
Deployment and Configuration Guide

Consolidated Mirth Dashboard

VERSION 1.0



Copyright

This document is Client Confidential and contains proprietary information, including trade secrets of CitiusTech. Neither the document nor any of the information contained in it may be reproduced or disclosed to any unauthorized person under any circumstances without the express written permission of Client.

Revision History

Document Version #	Revision Date	Prepared By	Approved By	Approval Date	Summary of Changes
1.0	22-07-2020	Suchetana Shetty	Akshaya Subramaniam	Click here to enter a date.	First Version
	Click here to enter a date.			Click here to enter a date.	



Contents

1	INTRODUCTION	4
2	DEPLOYMENT PRE-REQUISITES	5
2.1	SOFTWARE REQUIREMENTS.....	5
2.2	HARDWARE REQUIREMENTS.....	5
2.3	NETWORKING REQUIREMENTS	5
2.4	SECURITY REQUIREMENTS	5
2.5	DEPENDENCIES	5
2.6	PRE-DEPLOYMENT CONFIGURATION	6
3	DEPLOYMENT PROCEDURE	7
3.1	SCOPE.....	7
3.2	ENVIRONMENT	7
3.3	BACKUP PROCEDURE	7
3.4	DEPLOYMENT PROCEDURE	7
3.5	INCREMENTAL UPGRADE PROCEDURE	12
4	POST DEPLOYMENT	14
4.1	CONFIGURATION	14
4.2	VALIDATION.....	19
5	TROUBLESHOOTING AND SUPPORT	22
5.1	PROBLEM 1	22
5.2	PROBLEM 2	ERROR! BOOKMARK NOT DEFINED.
5.3	KNOWN ISSUES.....	ERROR! BOOKMARK NOT DEFINED.
5.4	24x7 SUPPORT	ERROR! BOOKMARK NOT DEFINED.
6	ROLL BACK PROCEDURE	23
7	UNINSTALLATION PROCEDURE.....	24



1 Introduction

There are multiple Mirth Instances implemented for a client. To monitor multiple instances, monitoring team should go and check separate dashboards of each Mirth instances.

As part of this IAG, a Mirth consolidated dashboard has been created where in all the Mirth instances' data are scraped into a single dashboard.

This document is intended for monitoring team which helps them deploy the application on to the monitoring server.



2 Deployment Pre-Requisites

Mirth Connect should be present.

2.1 Software Requirements

- Linux OS
- Go 1.14.2
- Grafana 7.0.4
- Prometheus 2.18.0
- Node Exporter 1.0
- Mirth Exporter

2.2 Hardware Requirements

Minimum Hardware Requirements:

- Processor: 4 cores
- Processor speed: 1 GHz
- Random access memory (RAM): 8 GB
- Hard disk capacity: 500GB

Recommended Hardware Requirements:

- Processor: 8 cores
- Processor speed: 3 GHz
- Random access memory (RAM): 16 GB

2.3 Networking Requirements

Default Ports:

- Grafana = :3000
- Prometheus = :9090
- Mirth exporter = :9140
- Node exporter = :9100

2.4 Security Requirements

Root user access or sudo privileges are required.

2.5 Dependencies

Mirth Connect should be there with deployed channels.



2.6 Pre-deployment Configuration

1. Root user or sudo privileges is required.
2. Access to ports 3000, 9090, 9140, 9100, 9140 is required.



3 Deployment Procedure

The following section provides the detailed steps to successfully deploy each module of the product.

3.1 Scope

- Deployment of Go compiler
- Grafana set up
- Prometheus set up
- Mirth Exporter configuration
- Node Exporter configuration

3.2 Environment

Not Applicable

3.3 Backup Procedure

Backup of Grafana database can be taken to ensure that you can always rollback to your previous version. This can be done by taking backup of **grafana.db** file. This is usually located at **/var/lib/grafana/grafana.db** on Unix systems. If you are unsure about what database you use and where it is stored, you can check your Grafana configuration file. If you have installed Grafana to custom location using a binary tar/zip, it is usually in **<grafana_install_dir>/data**.

3.4 Deployment Procedure

For GO set up, perform the following steps:

1. Download the GO tarball using **wget** command:

```
wget https://dl.google.com/go/go1.14.2.linux-amd64.tar.gz
```

2. Extract the downloaded tarball using the following command inside **/usr/local** directory. Ensure to run the following command as a root user:

```
sudo tar -C /usr/local -xzf go1.14.2.linux-amd64.tar.gz
```

3. Set path environment variable in order to find Go executable binaries by system. Open **/.bash_profile** using the following command:

```
sudo nano ~/.bash_profile
```

Now, add the following lines to it:

```
export PATH=$PATH:/usr/local/go/bin:$GOPATH/bin
```

Save and close the file and run the following command to reload running profile:

```
source ~/.bash_profile
```

4. Installation of GO can be verified with the following command:

```
go version
```



To set up Grafana, perform the following steps:

1. Add Grafana yum repository.

Run the following commands as user with sudo privileges or as root user to add repository content:

```
cat <<EOF | sudo tee /etc/yum.repos.d/grafana.repo
[grafana]
name=grafana
baseurl=https://packages.grafana.com/oss/rpm
repo_gpgcheck=1
enabled=1
gpgcheck=1
gpgkey=https://packages.grafana.com/gpg.key
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt
EOF
```

2. Run update to add Grafana the repo and to import the GPG key by the following command:

```
dnf update
```

3. Install Grafana using the following command:

```
dnf install grafana
```

4. Start Grafana server and enable it to start on boot:

```
systemctl daemon-reload
systemctl start grafana-server
systemctl enable grafana-server
```

5. To verify status of Grafana server run the following command:

```
systemctl status grafana-server
```

6. Default port of Grafana is 3000. To allow port 3000 for access to the dashboard, execute the following commands:

```
sudo firewall-cmd --add-port=3000/tcp --permanent
sudo firewall-cmd --reload
```

7. To open the Grafana Web UI, navigate to <http://localhost:3000> with your Web browser:



The Login page displays. The default login username and password are **admin** and it allows to change username and password during first login.

To setup Mirth Exporter, perform the following steps:



1. Download the file **mirth_exporter.go**.



2. Save **mirth_exporter.go** in the **Mirth Connect** folder.
3. To execute **mirth_exporter.go**, following command can be used:

```
go run mirth_exporter.go &
```

To set up Node Exporter, perform the following steps:

1. To install Node Exporter, login with root user or user with sudo privileges.
2. Using **wget** command, download **node_exporter** tarball:

```
wget https://github.com/prometheus/node_exporter/releases/download/v1.0.0/node_exporter-1.0.0.linux-amd64.tar.gz
```

3. Extract the file using the following command:

```
tar -zxpvf node_exporter-1.0.0.linux-amd64.tar.gz
```

4. Contents of the extracted folder can be checked using the following command:

```
tree node_exporter-1.0.0.linux-amd64.tar.gz
```

```
# tree node_exporter-1.0.0.linux-amd64
node_exporter-1.0.0.linux-amd64
├── LICENSE
├── node_exporter
└── NOTICE
```

5. Copy the binary file **node_exporter** to **/usr/local/bin** path using the following command:

```
cp node_exporter-1.0.0.linux-amd64/node_exporter /usr/local/bin
```

6. Set the file permissions of the **node_exporter** file using the following command:

```
chown root /usr/local/bin/node_exporter
```

7. To configure **node_exporter** to run as a service, create a **systemd** service file using the following command:

```
vi /etc/systemd/system/node_exporter.service
```

Then paste the following code and save the file:

```
[Unit]
Description=Prometheus Node Exporter
Wants=network-online.target
After=network-online.target

[Service]
User=node_exporter
Group=node_exporter
Type=simple
ExecStart=/usr/local/bin/node_exporter

[Install]
WantedBy=multi-user.target
```

8. Reload systemd manager with the following command:

```
systemctl daemon-reload
```

9. To start and enable **node_exporter** service, use the following commands:



```
systemctl start node_exporter
systemctl enable node_exporter
```

10. To check the status of service, execute the following command:

```
systemctl status node_exporter
```

```
● node_exporter.service - Node Exporter
   Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; vendor p
   Active: active (running) since Wed 2020-07-15 09:27:22 UTC; 1 weeks 1 days a
 Main PID: 28062 (node_exporter)
    Tasks: 6 (limit: 26213)
   Memory: 15.7M
   CGroup: /system.slice/node_exporter.service
           └─28062 /usr/local/bin/node_exporter
```

11. By default, **node_exporter** uses port 9100. To check the same, use the following command:

```
netstat -tunlp
```

```
0 :::9004          :::*              LISTEN          18436/java
0 :::9100          :::*              LISTEN          28062/node_exporter
0 :::9005          :::*              LISTEN          2045/java
```

12. To open port 9100 in firewall, use the following commands:

```
firewall-cmd --add-port=9100/tcp --permanent
firewall-cmd --reload
```

To set up Prometheus, perform the following steps:

1. To install Prometheus, login with root user or user with sudo privileges.
2. Create configuration directories for Prometheus using the following command:

```
mkdir /etc/prometheus
mkdir /var/lib/Prometheus
```

3. Set the ownership on **/var/lib/Prometheus** using the following command:

```
chown root /var/lib/prometheus/
```

4. Prometheus tar file can be downloaded using the following command:

```
dnf install wget -y
wget
https://github.com/prometheus/prometheus/releases/download/v2.18.0/promet
heus-2.18.0.linux-amd64.tar.gz -P /tmp
```

5. Once the download completes, extract the tarball file using the following command:

```
tar -zxpvf prometheus-2.18.0.linux-amd64.tar.gz
```

6. Directory structure can be seen using **tree** command as follows:

```
tree /tmp/prometheus-2.18.0.linux-amd64
```



```

/tmp/prometheus-2.18.0.linux-amd64
├── console_libraries
│   ├── menu.lib
│   └── prom.lib
├── consoles
│   ├── index.html.example
│   ├── node-cpu.html
│   ├── node-disk.html
│   ├── node.html
│   ├── node-overview.html
│   ├── prometheus.html
│   └── prometheus-overview.html
├── LICENSE
├── NOTICE
├── prometheus
├── prometheus.yml
├── promtool
└── tsdb

```

- The extracted directory contains 2 binary files **prometheus** and **promtool**. Copy these files to the `/usr/local/bin` path using the following commands:

```

cd /tmp/prometheus-2.14.0.linux-amd64
cp prometheus /usr/local/bin
cp promtool /usr/local/bin

```

- To set configurations for Prometheus, create a **prometheus.yml** file:

```
vi /etc/prometheus/prometheus.yml
```

- Paste the following configurations into the **.yml** file and save it:

```

# Global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds.
                        # Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default
                           # is every 1 minute.
  scrape_timeout: 15s # scrape_timeout is set to the global default
                      # (10s).
# A scrape configuration containing exactly one endpoint to scrape: # Here
# it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries
  # scraped from this config.
  - job_name: 'prometheus'

```

```

  # metrics_path defaults to '/metrics'
  # scheme defaults to 'http'.

  static_configs:
    - targets: ['localhost:9090']
    - job_name: 'mirth'

  # metrics_path defaults to '/metrics'
  # scheme defaults to 'http'.

  static_configs:
    - targets: ['localhost:9140']
    - job_name: 'node exporter'

  # metrics_path defaults to '/metrics'
  # scheme defaults to 'http'.

  static_configs:

```



```
- targets: ['localhost:9100']
```

10. To open port to external connections, type the following commands:

```
firewall-cmd --add-port=9090/tcp --permanent
firewall-cmd --reload
```

11. To configure Prometheus to run as a service, create a systemd service file using the following command:

```
vi /etc/systemd/system/prometheus.service
```

Then paste the following code and save the file:

```
[Unit]
Description=Prometheus Time Series Collection and Processing Server
Wants=network-online.target
After=network-online.target
```

```
[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/usr/local/bin/prometheus \
  --config.file /etc/prometheus/prometheus.yml \
  --storage.tsdb.path /var/lib/prometheus/ \
  --web.console.templates=/etc/prometheus/consoles \
  --web.console.libraries=/etc/prometheus/console_libraries
```

```
[Install]
WantedBy=multi-user.target
```

12. Reload systemd manager with the following command:

```
systemctl daemon-reload
```

13. To start and enable Prometheus service, use the following commands:

```
systemctl start prometheus
systemctl enable prometheus
```

14. To check the status of service, execute the following command:

```
systemctl status Prometheus
```

```
● prometheus.service - Prometheus Time Series Collection and Processing Server
   Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2020-07-16 14:12:56 UTC; 6 days ago
     Main PID: 321 (prometheus)
        Tasks: 9 (limit: 26213)
       Memory: 174.6M
      CGroup: /system.slice/prometheus.service
              └─321 /usr/local/bin/prometheus --config.file /etc/prometheus/prometheus.yml --web.enable-admin-api
```

15. By default, Prometheus uses port 9090. Check the same using the following command:

```
netstat -tunlp
```

```
0 | :::9090          :::*               LISTEN          321/prometheus
```

3.5 Incremental Upgrade Procedure

For Prometheus upgradation, perform the following steps:

1. Stop all running services using the following command:

```
systemctl stop prometheus.service
```



2. Download the Prometheus archive and checksum using the following command:

```
wget
https://github.com/prometheus/prometheus/releases/download/v2.19.2/promet
heus-2.19.2.linux-amd64.tar.gz
```

3. Now, unpack the archive:

```
tar -xvf prometheus-2.19.2.linux-amd64.tar.gz
cd prometheus-2.19.2.linux-amd64/
```

4. Copy the prometheus and promtool executables to the `/usr/local/bin` directory:

```
cp prometheus-2.19.2.linux-amd64/{prometheus,promtool} /usr/local/bin/
chown root /usr/local/bin/{prometheus,promtool}
```

5. Restart all the services:

```
systemctl daemon-reload
systemctl start prometheus
systemctl enable Prometheus
```

Grafana Upgradation:

Before upgrading, backup of Grafana database can be taken. This will ensure that you can always rollback to your previous version. This can be done by taking backup of **grafana.db** file. This is usually located at **/var/lib/grafana/grafana.db** on Unix systems. If you are unsure what database you use and where it is stored check your Grafana configuration file. If you have installed Grafana to custom location using a binary tar/zip, it is usually in **<grafana_install_dir>/data**.

1. To stop Grafana service, use the following command:

```
systemctl stop grafana-server.service
```

2. Follow the same installation steps mentioned in section [3.4](#) (for setting up Grafana) and execute **yum install** command with new package:

```
sudo yum update Grafana
```

3. To restart Grafana service, use the following commands:

```
systemctl daemon-reload
systemctl start grafana-server.service
systemctl enable grafana-server.service
```



4 Post Deployment

4.1 Configuration

Prometheus Configuration:

To scrape metrics from Mirth Exporter and Node Exporter, configure **prometheus.yml** file by performing the following steps:

1. Open **prometheus.yml** file using the following command:

```
vi /etc/prometheus/prometheus.yml
```

2. Add the following lines to **prometheus.yml** file:

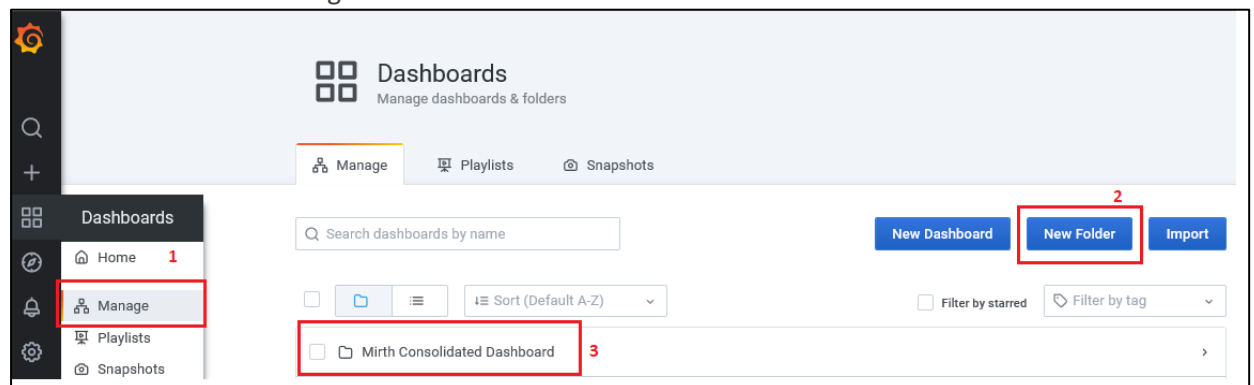
```
- job_name: 'mirth'
  static_configs:
    - targets: ['localhost:9140']
- job_name: 'node exporter'
  static_configs:
    - targets: ['localhost:9100']
```

3. Save the file.

Mirth Dashboard Configuration on Grafana:

To configure Mirth Dashboard, perform the following steps:

1. Login to Grafana using the URL <http://localhost:3000> or <http://<server ip, where Grafana is installed>:3000>.
2. Default username and password is admin, login using the same credentials.
3. Go to Dashboard → Manage. Click New folder and name its Mirth Consolidated Dashboard:



4. Download all json files which are under the following URL and save the files:
https://github.com/SalmanCitiustech/MirthDashboard/tree/MirthDashboard_Linux
5. There is one summary page **Mirth_Metrics_Dashboard.json** and 8 drill down pages present as shown in the following screenshot:



The screenshot shows the GitLab interface for a repository named 'Dashboard for Linux' by Suchetana Shetty. The left sidebar shows the 'Repository' section with 'Files' selected. The main content area displays a table of files in the repository.

Name	Last commit	Last update
CPU_details.json	Dashboard for Linux	2 weeks ago
Channel_Metrics.json	Dashboard for Linux	2 weeks ago
Channel_Throughput_Monitor.json	Dashboard for Linux	2 weeks ago
Disk_Details.json	Dashboard for Linux	2 weeks ago
Errored_Channels.json	Dashboard for Linux	2 weeks ago
Inactive_Instances.json	Dashboard for Linux	2 weeks ago
Memory_Details.json	Dashboard for Linux	2 weeks ago
Mirth_Metrics_Dashboard.json	Dashboard for Linux	2 weeks ago
Queued_Channels.json	Dashboard for Linux	2 weeks ago
README.md	Initial commit	2 months ago

6. On Grafana, go to **Configuration** → **Data Sources** and click **Add data source**:


The screenshot shows the Grafana Configuration page for 'Main Org.'. The 'Data Sources' tab is selected in the top navigation bar. In the left sidebar, the 'Configuration' menu is open, and 'Data Sources' is highlighted. A search bar is present with the placeholder text 'Search by name or type'. A red box highlights the 'Add data source' button in the top right corner.

7. Select **Prometheus**:

The screenshot shows the 'Add data source' page in Grafana. The title is 'Add data source' with the subtitle 'Choose a data source type'. There is a search bar with the placeholder text 'Filter by name or type'. Under the 'Time series databases' section, the 'Prometheus' option is selected. It includes the Prometheus logo, the text 'Prometheus', 'Open source time series database & alerting', and a 'Core' tag. There are 'Learn more' and 'Select' buttons. The 'Graphite' option is partially visible below.

8. Mention name for datasource. Provide URL <http://localhost:3000> as URL then click **Save & Test**



 **Data Sources / Prometheus**
Type: Prometheus

Settings

Dashboards

Name

Prometheus

Default

☐

HTTP

URL

http://localhost:3000

Access

Server (default)

Help >

Whitelisted Cookies

Add Name

Add

Auth

Basic auth

☐

With Credentials

☐

TLS Client Auth

☐

With CA Cert

☐

Skip TLS Verify

☐

Forward OAuth Identity

☐

Custom HTTP Headers

+ Add header

Scrape interval

15s

Query timeout

60s

HTTP Method

Choose

Misc

Disable metrics lookup

☐

Custom query parameters

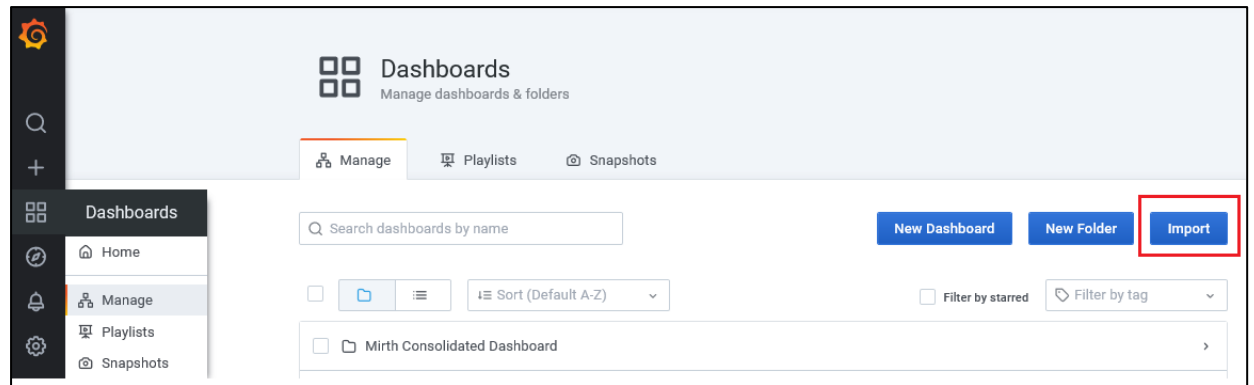
Example: max_source_resolution=5m&timeout=10

Save & Test

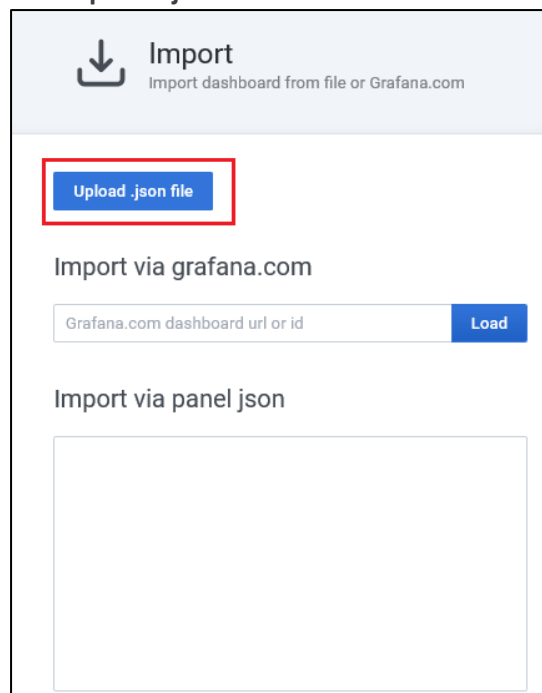
Delete

Back

9. On Grafana, go to **Dashboard** → **Manage** and click **Import**:




10. Click **Upload .json file**:



11. Browse to the downloaded .json file mentioned in Step 4. Choose folder name as **Mirth Consolidated Dashboard** as shown in the following window and click **Import**:





Import

Import dashboard from file or Grafana.com

Options

Name

Folder

Unique identifier (uid)

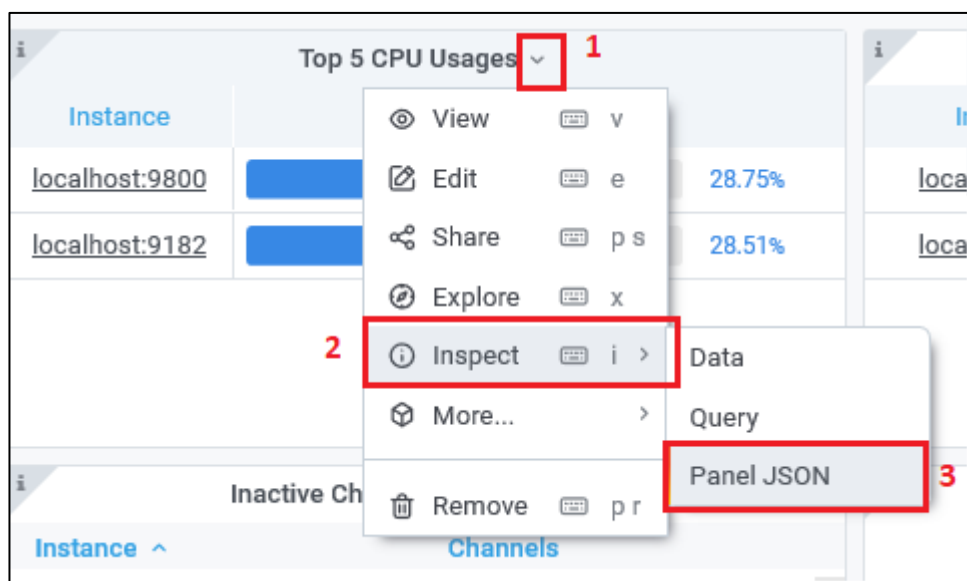
The unique identifier (uid) of a dashboard can be used for uniquely identify a dashboard between multiple Grafana installs. The uid allows having consistent URL's for accessing dashboards so changing the title of a dashboard will not break any bookmarked links to that dashboard.

12. Repeat steps 5 and 6 for all downloaded .json files.

Updating URLs of Grafana Dashboard and drill down pages:

To update URLs, present in panels of dashboard, perform the following steps:

1. Click the accordion which is next to the title of the panel and go to **Inspect** → **Panel JSON**. This will open the JSON panel:



2. Search for **url**:



Inspect: Top 5 CPU Usages
1 queries with total query time of 46 ms

Data Stats **JSON** Query

Select source
Panel JSON ▼ Copy to clipboard Apply

```
    "value": 120
  },
  {
    "id": "links",
    "value": [
      {
        "targetBlank": true,
        "title": "Details",
        "url": "http://localhost:3000/d/gLISNdMGk/cpu-details?orgId=1&refresh=10s"
      }
    ]
  },
  {
    "id": "mappings",
    "value": [
```

3. Replace **http://localhost:3000** with the server IP or server name where your Grafana is installed and save the file.
4. Repeat the preceding 3 steps for all the panels of dashboard and drill down pages.

4.2 Validation

1. **Validation of Prometheus:** Access the port 9090 of the server where Prometheus is installed and following screen should appear:



Prometheus Alerts Graph Status ▾ Help

☐ Enable query history

Expression (press Shift+Enter for newlines)

Execute - insert metric at cursor - ▾

Graph Console

◀ Moment ▶

Element	Value
no data	

Add Graph

2. **Validation of exporters:** Check the **Targets** under **Status** tab of Prometheus. All the status of exporters should be **UP**:

Prometheus Alerts Graph **Status ▾** Help

Targets

All Unhealthy

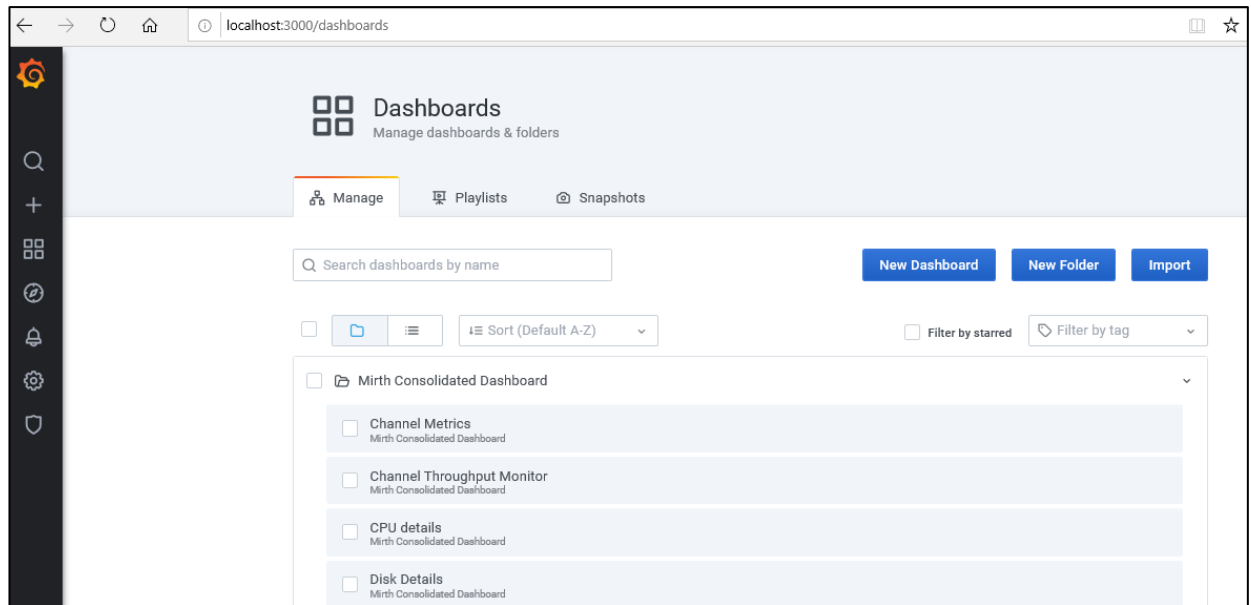
mirth (2/2 up) show less

node exporter (1/1 up) show less

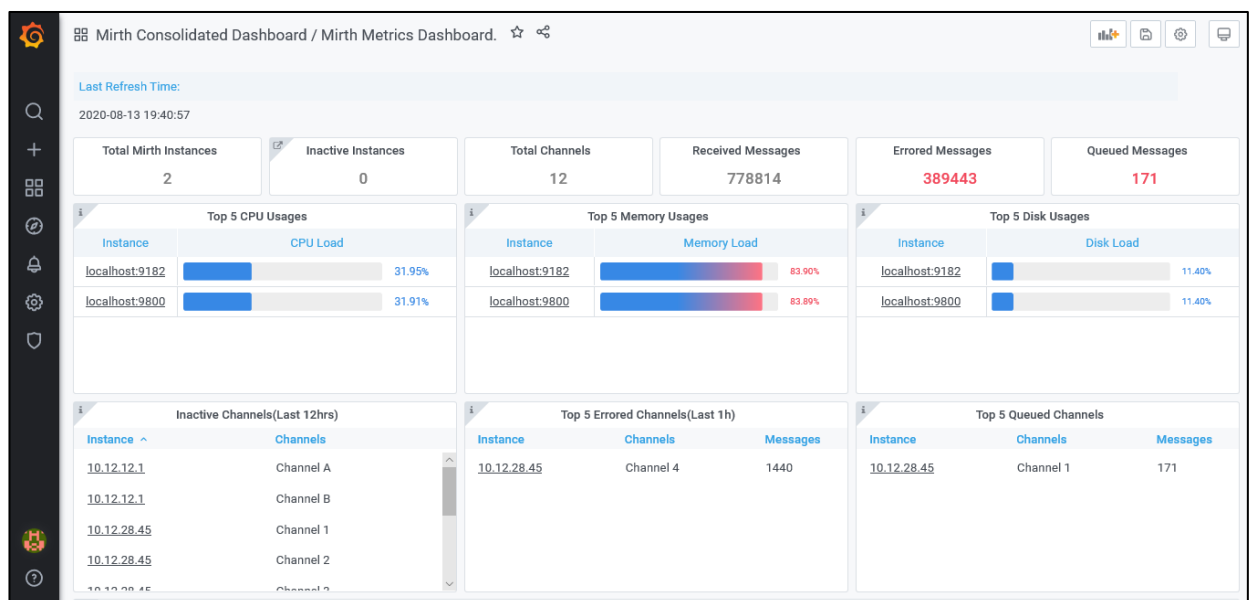
prometheus (1/1 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9140/metrics	UP	instance="localhost:9140" job="mirth"	11.712s ago	2.422s	
http://localhost:9100/metrics	UP	instance="localhost:9100" job="node exporter"	4.533s ago	10.05ms	
http://localhost:9110/metrics	UP	instance="localhost:9110" job="prometheus"	8.779s ago	4.774ms	

3. **Validation of Grafana:** Access <http://<server ip where Grafana is installed>/dashboards> and following screen should appear:



4. Select **Mirth Metrics Dashboard** under **Mirth Consolidated Dashboard** folder and the following screen should appear:



5. Check for all the drill downs by clicking **Instance** from all the panels containing **Instance** column.



5 Troubleshooting and Support

The following section describes a list of problems that may appear during or post installation of the system and how to resolve them:

5.1 Problem

Situation: Prometheus service is inactive.

Consequences: Prometheus is not able to access and not able to scrape any metrics.

Action: Add all the required targets following proper spacings and syntaxes. A single misspacing causes **prometheus.yml** to fail to execute and stops Prometheus service:

```
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>`
  - job_name: 'prometheus'

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ['localhost:9090']
      - job_name: 'node_exporter'

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ['localhost:9100']
      - job_name: 'mirth'

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ['localhost:9140']
```



6 Roll Back Procedure

To roll back the system to an older or previous version, perform the following steps:

1. To stop Grafana service, use the following command:

```
systemctl stop grafana-server.service
```

2. Follow the same installation steps mentioned in section [3.4](#) (for setting up Grafana) and execute the **yum install** command with older package:

```
sudo yum update Grafana
```

3. Copy backup of Grafana db file mentioned in section [3.5](#) to **/var/lib/grafana/grafana.db**.
4. To restart Grafana service, use the following commands:

```
systemctl daemon-reload
systemctl start grafana-server.service
systemctl enable grafana-server.service
```



7 Uninstallation Procedure

This section describes the procedure to uninstall the system.

Uninstallation of GO:

Type the following commands:

```
sudo rm -rf /usr/local/go
sudo apt-get uninstall purge golang*
```

Uninstallation of Grafana:

Type the following commands:

To remove Grafana:

```
sudo apt-get remove grafana
```

To remove Grafana and its dependencies:

```
sudo apt-get remove --auto-remove Grafana
```

Uninstallation of Mirth Exporter:

Remove file **mirth_exporter.go** from **Mirth Connect** folder using the following command:

```
rm mirth_exporter.go
```

Uninstallation of Node Exporter:

Type the following command:

```
sudo rm -rf /usr/local/bin/node_exporter
```

Uninstallation of Prometheus:

Type the following commands:

```
sudo rm -rf /usr/local/bin/prometheus
sudo rm -rf /usr/local/bin/promtool
sudo rm -rf /etc/prometheus
```