

Platys – Provisioning of modern data platforms simplified!

Guido Schmutz, Ulises Fasoli

 gschmutz

 <http://guidoschmutz.wordpress.com>

BASEL | BERN | BRUGG | BUKAREST | DÜSSELDORF | FRANKFURT A.M. | FREIBURG I.B.R. | GENÈVE
HAMBURG | KOPENHAGEN | LAUSANNE | MANNHEIM | MÜNCHEN | STUTTGART | WIEN | ZÜRICH

trivadis

Agenda

- Motivation for Platys
- Installing Platys
- Platys in Action

trivadis

Motivation for Platys?

Requirements for a Platform for Lab, PoC and PoV projects



For Labs, PoC and PoV projects as well as trainings we wanted a platform

1. which is easy to deploy
2. supports modular, extensible and configurable set of services
3. where new services can be added quickly and tested as part of an existing platform (set of services)
4. where no infrastructure specialist is needed for provisioning
5. lightweight, low footprint (resources needed)
6. simple, lightweight transportation
7. runs on-premises and in the cloud

Platform for Lab, PoC and PoV projects



		Bare Metal	VM	Cloud PaaS	Docker & Docker Compose
1	Easy Deployment	+	++	+++	+++
2	Modular, extensible	++	++	++	+++
3	Add new services quickly	+	+	++ (1)	+++
4	No infrastructure specialist needed	+	++	+++	+++
5	Lightweight footprint	+	++	+	+++
6	Simple, lightweight transportation	-	+	++ (2)	+++
7	On prem & cloud	+	+++	++	+++ (3)

- 1) If exists as PaaS
- 2) Within same cloud provider
- 3) Runs anywhere Docker & Docker Compose runs

What is Docker, what is Docker Compose?



Docker – running a single service

```
docker run -d -p 18630:18630 -e SDC_CONF_MONITOR_MEMORY:true -v sdc-data:/data  
--name streamsets-dc streamsets/datacollector:3.15.0
```

Docker Compose – running a set of services belonging together

```
docker-compose up -d
```

```
streamsets-1:  
  image: streamsets/datacollector:3.15.0  
  ports:  
    - 18630:18630  
  environment:  
    SDC_CONF_MONITOR_MEMORY: 'true'  
    SDC_CONF_PIPELINE_MAX_RUNNERS_COUNT: 50  
  volumes:  
    - ./sdc-data:/data  
kafka-1:  
  image: confluentinc/cp-enterprise-kafka:5.5.0  
  depends_on:  
    - zookeeper-1  
  ports:  
    - 9092:9092  
  ...
```

platys – Trivadis Platform in a Box



- open-source toolset, developed by the Trivadis Platform Factory
 - help provisioning Modern Data Platforms based on [Docker](#) and [Docker Compose](#)
 - its main use is for small-scale Data Lab projects, Proof-of-Concepts (PoC) or Proof-of-value (PoV) projects as well as trainings
-
- See [here](#) for how to install platys
 - <https://github.com/TrivadisPF/platys>

Trivadis platys – Platform in a Box



Platform Stack

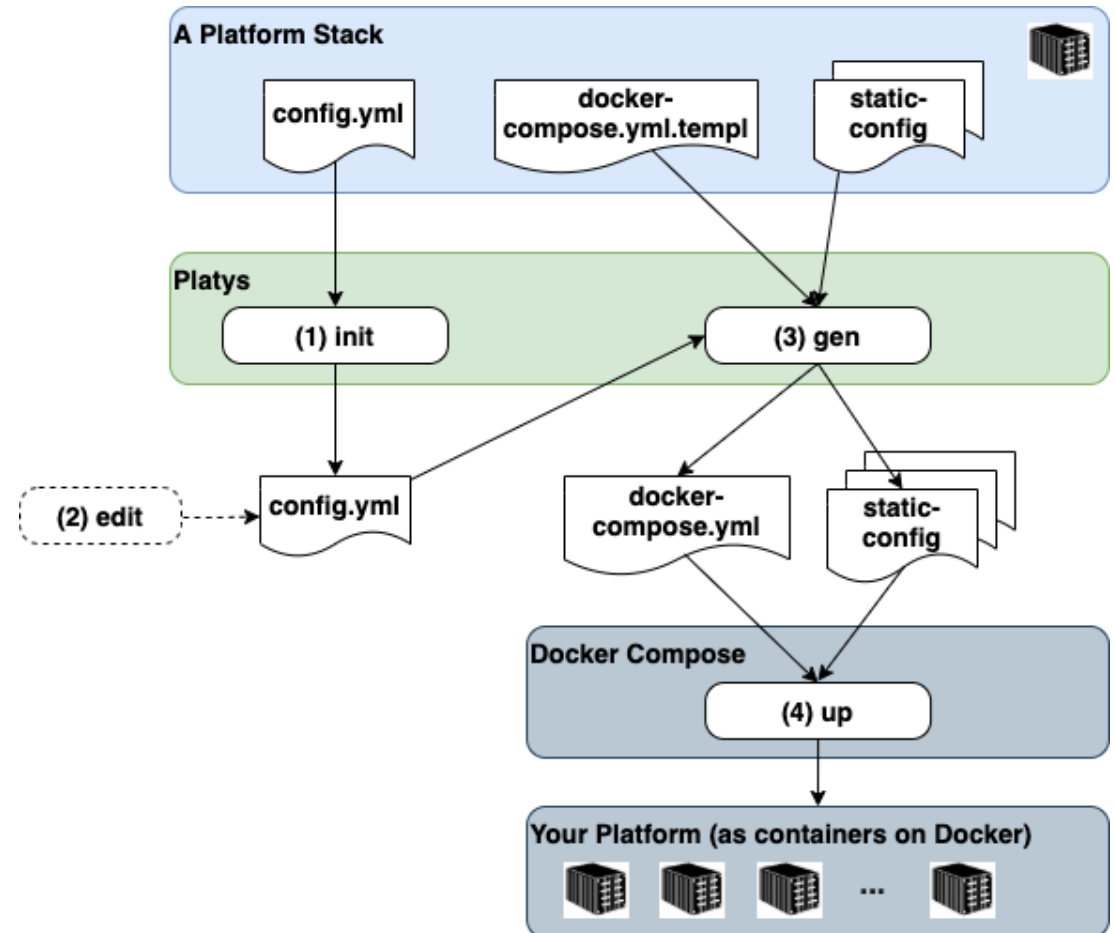
- A given set of services to choose from
- A generator for generating a set of these services as a Platform

Platform

- An instance of a set of services generated based on a Platform Stack
- Fully runnable platform based on Docker Compose and Docker

Platys

- The CLI tool to control the Platform Stack



Installing Platys

Trivadis platys – How to install? (I)



Platys runs on **Linux**, **Mac OS-X** and **Windows**

[Installation](#) is comparable to installing docker-compose, on Ubuntu perform the following steps

1. Download the binary

```
sudo curl -L  
"https://github.com/TrivadisPF/platys/releases/download/2.4.0/p  
latys_2.4.0_linux_x86_64.tar.gz" -o /tmp/platys.tar.gz
```

2. Untar and move it to /usr/local/bin

```
cd /tmp  
tar zvxf /tmp/platys.tar.gz  
sudo mv platys /usr/local/bin/  
sudo chown root:root /usr/local/bin/platys  
sudo rm /tmp/platys.tar.gz
```

Trivadis platys – How to install? (II)



3. Download the binary

```
sudo ln -s /usr/local/lib/platys.dist/platys /usr/local/bin/platys
```

4. Use the `--version` option to check that the generator has been installed successfully

```
$ platys version  
Platys - Trivadis Platform in a Box - v 2.4.0  
https://github.com/trivadis/pf/platys  
Copyright (c) 2018-2020, Trivadis AG
```

trivadis

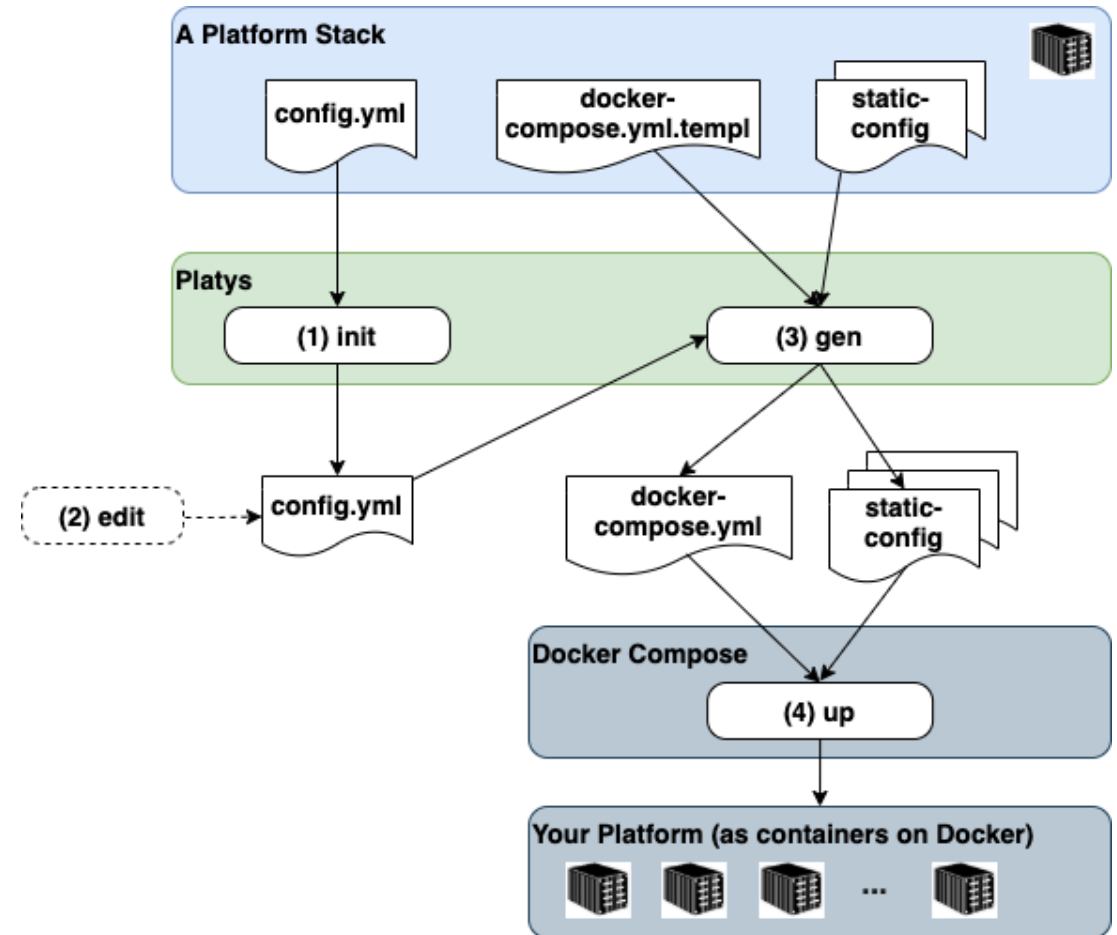
Platys in Action

Trivadis platys – Platform in a Box



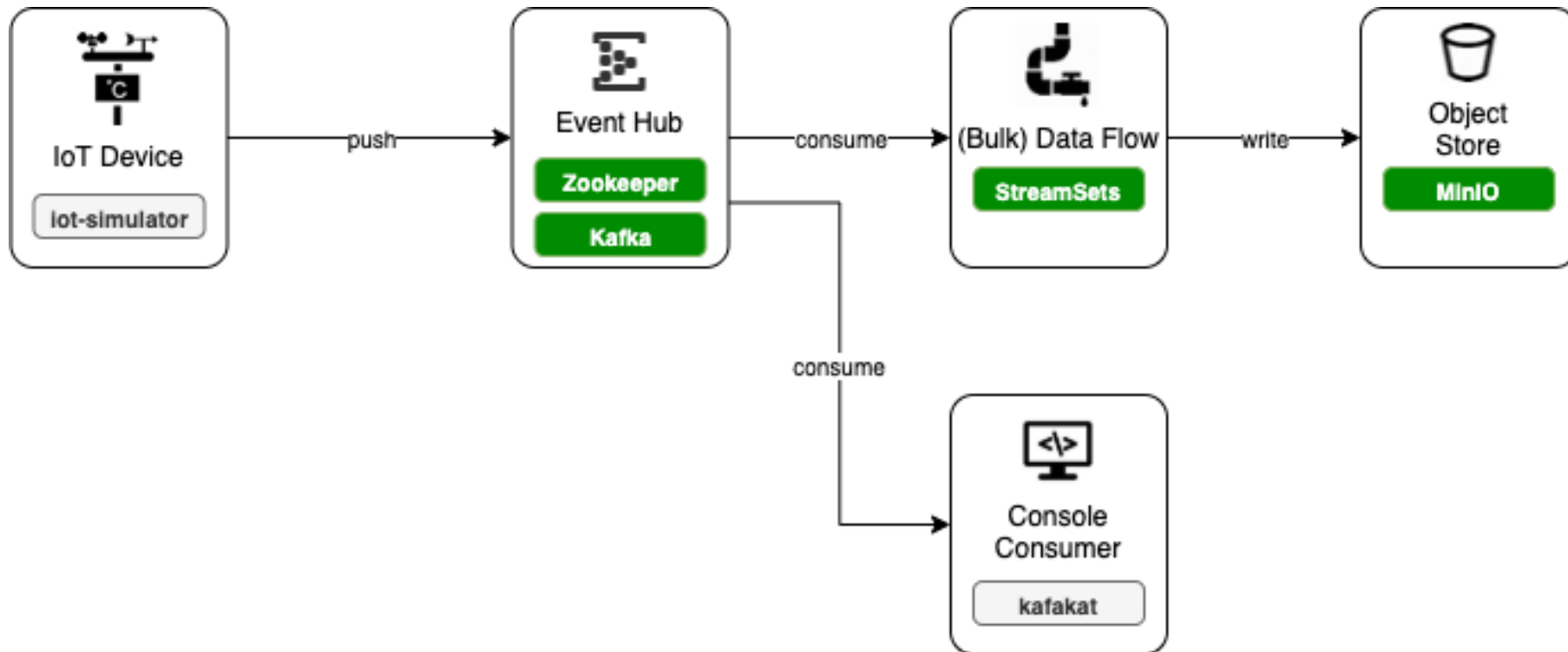
Steps for working with platys

1. Init a platform based on a given platform stack
2. Edit the config.yml files to enable services needed
3. Generate the docker-compose platform
4. Start the platform



Trivadis platys – How does it work? (I)

Let's see Platys in Action by generating a simple Platform with the following service



Trivadis platys – How does it work? (II)



1. Create a folder as your platform working directory

```
mkdir platys-demo  
cd platys-demo
```

2. Initialize the platform using the `platys init` command

```
platys init -n platys-demo-platform  
            -s trivadis/platys-modern-data-platform -w 1.7.0
```

3. A `config.yml` file is created locally, holding all the possible options of the modern-data-platform 1.7.0 stack. See [here](#) for a documentation of the various options.

Trivadis platys – How does it work? (III)

4. Use an editor to set the needed services to `true`.

```
# Default values for the generator
# this file can be used as a template for a custom configuration
# or to know about the different variables available for the generator
platys:
  platform-name: 'platys-demo'
  stack-image-name: 'trivadis/platys-modern-data-platform'
  stack-image-version: '1.7.0'
  structure: 'flat'

# ===== Apache Zookeeper =====
KAFKA_enable: true
# one of enterprise, community
KAFKA_edition: 'community'
KAFKA_volume_map_data: false
KAFKA_broker_nodes: 3
KAFKA_delete_topic_enable: false
KAFKA_auto_create_topics_enable: false
```


Trivadis platys – How does it work? (IV)



5. Run `platys gen` to generate your platform

```
platys gen
```

6. All the necessary artefacts for the Docker Compose platform have been generated and are ready to be started
7. Start the platform using `docker-compose up`

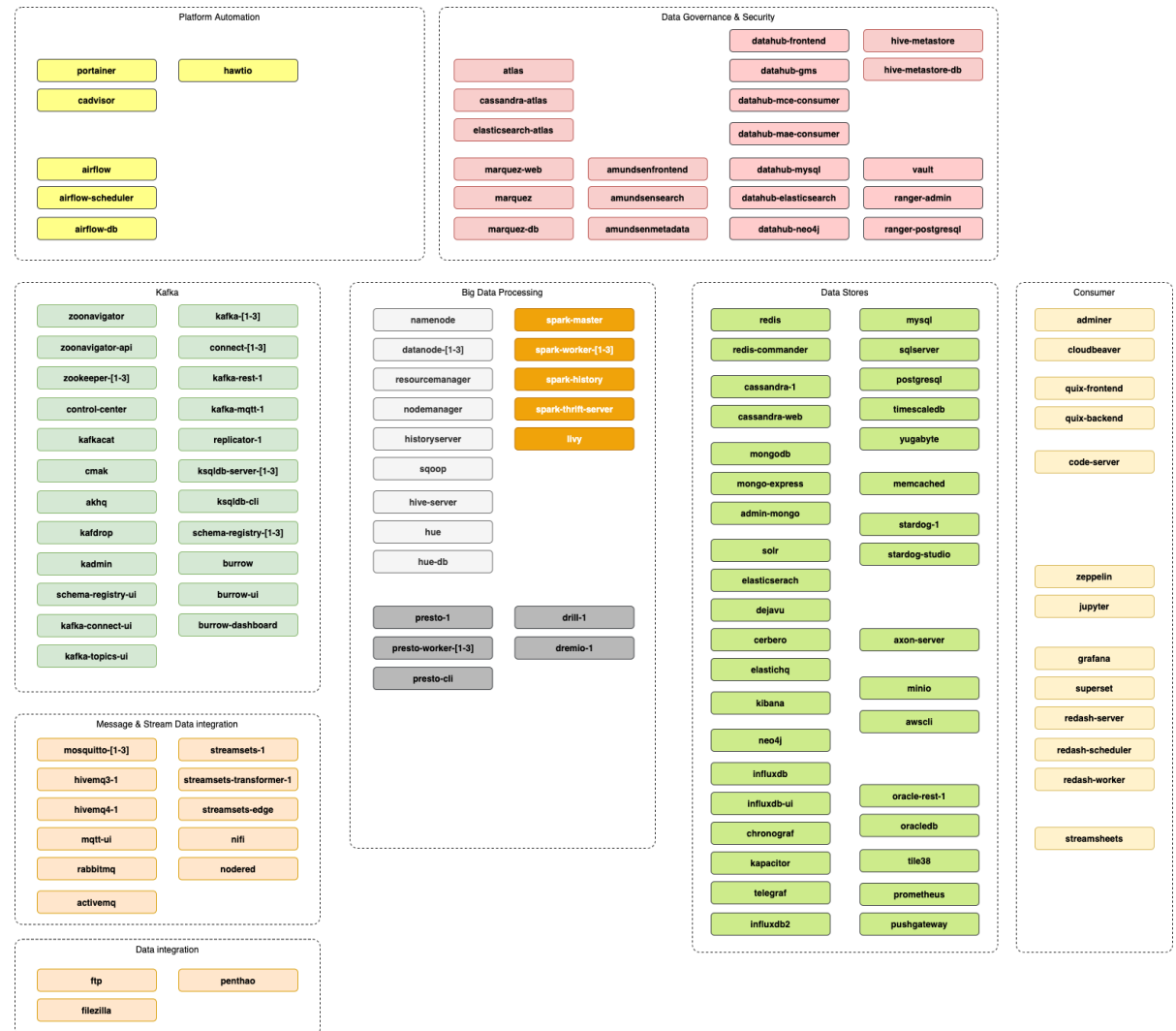
```
docker-compose up -d
```

modern-data-platform V1.7

trivadis

These are the services the current platform supports

Check the [Configuration](#) documentation for how to use



What else?

- `docker-compose.override.yml` for adding / re-configuring things not (yet) covered by platys
- `-s` flag for directly creating a valid `config.yml`
- [Cookbooks](#) will be made available showing various aspects of the modern data platform
- Multiple platforms in one box ;-) => ARM and x86-64
- other platform can be added by replicating what has been done for the modern data platform (docker container)
- [Slack channel](#) for questions and feedback
- We are hiring :-)

Structure: Flat vs. Subfolder

Flat

```
platys-demo
├── config.yml
├── docker-compose.yml
├── conf
├── container-volume
├── data-transfer
├── init
├── plugins
└── scripts
```

Subfolder

```
platys-demo
├── config.yml
└── my-platform
    ├── conf
    ├── container-volume
    ├── data-transfer
    ├── init
    ├── plugins
    ├── scripts
    └── docker-compose.yml
```

Trivadis platys – Tips and Tricks



- In the init you can also pass a list of services to directly set to true to omit having to edit the `config.yml`

```
platys init -n platys-demo-platform  
            -s trivadis/platys-modern-data-platform -w 1.7.0  
            -f -y KAFKA,SPARK,JUPYTER
```

- You can list the available service of a given platform stack by using the `list_services` command

```
platys list_services -s trivadis/platys-modern-data-platform  
                    -w 1.7.0
```



Making a **WORLD** possible
in which **intelligent IT**
facilitates **LIFE and WORK** as a
matter of course.