button.

This page can also be consulted while the application is processing long-running tasks like indicator calculations. CPU and memory usage for "indicator-calc-scala", in that example, should be near the top of the machine's capacity.

Troubleshooting

This application should be considered beta software, and may become unstable in situations that have not been encountered and tested. If the software becomes unusable even with all services running, it may be necessary to rebuild the database.

Note: this will erase the data and return the machine to default settings. Export indicator result CSVs, if possible, before following this process.

SSH into the server and switch user to "postgres" with

```
$ sudo su postgres
```

Then execute the 'psql' command to enter the PostgreSQL database.

List all tables with the PostgreSQL "\list" command as seen below:

```
vagrant@vagrant64-open-transit-indicators:~/oti/open-transit-indicators$ sudo su postgres
postgres@vagrant64-open-transit-indicators:/home/vagrant/oti/open-transit-indicators$ psql
psql (9.1.14)
Type "help" for help.

postgres=# \list
                                  List of databases
       Name        |       Owner        | Encoding |   Collate   |    Ctype    |   Access privileges
-------------------+--------------------+----------+-------------+-------------+-----------------------
 postgres          | postgres           | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0         | postgres           | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
                   |                    |          |             |             | postgres=CTc/postgres
 template1         | postgres           | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
                   |                    |          |             |             | postgres=CTc/postgres
 template_postgis  | postgres           | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 transit_indicators | transit_indicators | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
(5 rows)

postgres=# drop database 'transit_indicators';
```

To delete the database, issue the command:

```
drop database 'transit_indicators';
```

Exit postgres with the "\q" command, then rerun the deployment script (/deployment/provision.sh).

Acknowledgements

This administrators' manual was developed to support the World Bank Open Transit Indicators project, with funding provided by Australian Aid (formerly known as AusAid).

The World Bank project team, led by **Ms. Holly Krambeck**, **Dr. Li QU**, and **Mr. Christopher DeSerio**, would like to express their sincere thanks to **Dr. Yulin JIANG**, Director of China Urban Sustainable Transport Research Center, for her unwavering support for the project and substantial inputs provided by her team, including **Dr. Cheng LI** and **Dr. Xianglong LIU**. The team would also like to thank **Mr. Lei YAN** from the Zhengzhou Bus Company for his diligent work in testing the platform and invaluable technical support he has provided to the other participating cities. And a special thanks to **Ms. Linghong ZOU**, the team intern during the summer of 2013, who helped build the critical foundations for the project.

The team would like to thank **Ms. Van Anh Thi Tran**, Senior Transportation Specialist and Task Team Leader for the Haiphong transport project, for her support for this initiative. The team would also like to express its appreciation for **Ms. Kate Chapman** and the Humanitarian OpenStreetMap Team, working with **Dr. Tran Khanh Toan** and **Professor Le Sy Xinh** from the Vietnam Maritime University's GIS program, for their excellent work on open source digital mapping and GTFS data collection activities.

The team would also like to thank **Mr. Aaron Antrim**, founding principal of Trillium Solutions**; Mr. Michael Smith**, founding principal of Transitime; and **Ms. Bibiana McHugh**, IT Manager of Trimet – these three remarkable individuals provided critical guidance and technical knowledge to the participating pilot governments and to the team on GTFS and GTFS-RT.

The team would like to express its appreciation of **Mr. Kevin Webb**, founding principal of Conveyal, and **Mr. James Wong**, independent transit consultant, for inspiring the development of this project through their ground-breaking work in GTFS data analysis and open-source software development.

Finally, the team would like to thank **Mr. John Branigan**, Azavea GIS Project Manager, and his extensive team of talented and exceedingly hard working developers for truly going above and beyond the call of duty in building a software application to support better transit planning in developing countries.

Resources

Additional information on the project can be found from the following resources:

Open Transit Indicators
https://github.com/WorldBank-Transport/open-transit-indicators

Transitime "stop_times.txt" Generator: a software tool for generating a file required to evaluate transit system on-time performance in Open Transit Indicators
https://github.com/WorldBank-Transport/Transitime

International GTFS Training Materials: Link Repository
https://github.com/WorldBank-Transport/GTFS-Training-Materials/wiki/Link-repository-for-international-GTFS-training-materials