

Contents

[Description 2](#_Toc515666599)

[How to Do It? 4](#_Toc515666600)

[Installing DNIF 4](#_Toc515666601)

[Identifying the dataset and its source 5](#_Toc515666602)

[Understanding DNIF 6](#_Toc515666603)

[Capturing the Dataset & Feeding it to DNIF using API 7](#_Toc515666604)

[Automating Dynamic Data Feeding 16](#_Toc515666605)

[Creating Package 18](#_Toc515666606)

[Creating Widget(s) 20](#_Toc515666607)

[Creating Dashboard 23](#_Toc515666608)

[Configuring SMTP for DNIF 26](#_Toc515666609)

[Creating Alert(s) 26](#_Toc515666610)

# Description

**Process 2: Dynamic Data in DNIF**

The process is divided into 5 stages each of small description. This will help us further-to understand how we are solving problems in each stage or what is necessary for that particular stage to be completed and pass on to the next stage.

*Stage 1:*

* Select & understand data-set from a domain of interest.
* This is the initial step, which of-course sets the entire purpose of what kind of data you are looking to analyze and make visuals on. It is an important step as your quality of work depends entirely on this. (There is no limitation to the number of data sources, it will only be limited by imagination and lack of skills in further stages – meaning multiple datasets could be merged in to one forming single source using logic and code & then feed it to DNIF)

*Stage 2:*

* Understand DNIF platform and its capabilities limited to project scope.
* This step was introduced so that, one is beforehand ready and understands what the platform is, how it works, what are its capabilities, its features, the query language, deployment models, limitations, etc.

*Stage 3:*

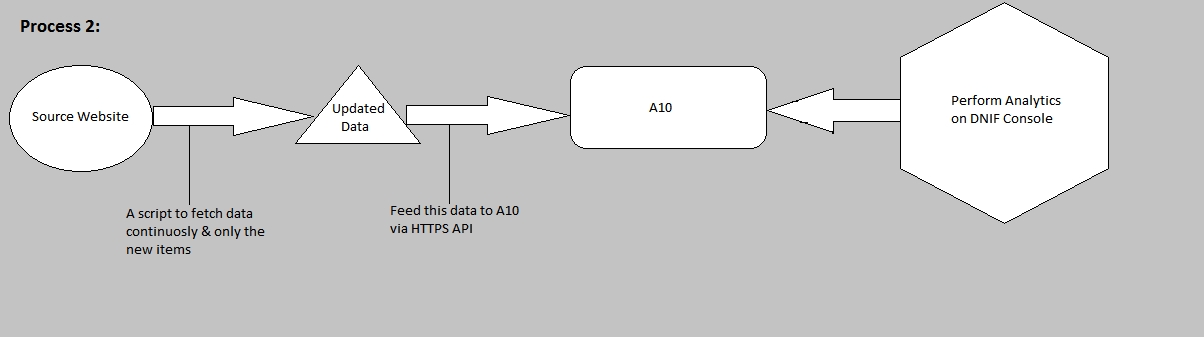
* Capturing data-set continuously.
* This stage is where the real game begins. Look out for the file format – ultimately we need a JSON file. Fetch the data continuously from the source selected in stage one and store/ update it in a file (ex; mydata.csv/ mydata.json)
* For this, one can write a code to do so or using an existing one from the repo.
* This file will get updated continuously depending on your scheduler/cron job or any other way of your choice.

*Stage 4:*

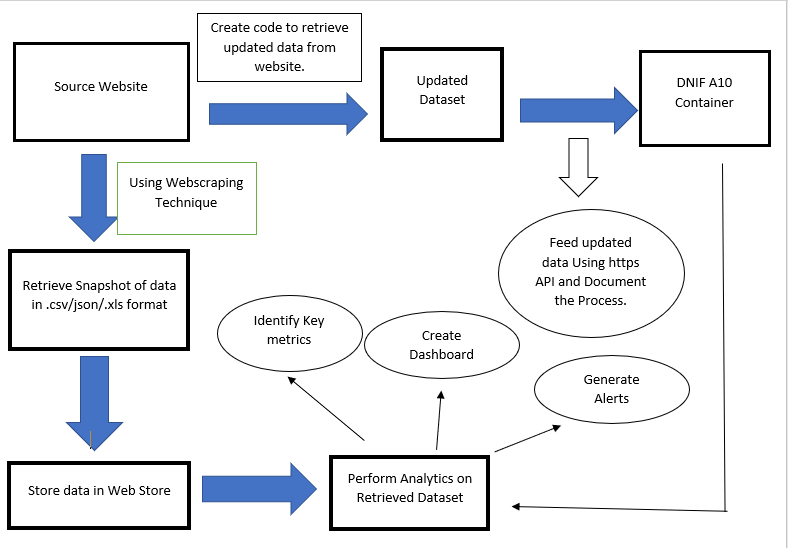
* The data captured needs to be fed to DNIF for it to be analyzed and played with.
* Use DNIF API which does this job for you and use the existing unified code (all in one code) from the repo - under Process 2 folder; a file called "SourceToDnif.py".

*Stage 5:*

* Perform analytics, create dashboards, & build alerts.
* Once your data is inside DNIF by uploading through the event store, all the skills and understanding acquired in stage 2 could be put into action.



*Fig 1: A Squeezed Perspective*



*Fig 2: A Squeezed Perspective of Process1 & Process 2 Combined*

# How to Do It?

Let us go in steps in order to achieve the outcome desired. These steps are descriptive in details and do not directly highlight which stage they accord to; Although if one has performed process 1, then the idea of how this process is being done and its correspondence with stages will be quite clear in understanding. Let us dive in and perform these sequentially.

**STEP 1:**

## Installing DNIF

* To do this, first “Sign-Up” with DNIF and read the pre-requisites guide here:

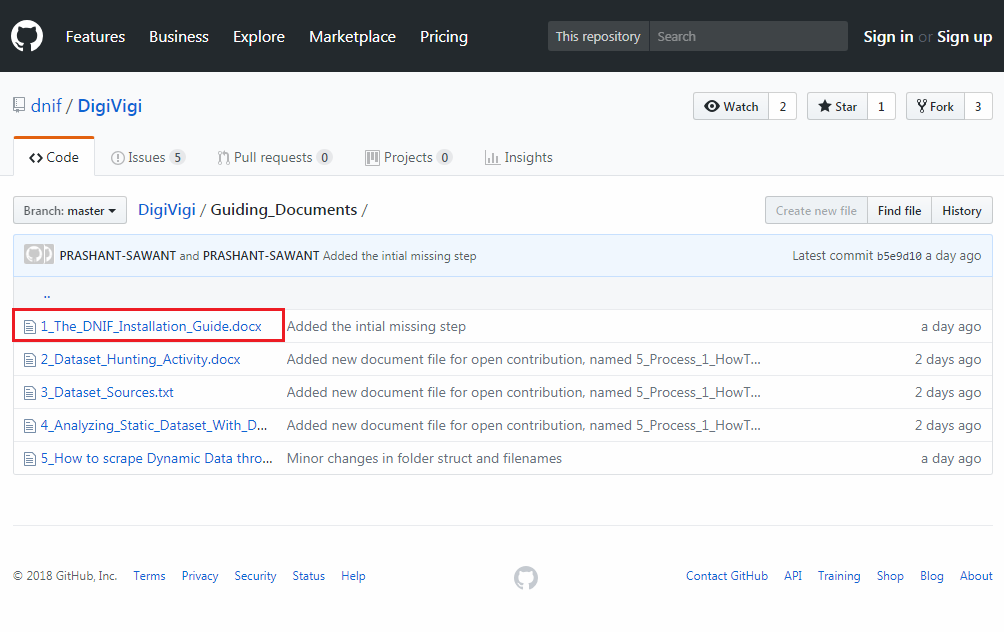
Sign-up: <https://dnif.it/signup.html>

Pre-requisites: <https://dnif.it/docs/guides/getting-started/prerequisites.html>

* Then follow the steps mentioned in **“1\_The\_DNIF\_Installation\_Guide.docx”** which is present in the repository. Here’s a link to it: <https://github.com/dnif/DigiVigi/tree/master/Guiding_Documents>
* There are videos and other documents as well which are provided by the Organization, which can guide, but it does not cover some details in it which we require. Here’s a link to it:

<https://dnif.it/docs/guides/getting-started/installing-dnif.html>

**NOTE:** In our case we have done the setup on a Virtual Machine. One can have a separate server like machine with a dedicated Ubuntu/ CentOS.



*Fig 3: DNIF Installation Pointer*

**STEP 2:**

## Identifying the dataset and its source

* Here’s a guide of how we did it for starters. Refer following documents:

1. Activity initialization:

<https://github.com/dnif/DigiVigi/blob/master/Guiding_Documents/2_Dataset_Hunting_Activity.docx>

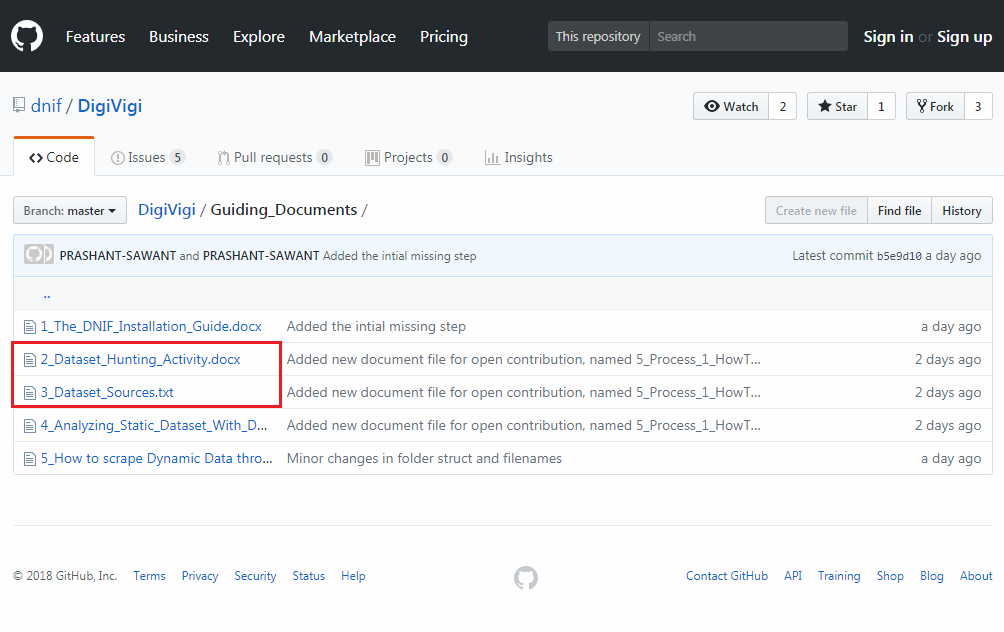
1. Jotting down dataset sources

<https://github.com/dnif/DigiVigi/blob/master/Guiding_Documents/3_Dataset_Sources.txt>

* Although it is not a must that our dataset sources might be of a good use to most analysts, since the purposes might differ.
* So in this current documented example, we’ve chosen a structured and static dataset from the source:

<https://www.webiron.com/abuse_feed/>

* **Webiron is not a static source** - In fact it posts newer data everyday on its website link mentioned above. In our case we will be scrapping data from their web page every day using a scheduler for our script mentioned in further steps.



*Fig 4: Dataset Repository*

**STEP 3:**

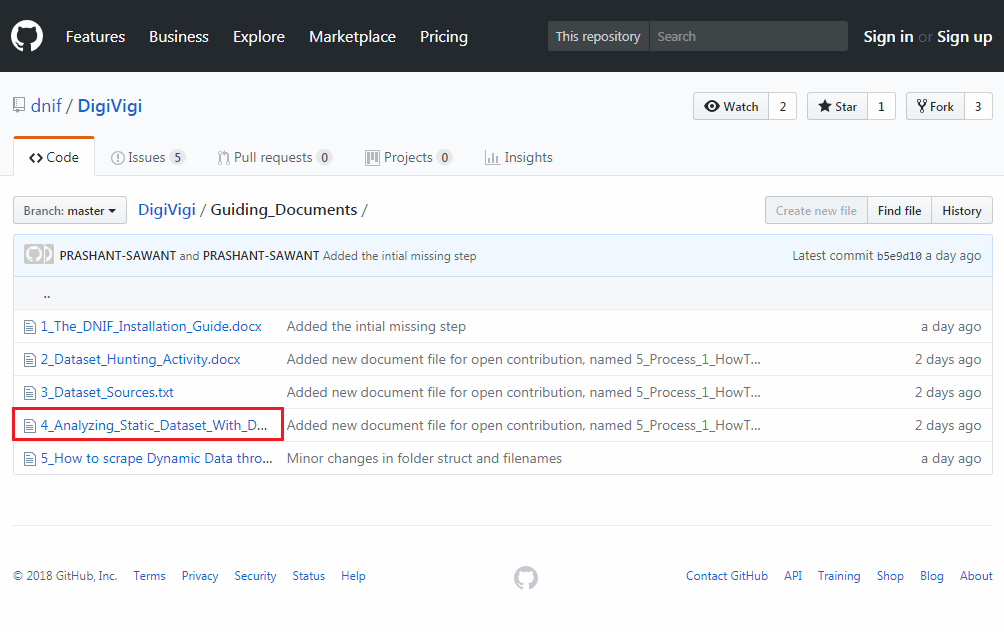
## Understanding DNIF

* This one here is a self-study step. But there’s no need worry, as DNIFs query language is very simple to learn if one is well versed with SQL. Yes, heard it right. They have this entire documentation of the query commands with examples.
* Head over here for getting an idea about it:

<https://dnif.it/docs/>

* The link above has everything in it for one to understand DNIF on a practical basis. From query language to tutorials and other documentations, most of it is present here originally.
* Also one of our contributors to “**DNIF Open Source Project**” has already given **a glimpse of data analysis**, widgets, dashboards and packages under the repository.

<https://github.com/dnif/DigiVigi/blob/master/Guiding_Documents/4_Analyzing_Static_Dataset_With_DNIF.docx>



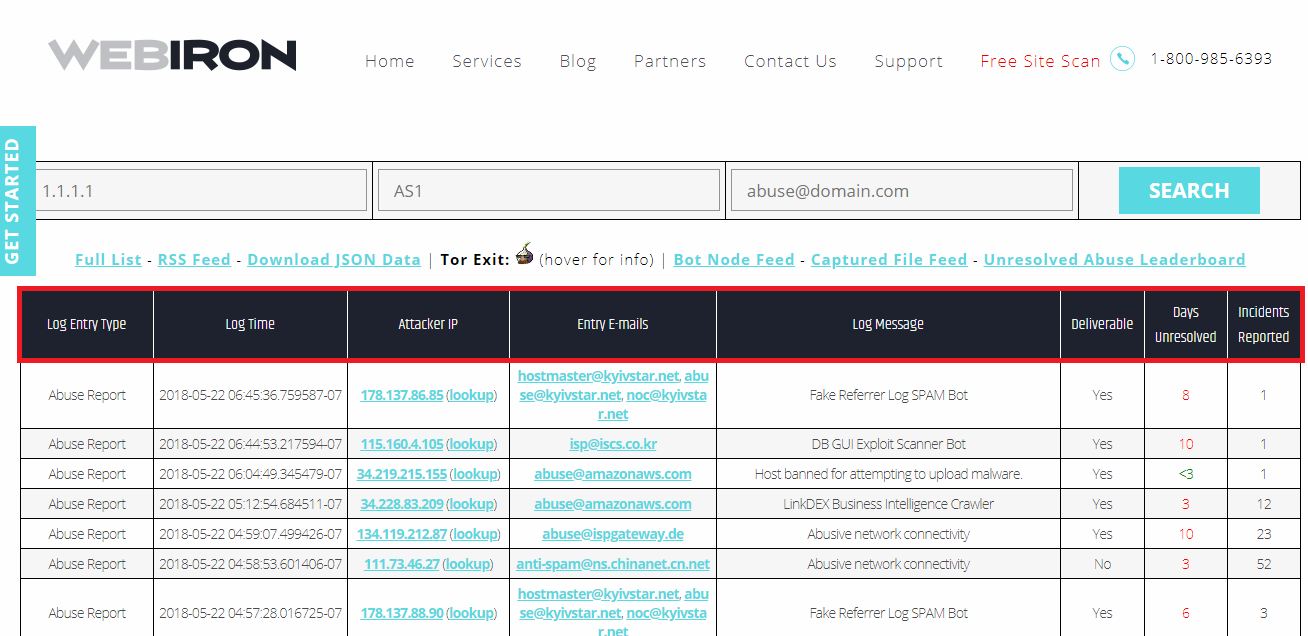
*Fig 5: Analysis Documentation*

**STEP 4:**

## Capturing the Dataset & Feeding it to DNIF using API

Let us have a look at our source of dataset. It is from Webiron which provides email abusers list on a daily basis. Each day a new list is maintained, and we need to scrape off this daily data and get it to DNIF. To keep it simple, we’ll go in some mini steps and try and keep it less complicated.

**Data-set Source:** <https://www.webiron.com/abuse_feed/>



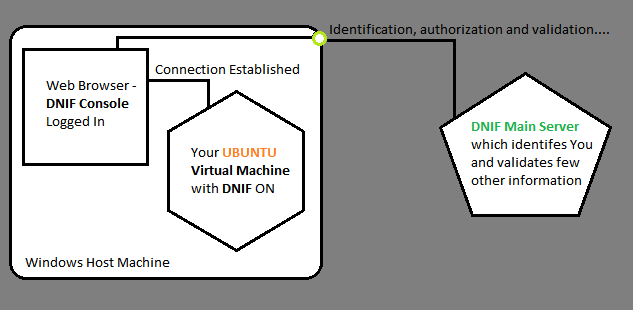
*Fig 6: Webiron Abuse Feed Header Highlight*

**NOTE:**

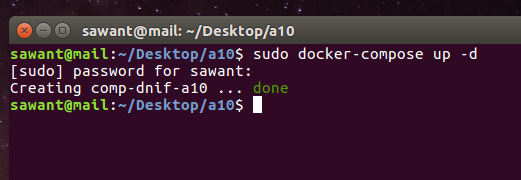
**NOW COMES A SERIES OF IMPORTANT STEPS THAT NEED TO BE DONE WHICH ARE IN THE ORDER DEPICTED**

***IS YOUR DNIF ECOSYSTEM WORKING FINE?* FIRST LETS DO THAT …**

**1st:** Make sure your DNIF container is up and running. The Pentagon part is your DNIF container which is running on Ubuntu.

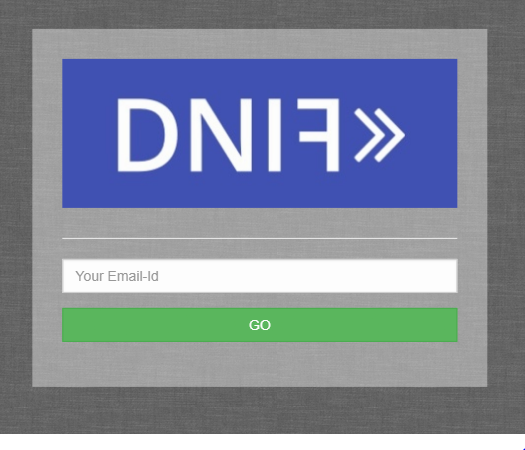


*Fig 7: Our DNIF Ecosystem*



*Fig 8: Running DNIF*

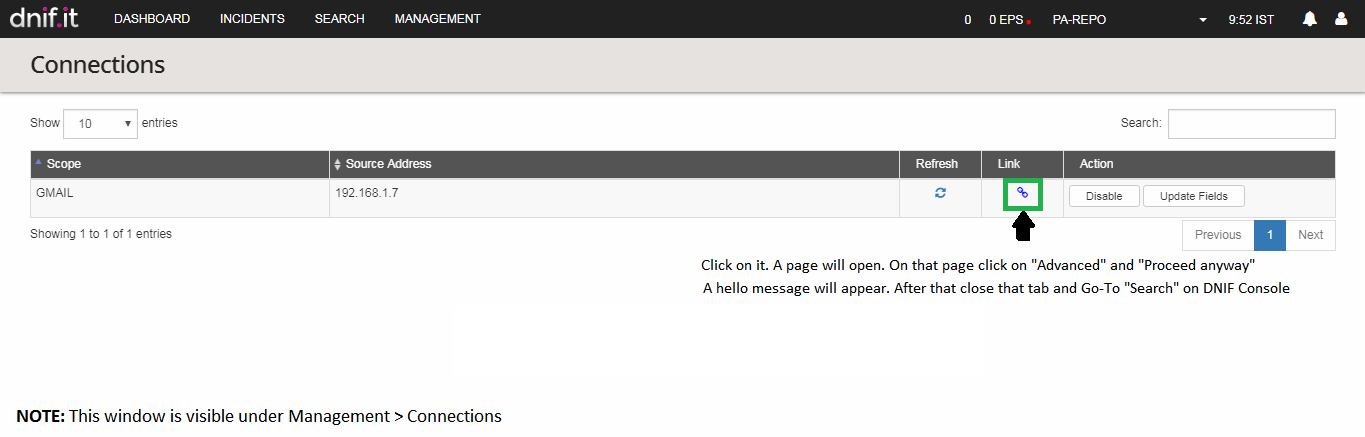
**2nd:** Log in to your DNIF console and also check your entire ecosystem is successfully connected.



*Fig 9: DNIF Login Console from Browser*

**3rd:** Under Management -> Connections -> Click the link icon -> Connection confirmation page will open up. Close that page, and now your link icon should become blue

If the confirmation page displays broken https, then click on Advanced Option -> Proceed. This project is not commercial and we’re in internal network, there is no harm in a unencrypted transfer of bytes.

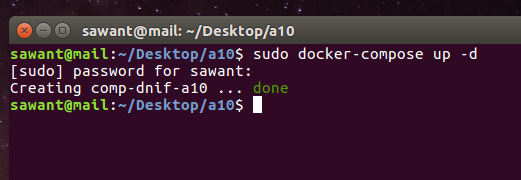


*Fig 10: Establishing Connection*

**Let us start with our Mini Steps Series**

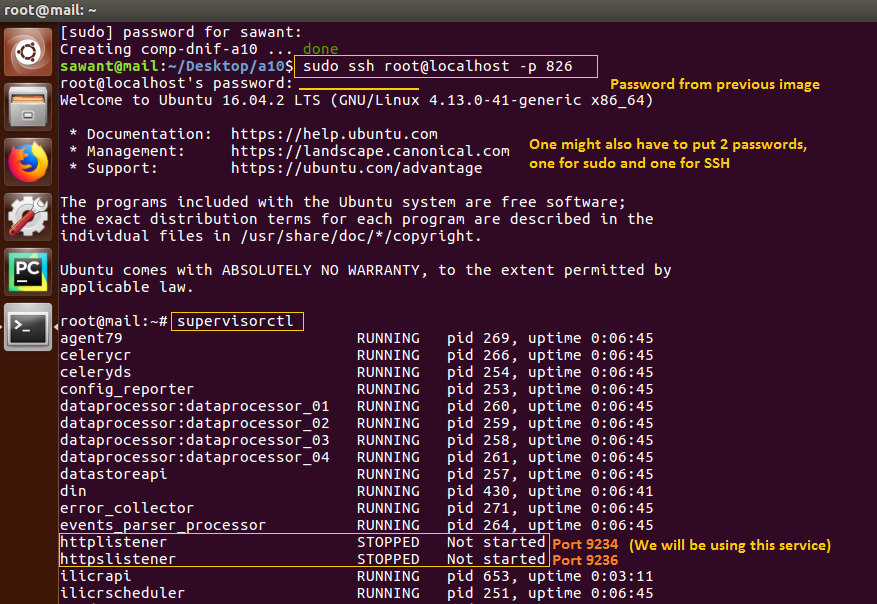
***Mini Step 1: Readying the DNIF Container to handle dynamic data***

1. On you Ubuntu Machine, make sure your DNIF Container is up



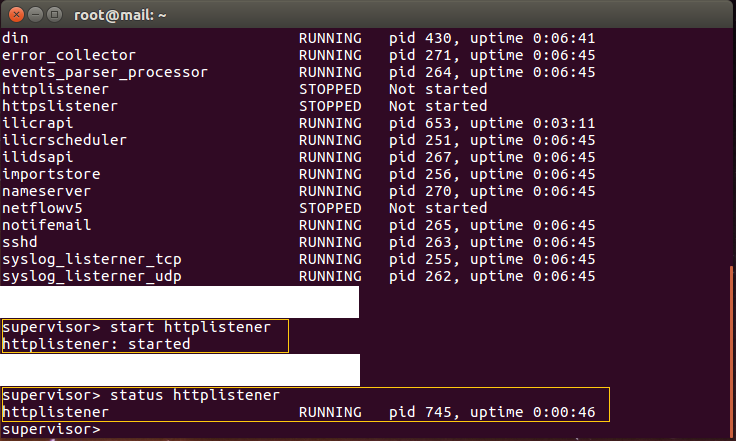
*Fig 11: DNIF Container Up*

1. Then do a SSH Connection to you DNIF
2. Once inside the DNIF container, type **supervisorctl** – and one can see all the services running in this DNIF container



*Fig 12: SSH to DNIF Container*

1. Now **ON** the HTTP/ HTTPS listener

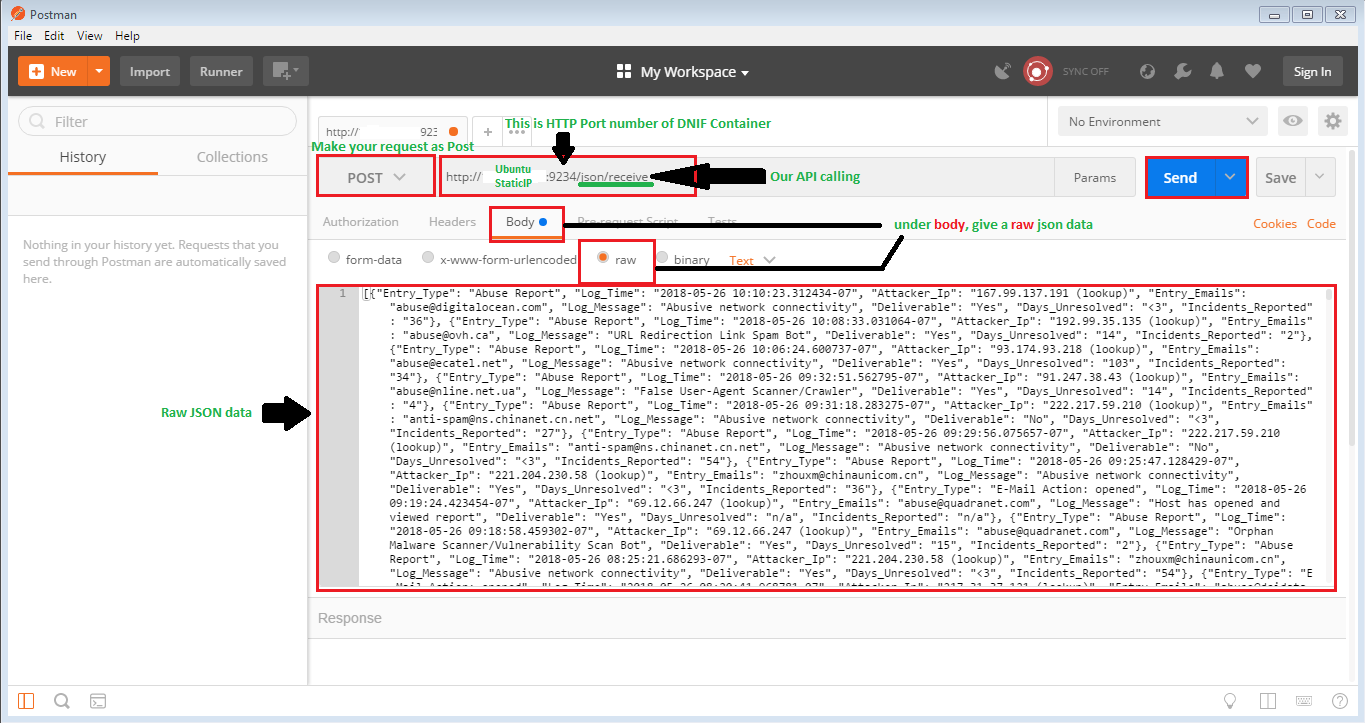


*Fig 13: HTTP Service ON*

1. To check if your DNIF API is ready for use, using Postman tool, one can test this. Postman is an app which helps test the Web APIs in use.

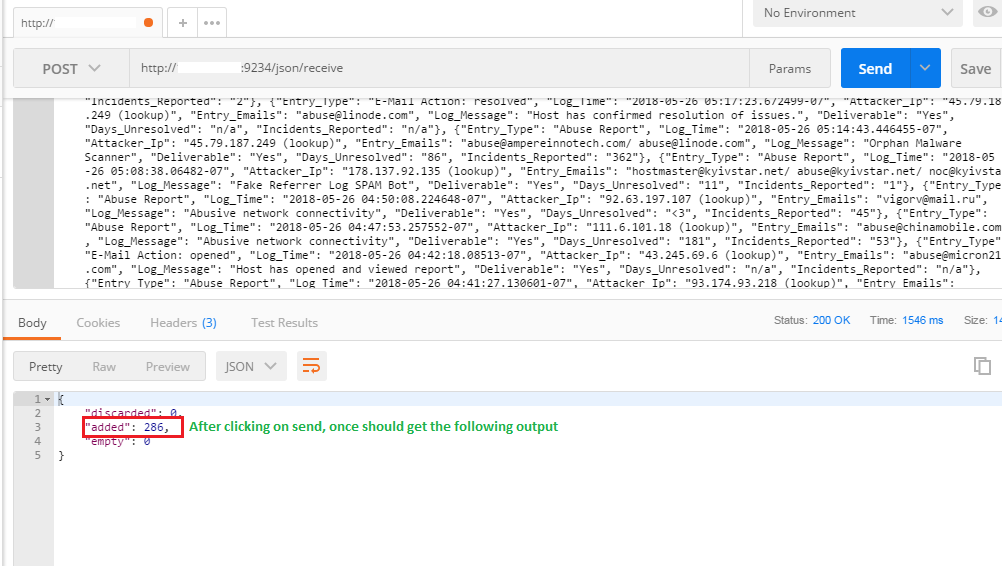
***Mini Step 2: Postman (Optional)***

* Using postman, we will first send a demo JSON data in raw format. This will help us to figure out if the API is working to its purpose.



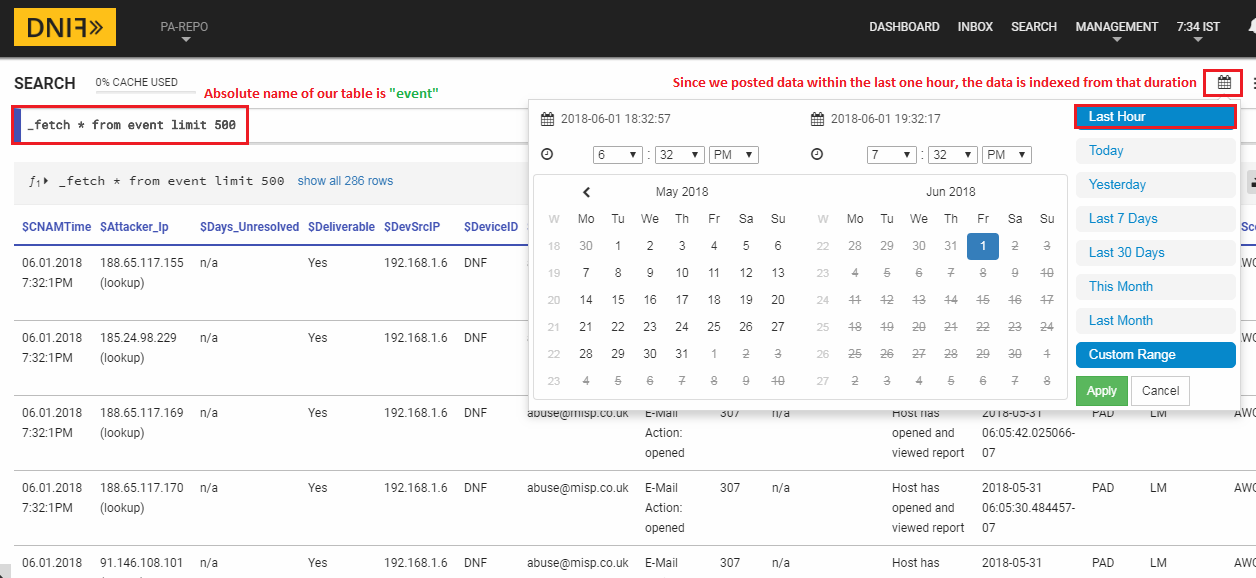
*Fig 14: Parameters in Postman*

* The request must be POST
* Adjacent to POST, put the address to post at: <http://YourUbuntuIP:9234/json/receive>
* After that, under “Body” -> “raw” paste your test JSON data.
* Then click on “SEND”



*Fig 15: Postman O/P*

***Mini Step 3: Check in the Web Console if data has arrived to DNIF***



*Fig 16: Posted Data O/P*

If in case the data is not visible on the DNIF console even after doing as instructed. Do a SSH to your DNIF container and restart the **“Events\_Parser\_Processor”** service. Then carry out ***Mini Step 2*** again, unless the data is visible on DNIF web console.

***Mini Step 4: The Python Code***

* Edit the python code script from repository to capture data in JSON, and throw it to DNIF API.
* Make sure **necessary changes** have been made to the code, like **your Ubuntu Static IP** & your local drive location where you are going to save JSON file storage.
* Pay attention to the **backslash/ forward slash** mechanism as they differ in Linux and Windows systems **when accessing directories**.
* Some code explanation is commented inside the code script itself.
* The code is available here:

<https://github.com/dnif/DigiVigi/blob/master/Process_2/PS_SourceToDnif.py>

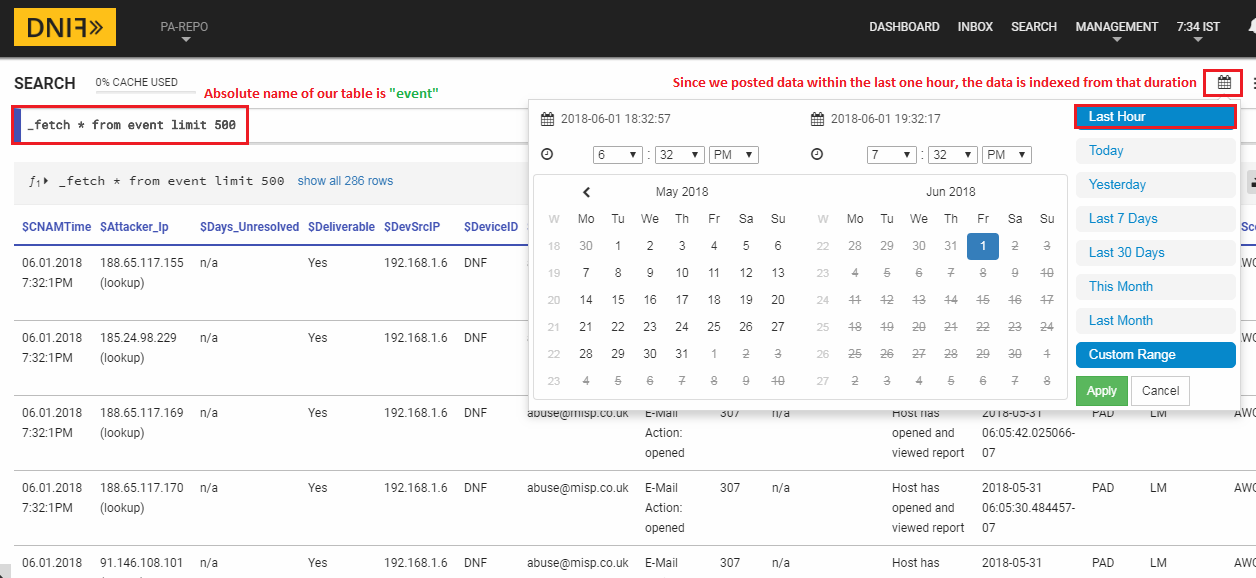


*Fig 17: Source To DNIF Code (All in One)*

***Mini Step 5: Running the Code***

* Run your script once in PyCharm like IDE and test it out.
* If script exits with process code 0, then your script runs fine.
* If any errors are thrown, one might have to debug it themselves. Type the error code on Google, and find out the solutions.
* Although some script error is highly unlikely, but could occur due to negligence.

***Mini Step 6: Check in the Web Console if data has arrived to DNIF***

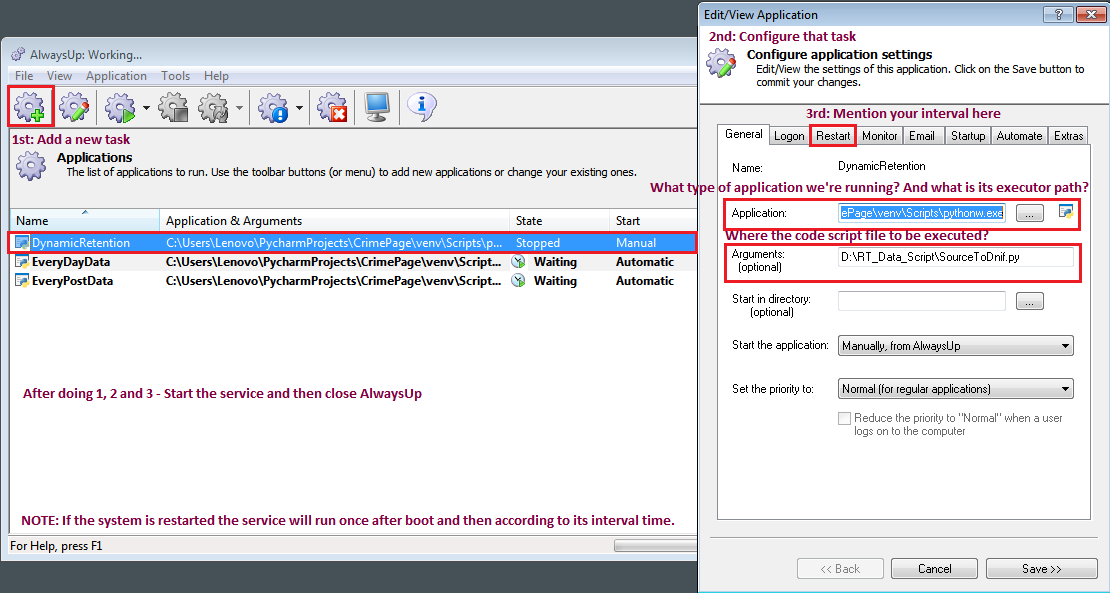


*Fig 18: Posted Data O/P*

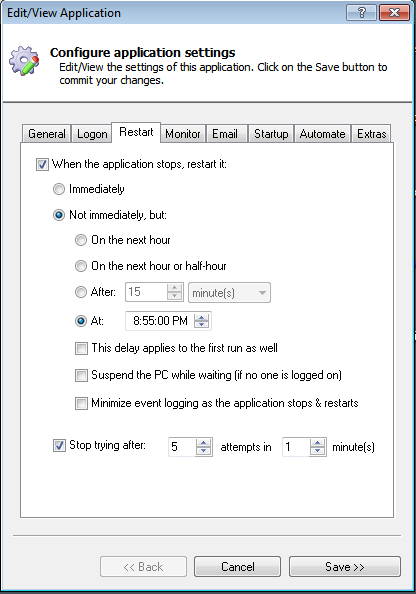
**STEP 5:**

## Automating Dynamic Data Feeding

Most of the complex work is done, and this is pretty much a one-time thing – that is, automating the script for a daily basis. For this we have used **“AlwaysUp”** Trial Version. It is reliable software, works very nice, highly regarded in the industry as well. If one is in commercial work, then they can pay upfront for prolonged usage, since the amount is very small.



*Fig 19: AlwaysUp Demo*



*Fig 20: Setting the interval*

Once everything is configured, all you’ll have to do is check on the Web Console after the timely interval – whether the recent data has been posted or not.

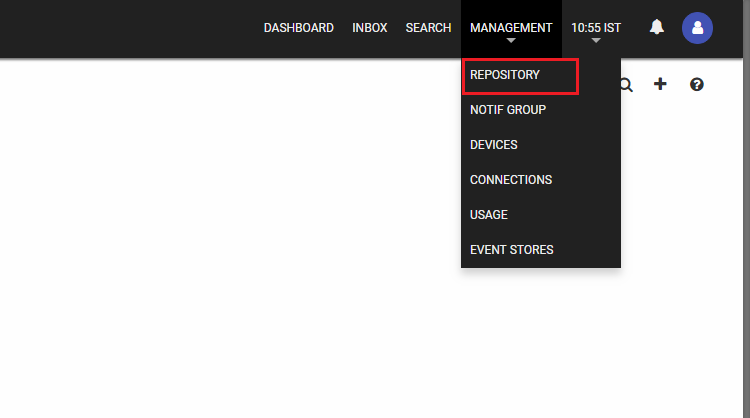
**STEP 6:**

## Creating Package

Before we create widgets and dashboard, we will first need to create a package. A package in DNIF has a larger purpose (Check out their documentation to know more)

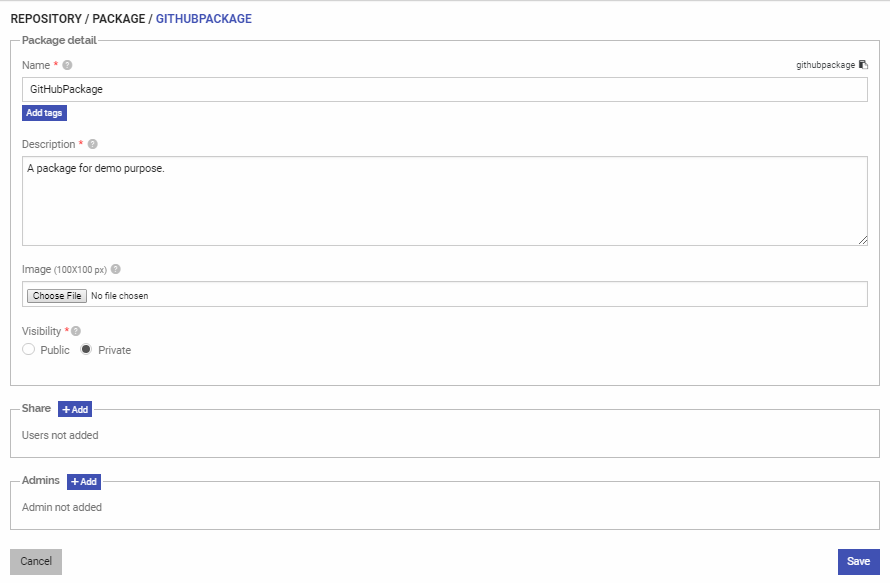
Here’s how we create a package in DNIF.

**1st:** Under management tab, click on “Repository”, to further create package. A package is like a box full of widgets, dashboards, reports and stuff. It’s like a collection.

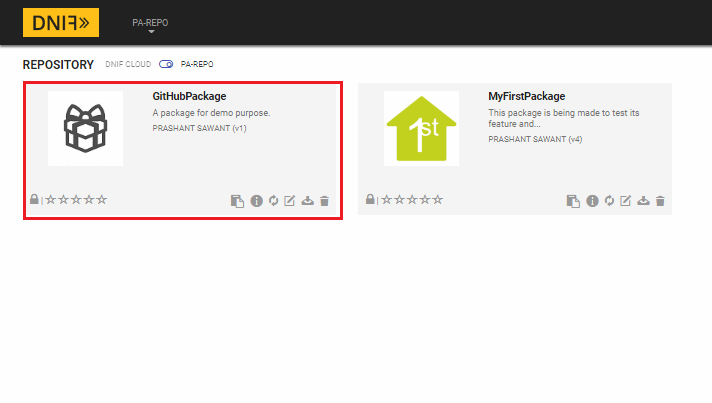


*Fig 21: Creating Package Navigation*

**2nd:** Once when under repository window, click on the plus symbol to add a new package. Add the necessary details which are fundamentally understood. When all is done – SAVE.



*Fig 22: Package Creation*

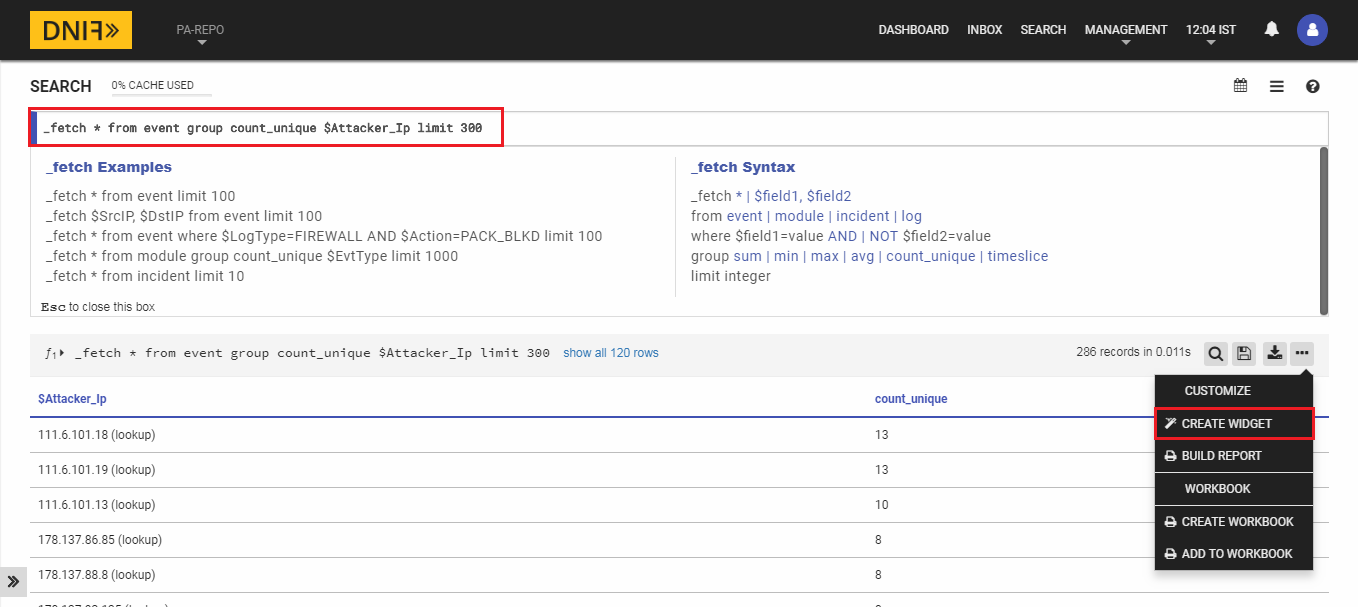


*Fig 23: Package Created*

**STEP 7:**

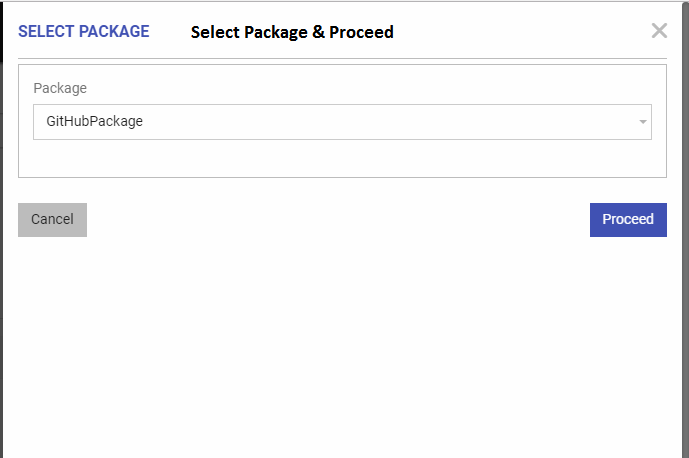
## Creating Widget(s)

**1st:** To create a widget, we first write a meaningful query which can then be visually addressed in a widget. So select Create Widget as shown in the image below.



*Fig 24: Creating Widget - Navigation*

Then you will be asked to select which package this widget will go under before actually creating the widget.

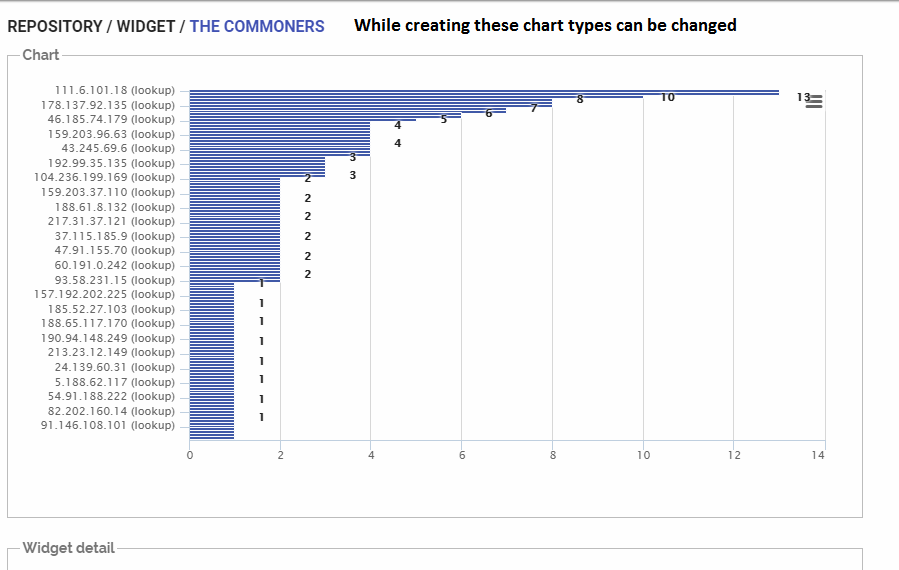


*Fig 25: Select Package*

**2nd:** Next we fill in fundamental details that are required to give meaning to our widget. Again, very detailed information on widgets is provided on DNIF’s website.



*Fig 26: Widget Information*



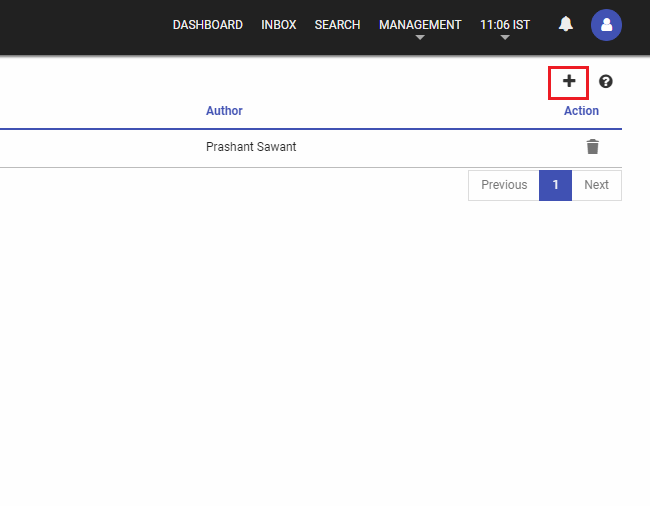
*Fig 27: Visuals*

**STEP 8:**

## Creating Dashboard

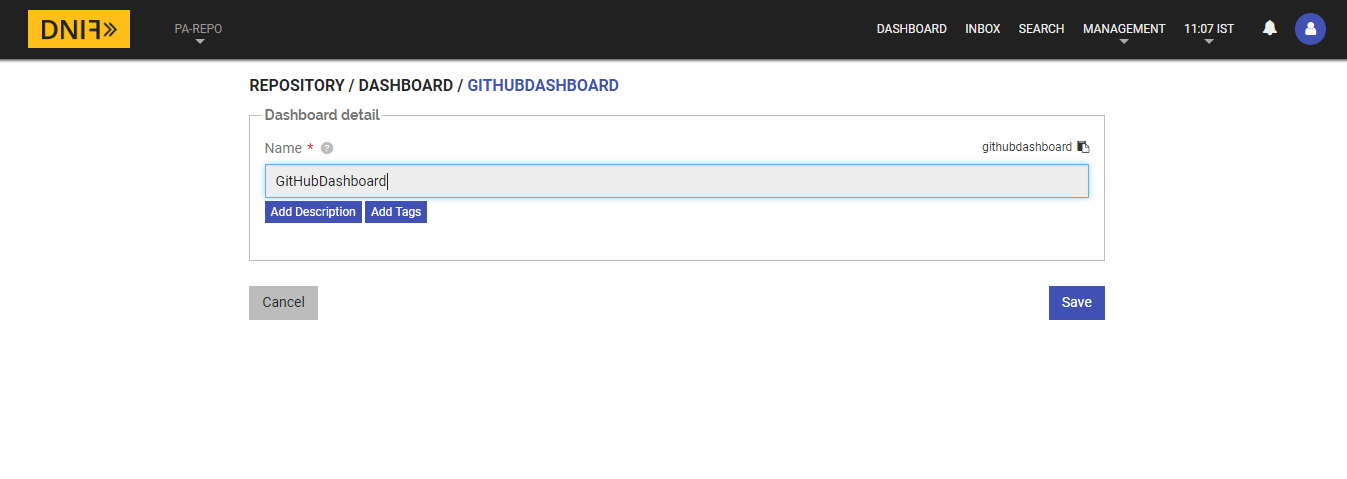
To use widgets, that are to showcase, we will need to put it on a dashboard. But before that we need a dashboard. So we create one.

**1st:** Under the “Dashboard” tab, click on the ‘+’ symbol. Then a package selection screen will appear. So select the appropriate package we have created earlier.



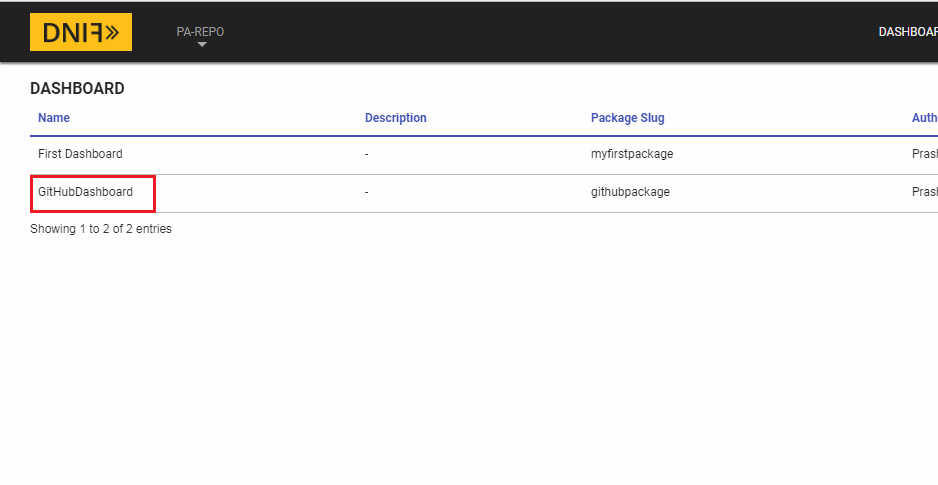
*Fig 28: Adding a Dashboard*

**2nd:** Give the dashboard a name of your choice.



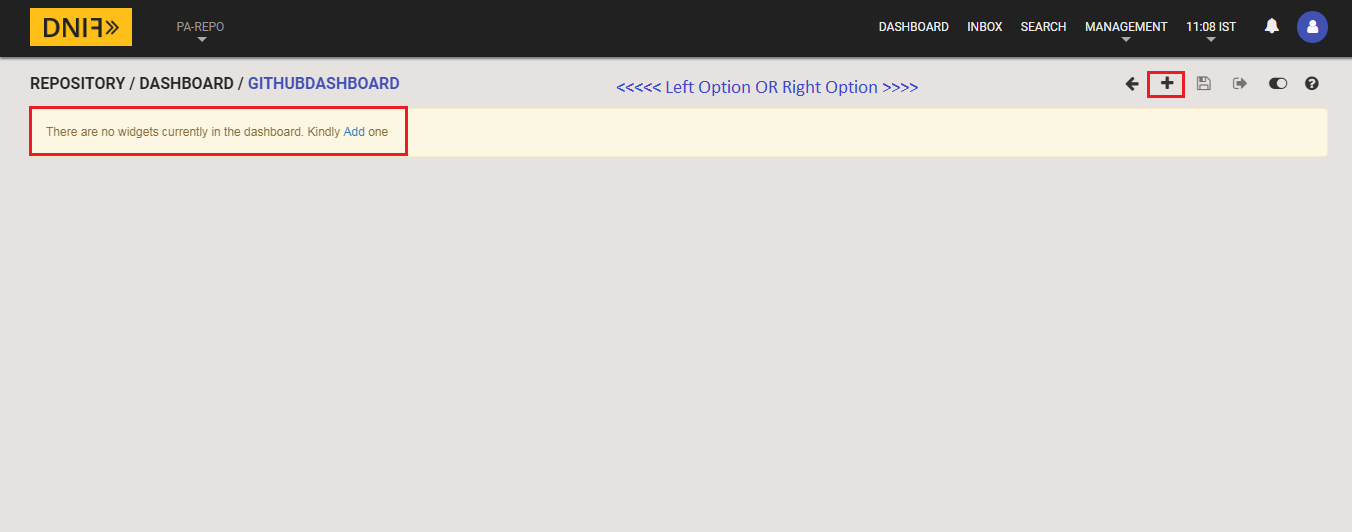
*Fig 29: Creating Dashboard*

**3rd:** Now that our dashboard is created let us put that widget we created earlier. So under dashboard page, click on the dashboard we just created.



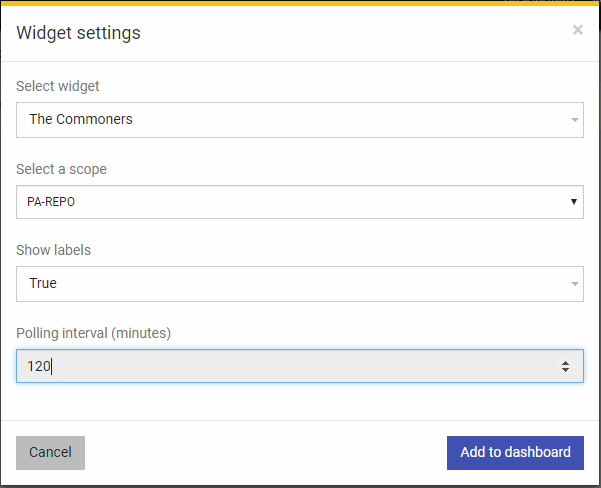
*Fig 30: Our Dashboard*

**4th:** Add a widget.

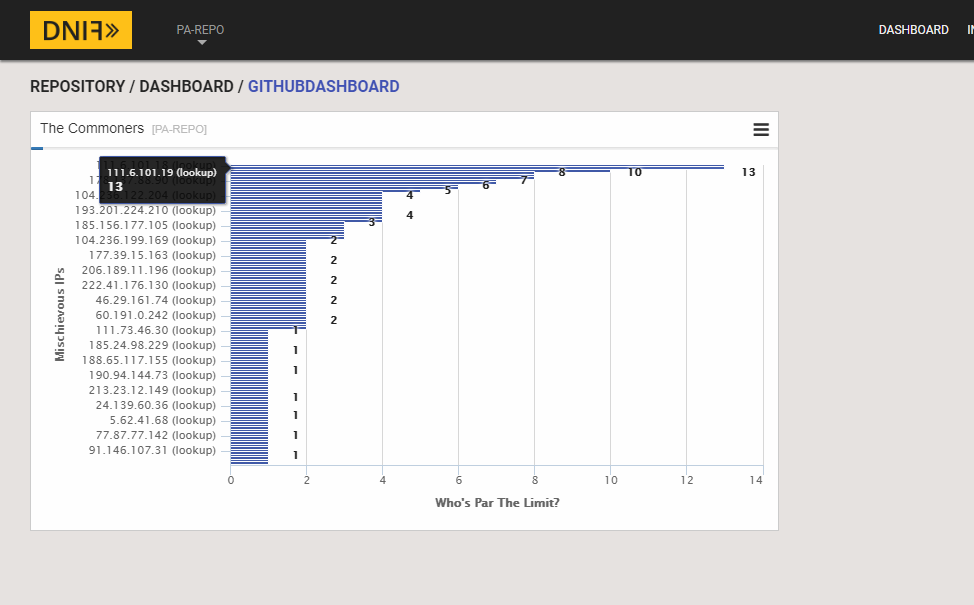


*Fig 31: Adding Widgets to Dashboard*

**5th:** After checking upon the widget settings, click on “Add to Dashboard” and your dashboard has now started to build up.



*Fig 32: Widget to Dashboard Settings*



*Fig 33: Dashboard Created*

**STEP 9:**

## Configuring SMTP for DNIF

Configuring SMTP is not a tricky thing at all. All you need to do is follow some simple steps which are documented separately under the “dnif/DigiVigi” repository. We’re not adding the document content here to avoid duplicity and avoid the already lengthy document.

Here’s the link to that document:

<https://github.com/dnif/DigiVigi/blob/master/Guiding_Documents/6_The_SMTP_Config_Guide.docx>

**STEP 10:**

## Creating Alert(s)

Creating alerts means you want notifications if certain business/ security rule is violated or if you want some daily reporting mechanism. And this can be done inside DNIF as well. This is possible only after SMTP is configured and run on DNIF, which is a one-time thing.

Here are some heads up links one should consider going through. They will help.

**Configuring SMTP:**

<https://dnif.it/docs/guides/tutorials/configuring-smtp-in-docker.html>

Or one can always refer previous step.

**How to create reports in DNIF:**

<https://dnif.it/docs/guides/tutorials/how-to-create-reports.html>

There some detailed examples given on this page which may or may not be covered under this exercise.

**Hot to create notification groups:**

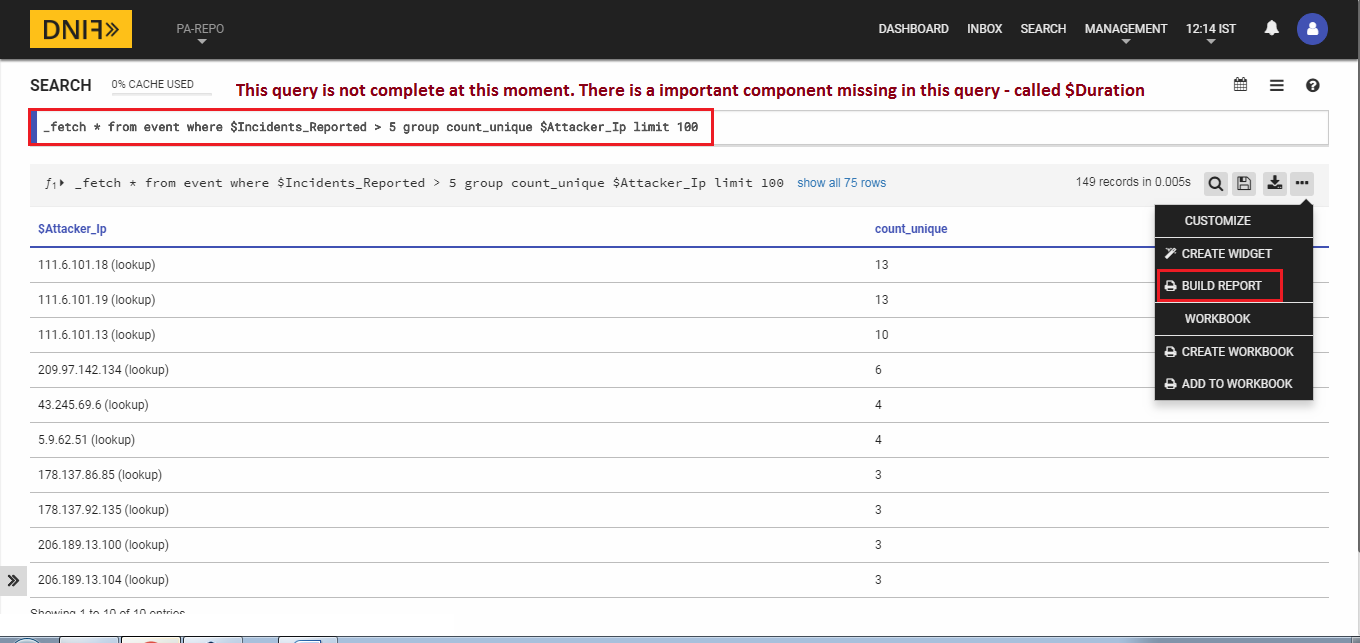
<https://dnif.it/docs/guides/tutorials/create-and-view-notif-group.html>

**How to raise alerts:**

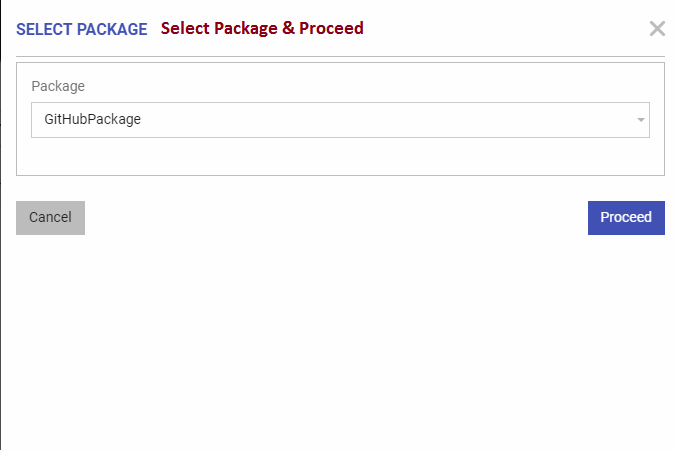
<https://dnif.it/docs/learn/DQL/trigger.html>

There some detailed examples given on this page which may or may not be covered under this exercise.

**1st:** Let us build a report using our previous widget query.



*Fig 34: Build Report Option*

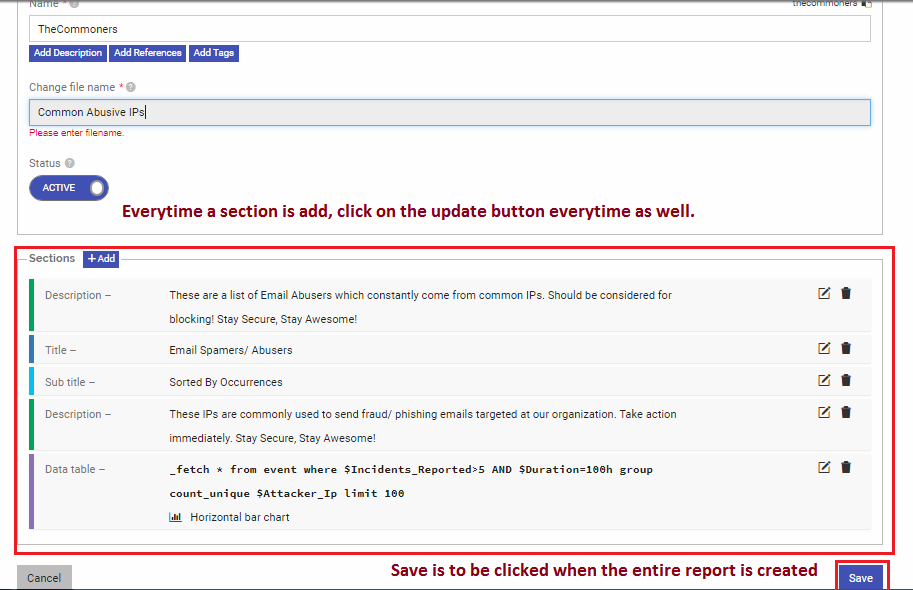


*Fig 35: Select Package*

**2nd:** After selecting the package, under section type – query, check if your query for report is valid or not (We’re using the same query from the widget here). In order to check/ validate your query, click on execute (the execute button is for validation only). And only after the query is fine, will the “Add column” & “Add chart” buttons will activate.

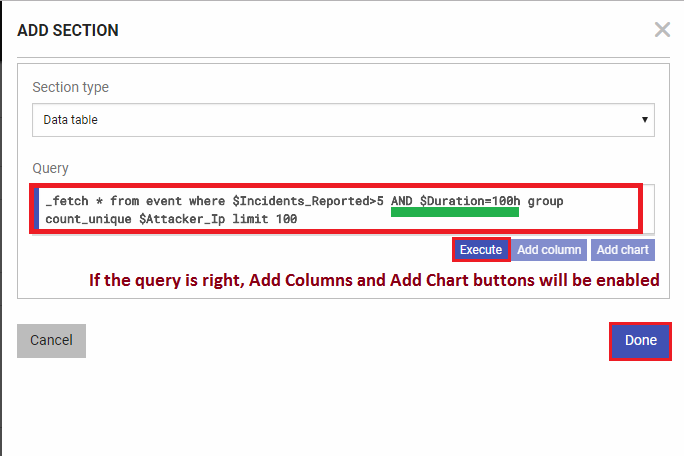
**NOTE 1:** There is other “**Section** **Type**” as well – like Title, Sub-title, Description, etc.

**NOTE 2:** The report query should have “$**Duration**” in the where clause. It is a necessary parameter.

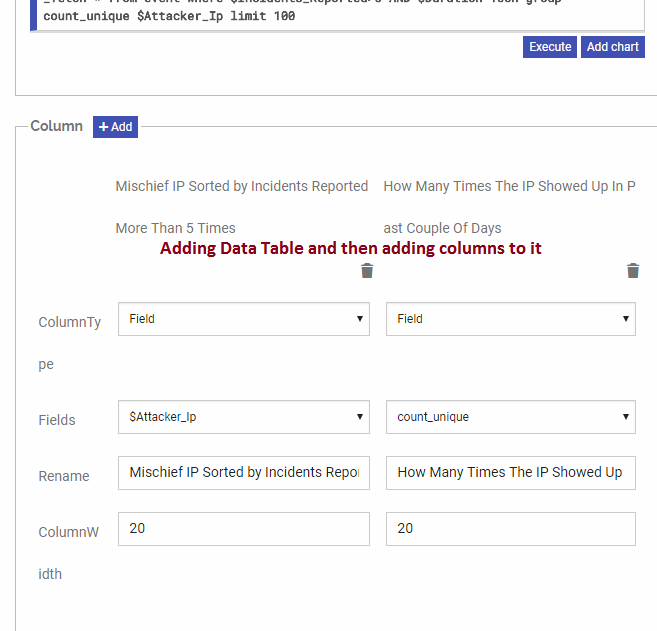


*Fig 36: Report Build Section*

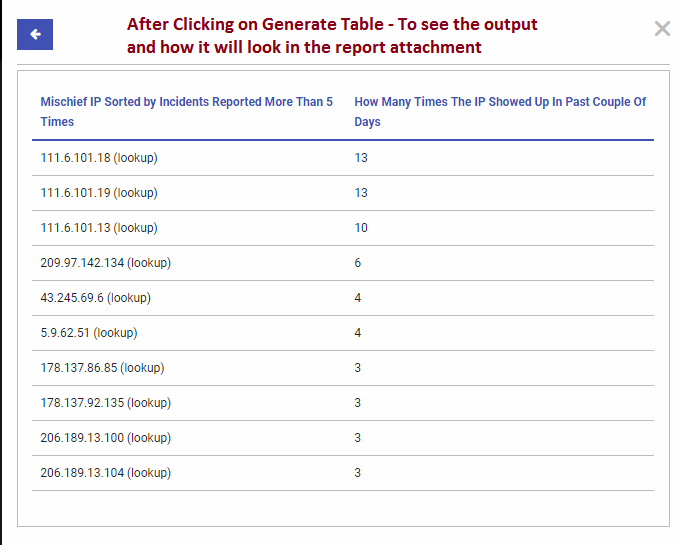
* Above is how a complete report details page should look like. Fill in the general details and let us start adding sections to our report.



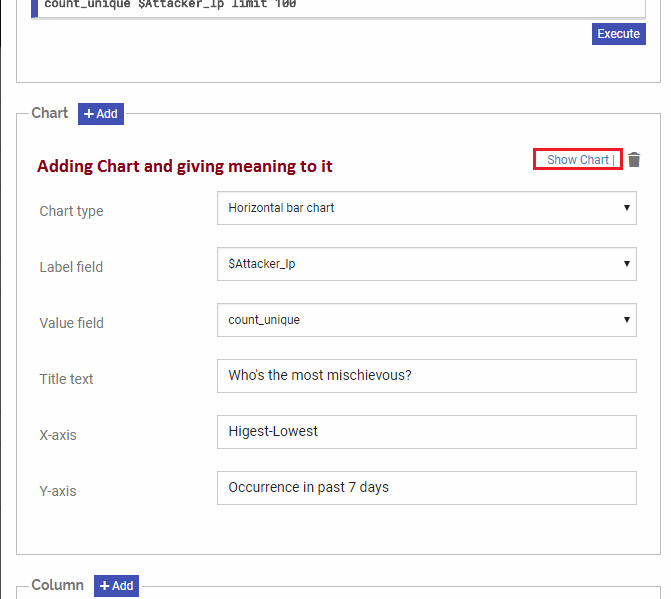
*Fig 37: Executing Report Query*



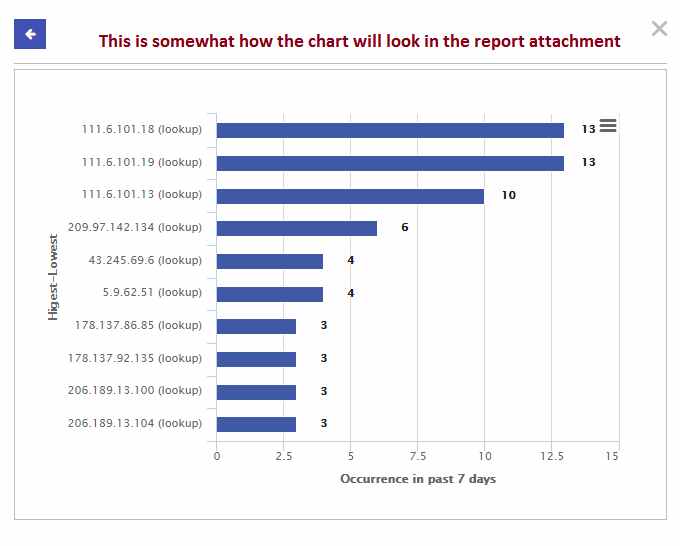
*Fig 38: Add Columns to Data Table Section*



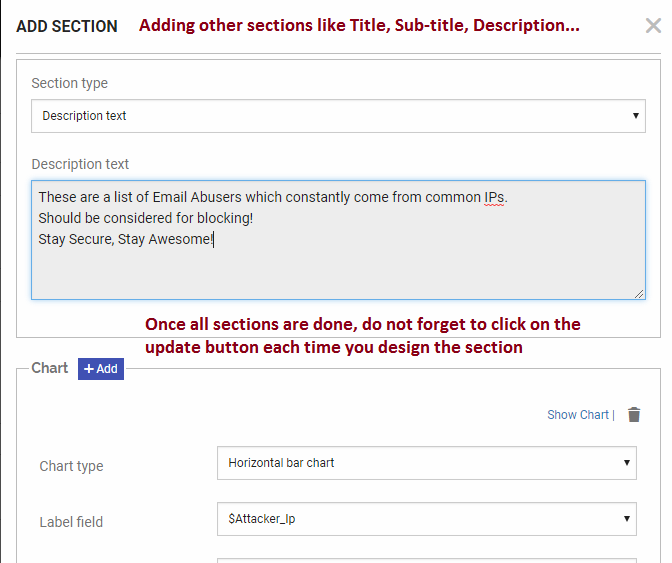
*Fig 39: After Clicking Generate Table Button*



*Fig 40: Adding Charts*



*Fig 41: How Report Visuals will look*

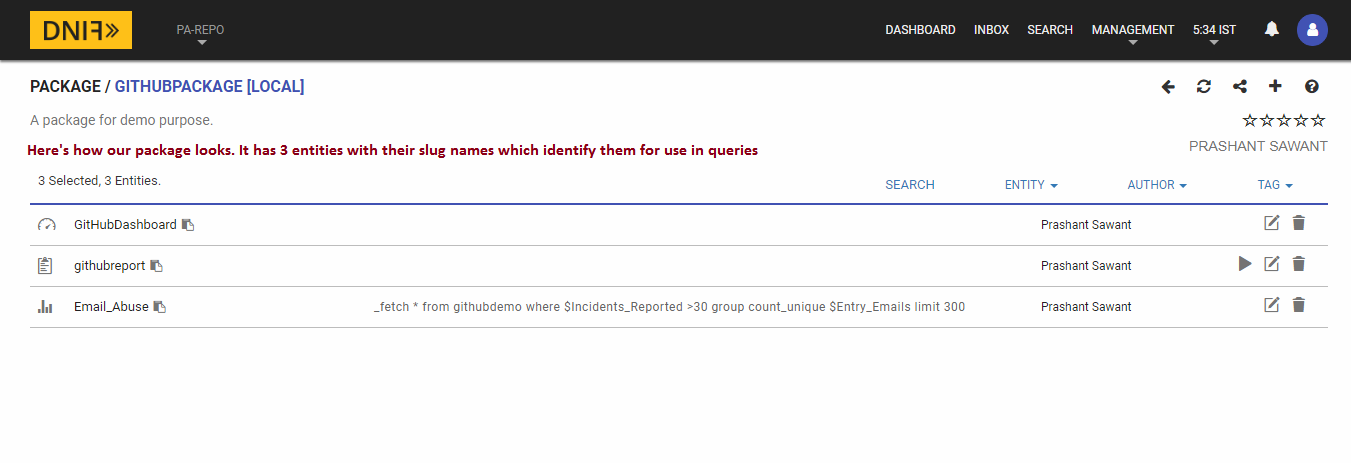


*Fig 42: Other Section Example*

* Once all the section information is provided, click on “Update” button after every section addition.
* After all sections have been added, save the report which has been created. This will save the report which we have made under our package.

**3rd:** Next we will have to check if all those contents we created are saved properly or not.

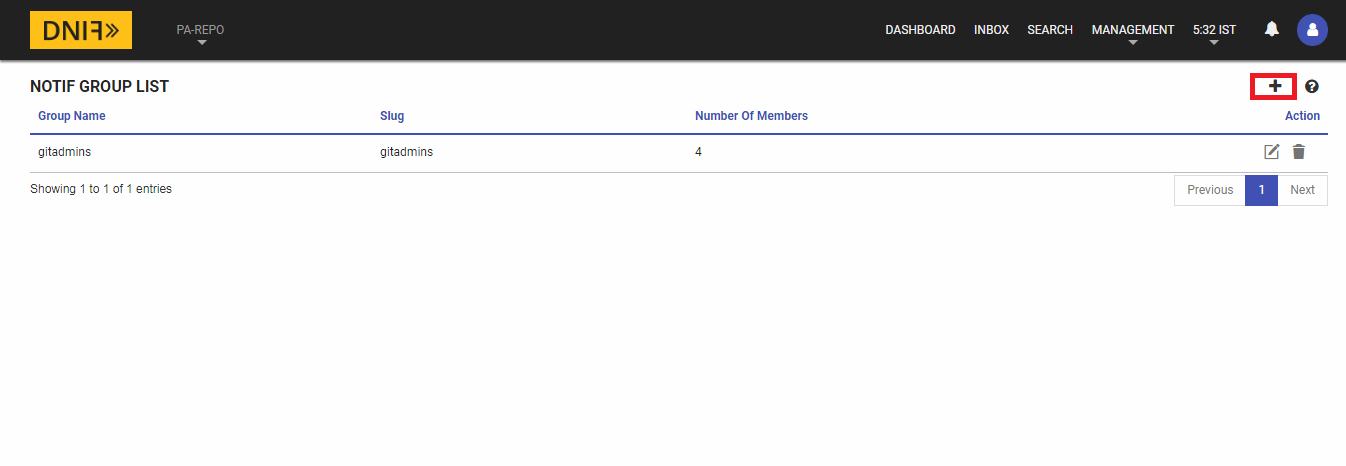
* Under “Repository Tab” where our package is – view that package and click on editing button (one with pencil symbol)
* So one can also add a report from under package.



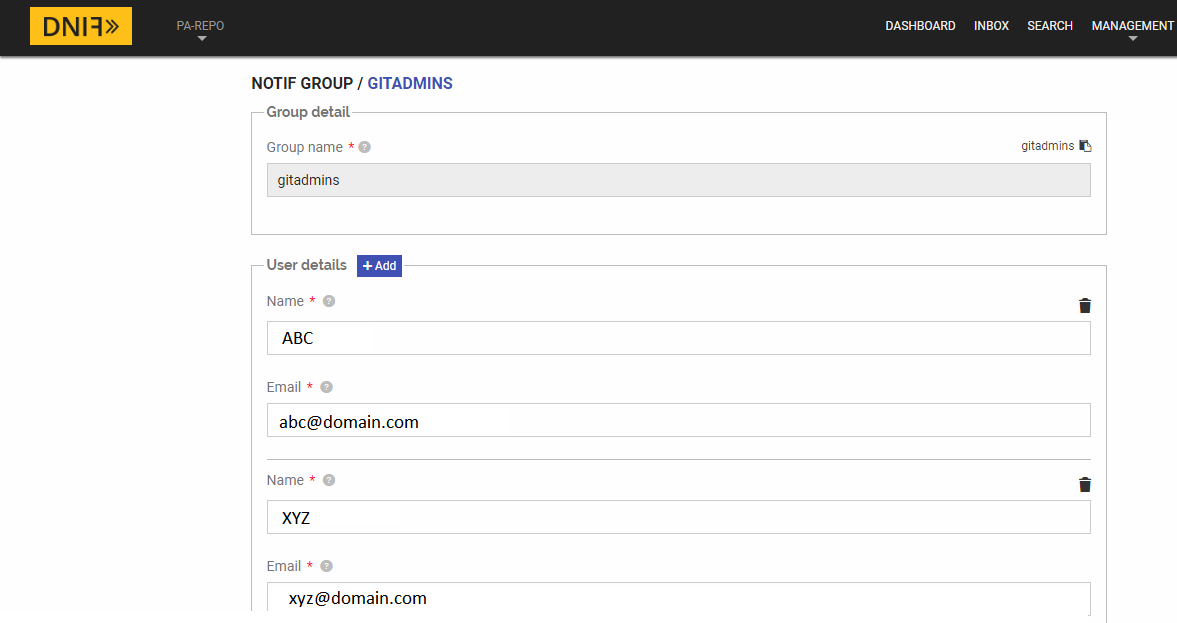
*Fig 43: Adding report option 2*

**4th:** Sending alerts/ reports to email users.

* One can send email alert/ report to individual email address or to a notification group – group of email addresses.
* Although the demonstrated example is one which sends email alert to a single user. But let us learn to create a notification group as well.
* Under Management > Notif

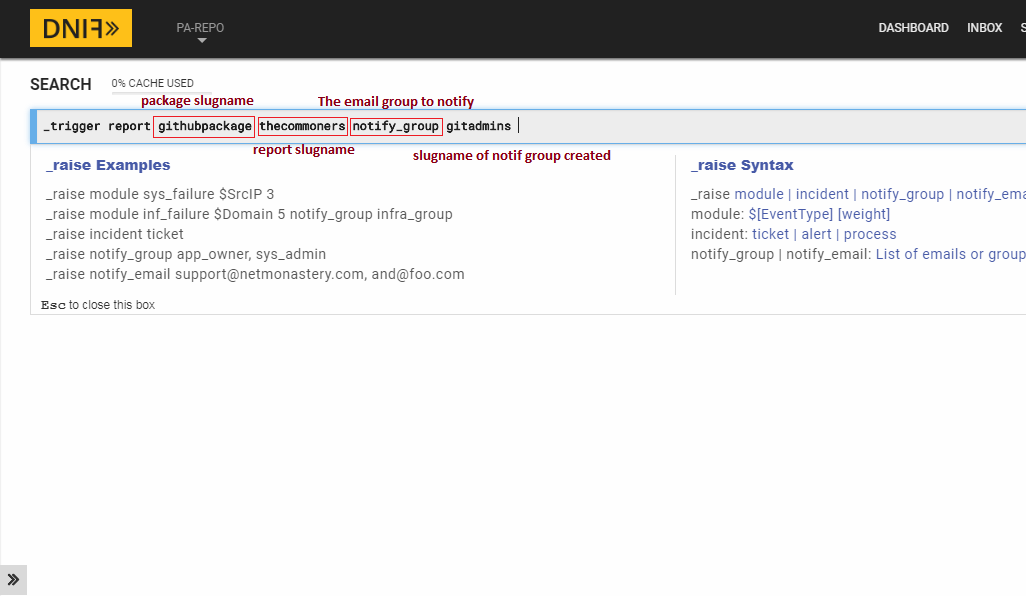


*Fig 44: Notification Groups Page*



*Fig 45: Creating Notification Group*

**5th:** Running our first trigger directive query.



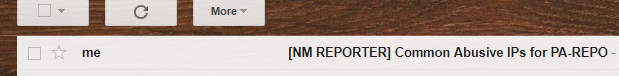
*Fig 46: Running the trigger*

Once this trigger query is run, all we have to do is check for that email.

This trigger can also be automated by putting it inside \_raise directive.

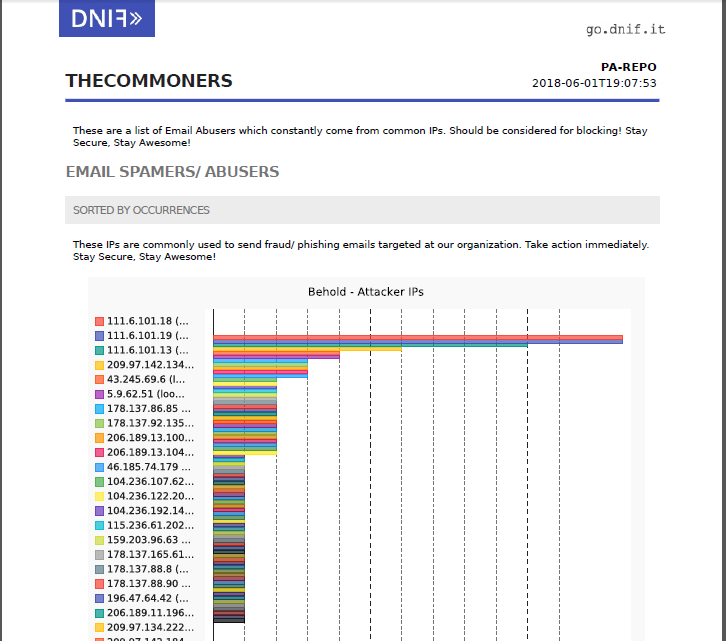
To know about \_raise, visit DNIF Website Documentations page.

**6th:** Email Alert Generated



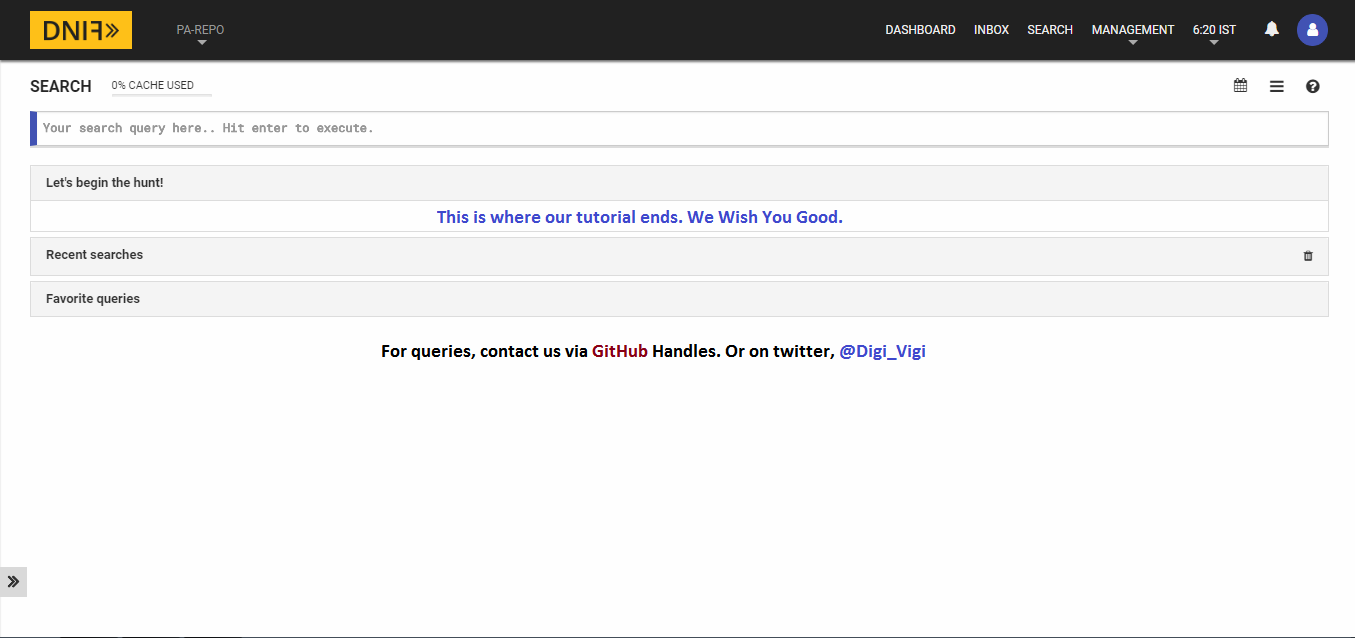
*Fig 47: Email Generation*

**7th:** Our report, which comes as an attachment to the email above.



*Fig 48: PDF Report Generated*

**PROCESS 2 COMPLETE:**

****

*Fig 49: End Credits*

**END NOTE:**

The DNIF documentation guide, the guideline on our repository, & DNIF Google Group – dnifHQ is all the help one would need to complete this process. Thank You.