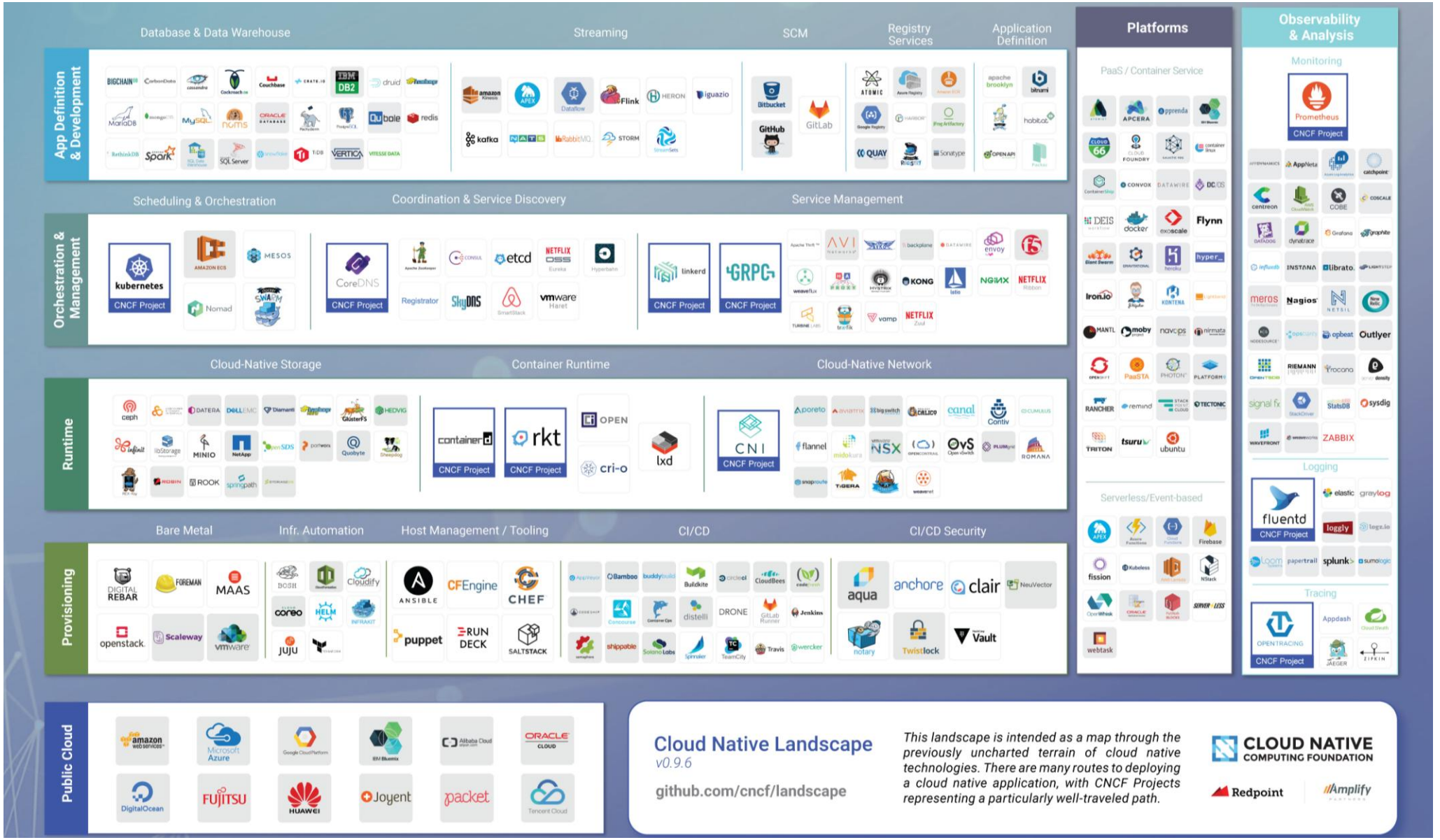
Docker-compose, vagga і інші «страшні» слова

bit.ly/msp-docker

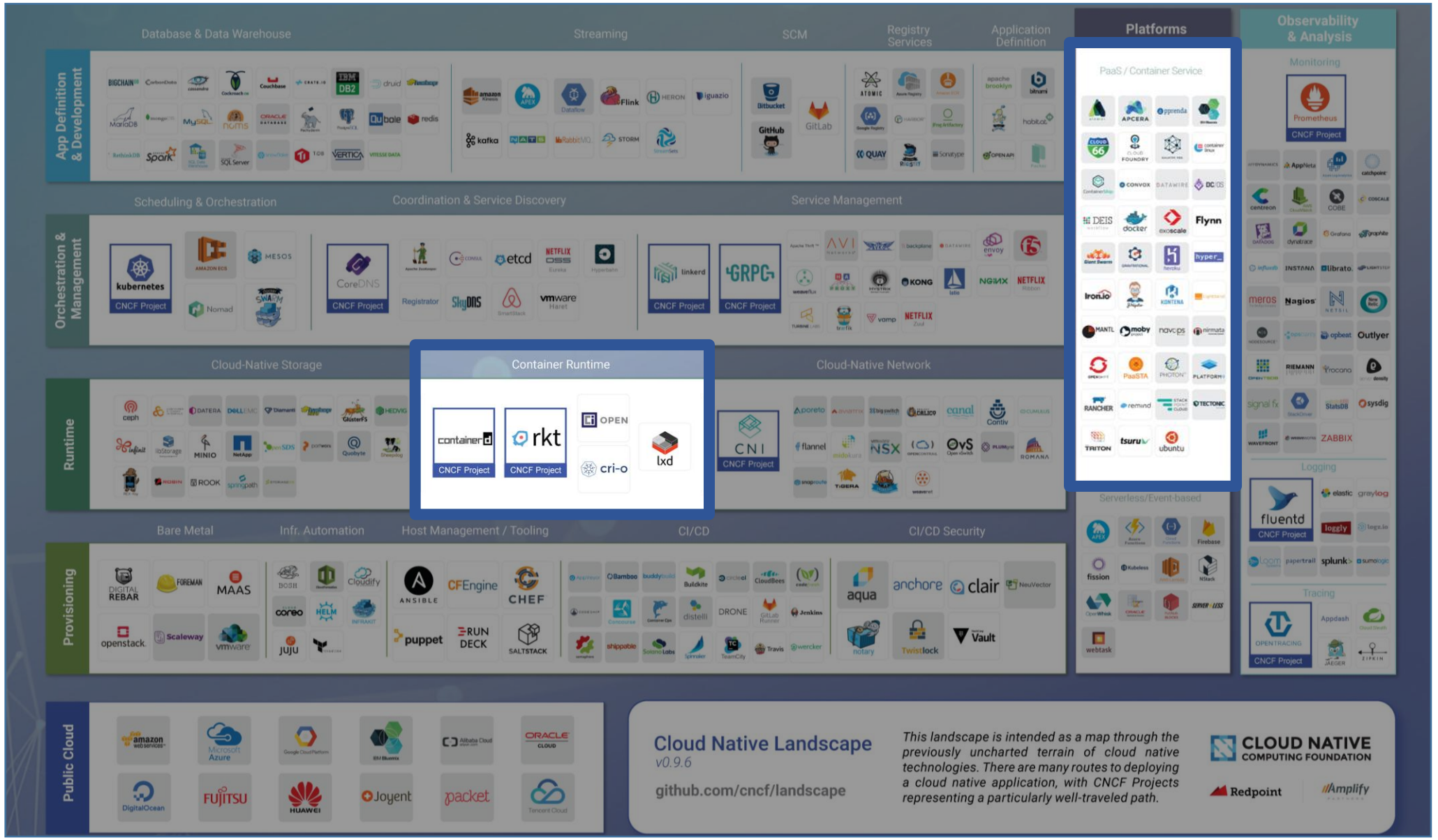
About me

- Work as DevOps Engineer at PDFfiller
- IT KPI (co-founder)
- Linux & Open Source enthusiast
- In love with JavaScript & Python
- Write on Bash and PHP

So many…


App Definition & Development
Database & Data Warehouse
Streaming
SCM
Registry Services
Application Definition
Orchestration & Management
Scheduling & Orchestration
Coordination & Service Discovery
Service Management
Runtime
Cloud-Native Storage
Container Runtime
Cloud-Native Network
Provisioning
Bare Metal
Infr. Automation
Host Management / Tooling
CI/CD
CI/CD Security
Public Cloud
Platforms
PaaS / Container Service
Serverless/Event-based
Observability & Analysis
Monitoring
Logging
Tracing
Cloud Native Landscape
v0.9.6
github.com/cncf/landscape
This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.
CLOUD NATIVE COMPUTING FOUNDATION
Redpoint
Amplify

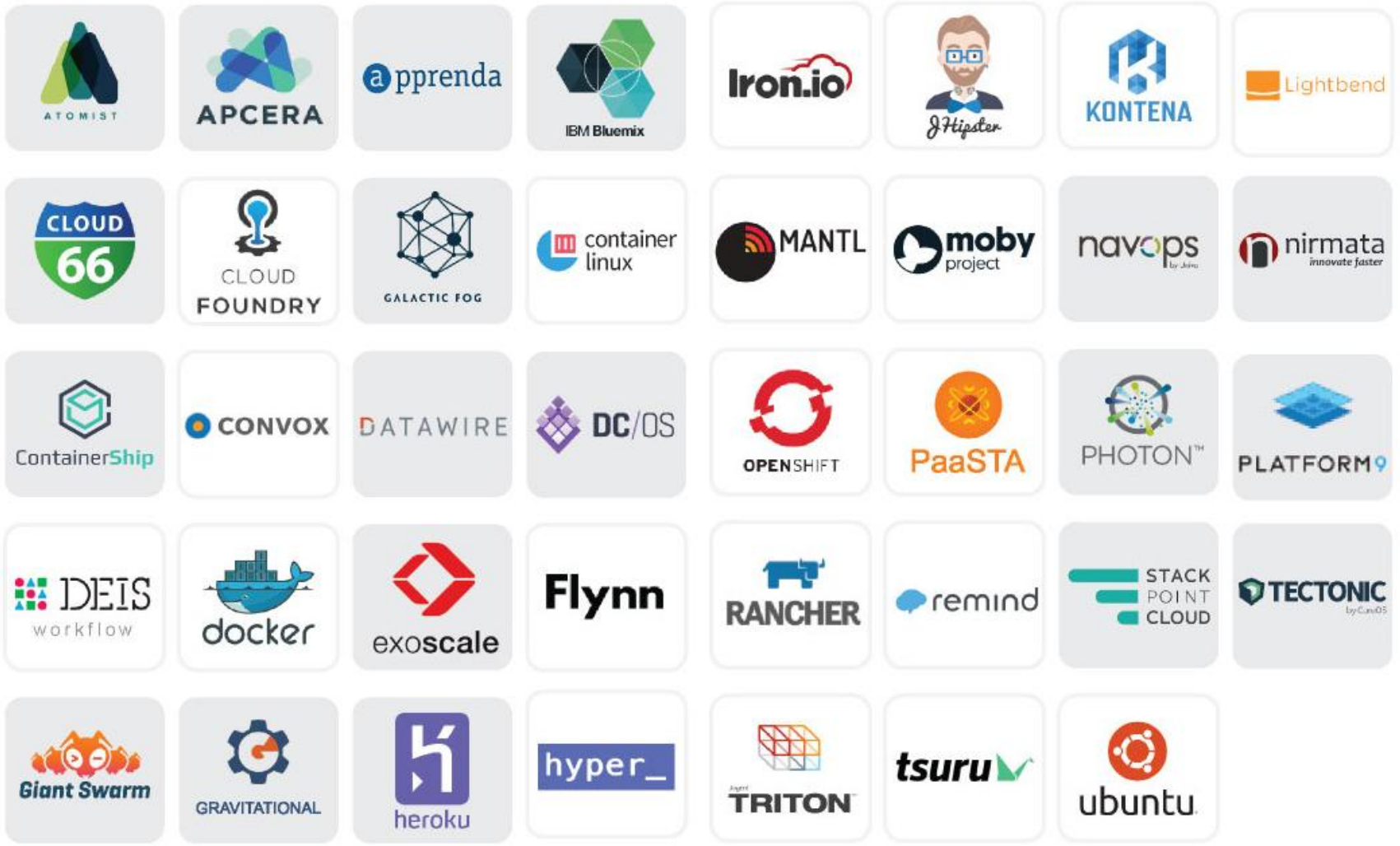
So many…


Database & Data Warehouse
Streaming
SCM
Registry Services
Application Definition
Platforms
Observability & Analysis
App Definition & Development
Scheduling & Orchestration
Coordination & Service Discovery
Service Management
Orchestration & Management
Container Runtime
Cloud-Native Storage
Cloud-Native Network
Runtime
Bare Metal
Infr. Automation
Host Management / Tooling
CI/CD
CI/CD Security
Provisioning
Public Cloud
PaaS / Container Service
Serverless/Event-based
Monitoring
Logging
Tracing
Cloud Native Landscape
v0.9.6
github.com/cncf/landscape
This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.
CLOUD NATIVE COMPUTING FOUNDATION
Redpoint
Amplify

Container runtime

Container services & PaaS

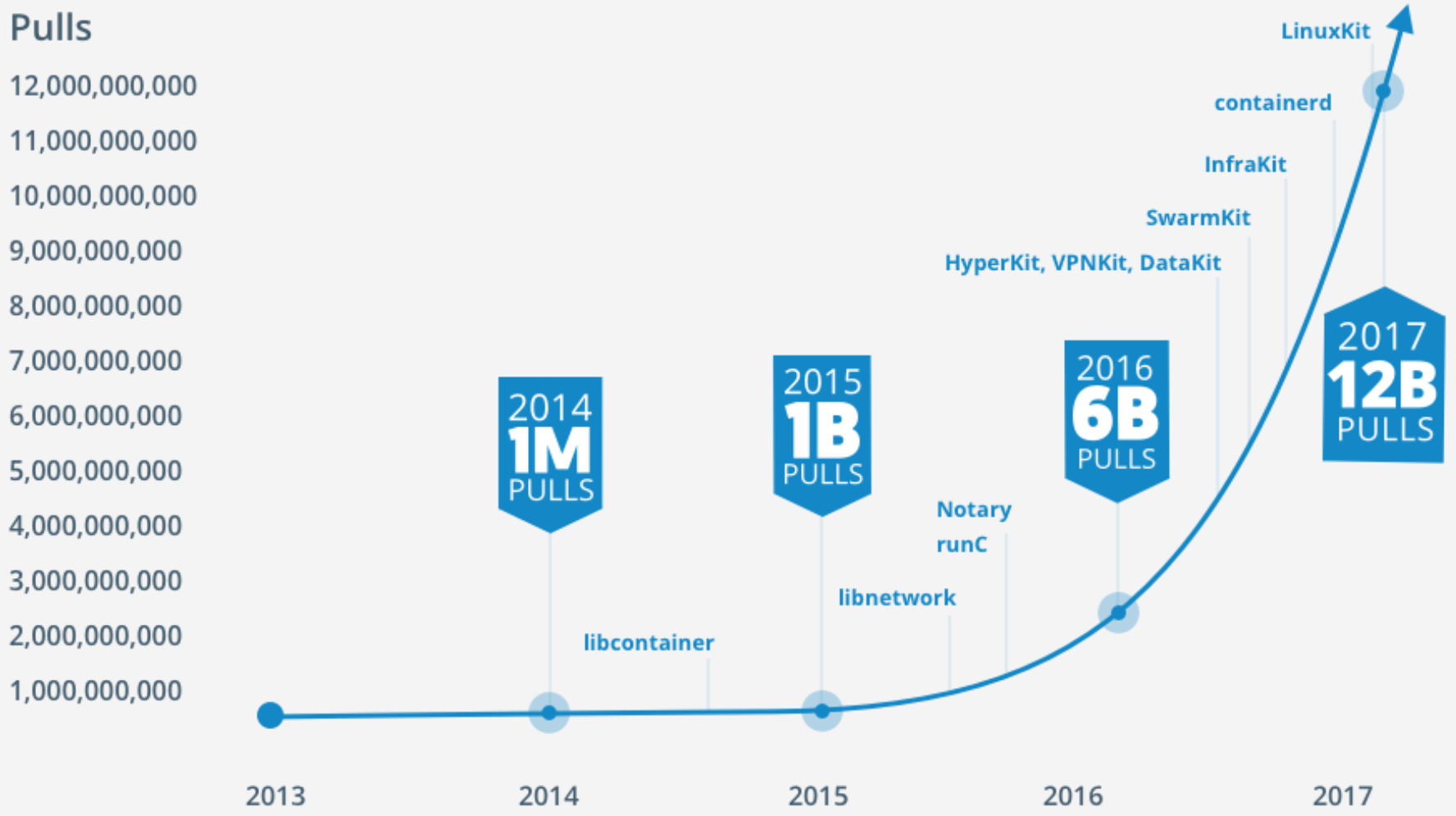
PaaS / Container Service



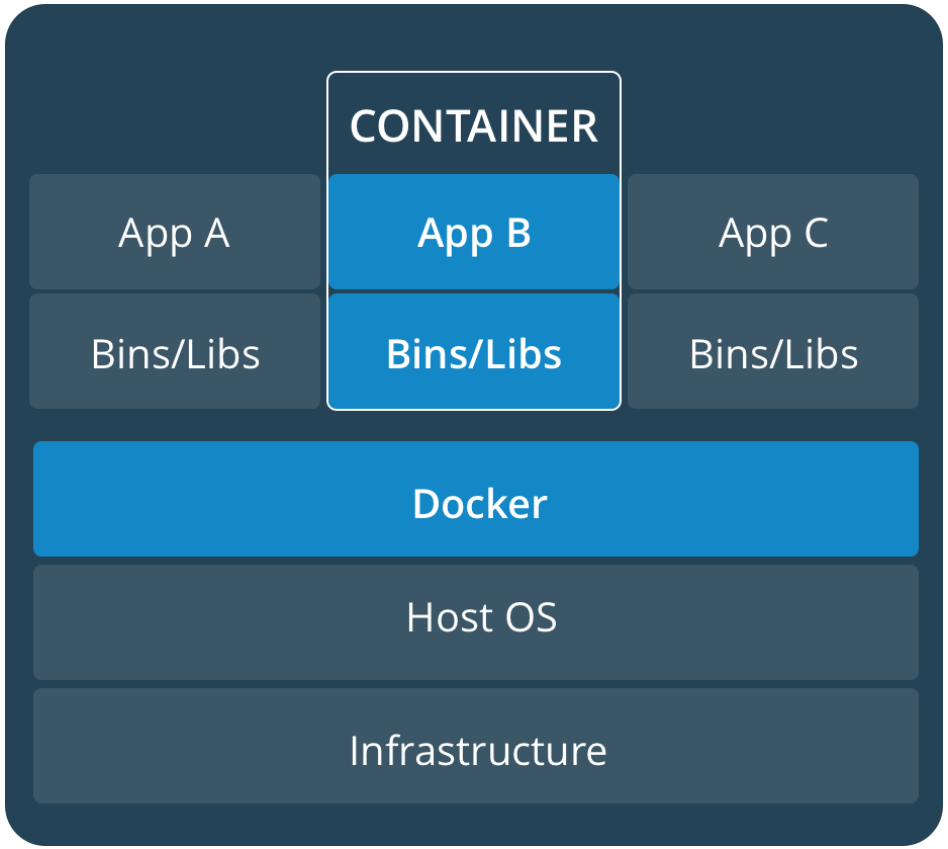
Just container services, which can run local

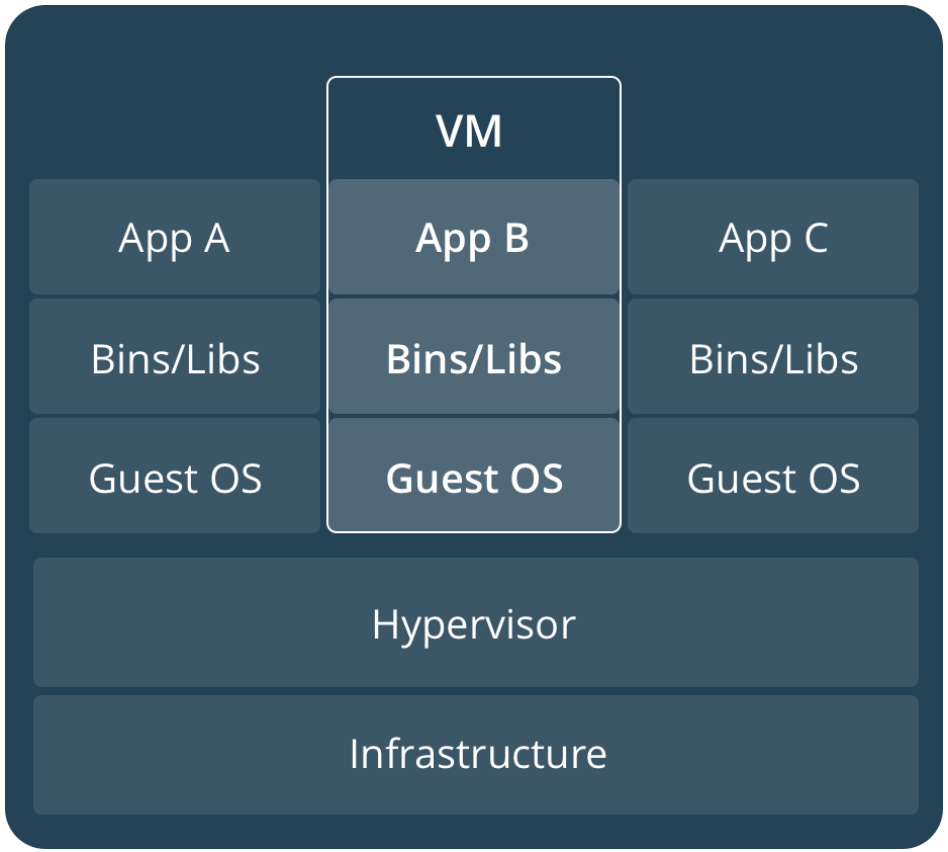
Docker

Docker Era

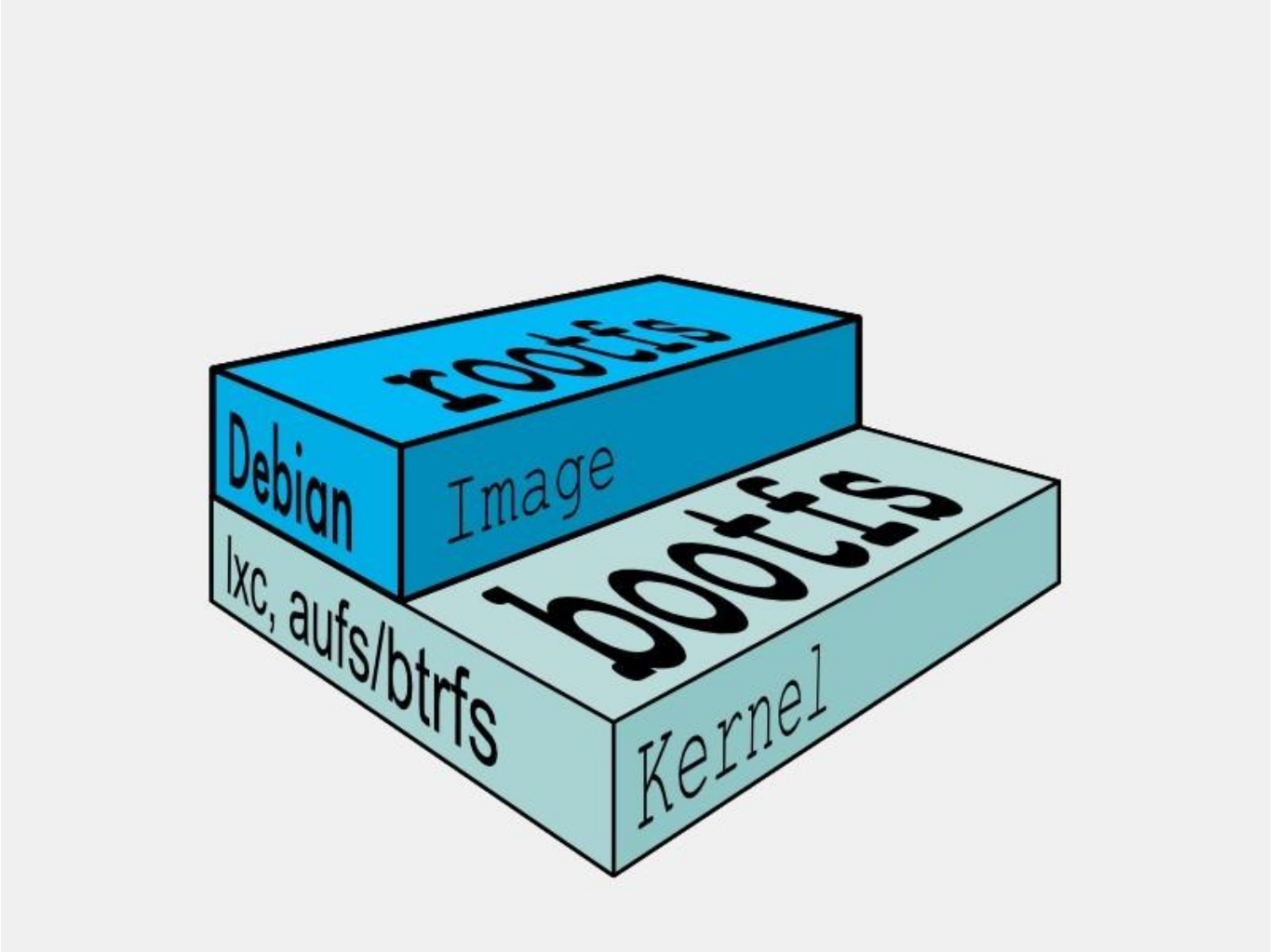

Pulls
12,000,000,000
11,000,000,000
10,000,000,000
9,000,000,000
8,000,000,000
7,000,000,000
6,000,000,000
5,000,000,000
4,000,000,000
3,000,000,000
2,000,000,000
1,000,000,000
2013
2014
2015
2016
2017
2014 1M PULLS
2015 1B PULLS
2016 6B PULLS
2017 12B PULLS
libcontainer
libnetwork
Notary
runC
HyperKit, VPNKit, DataKit
SwarmKit
InfraKit
containerd
LinuxKit

Container vs VM

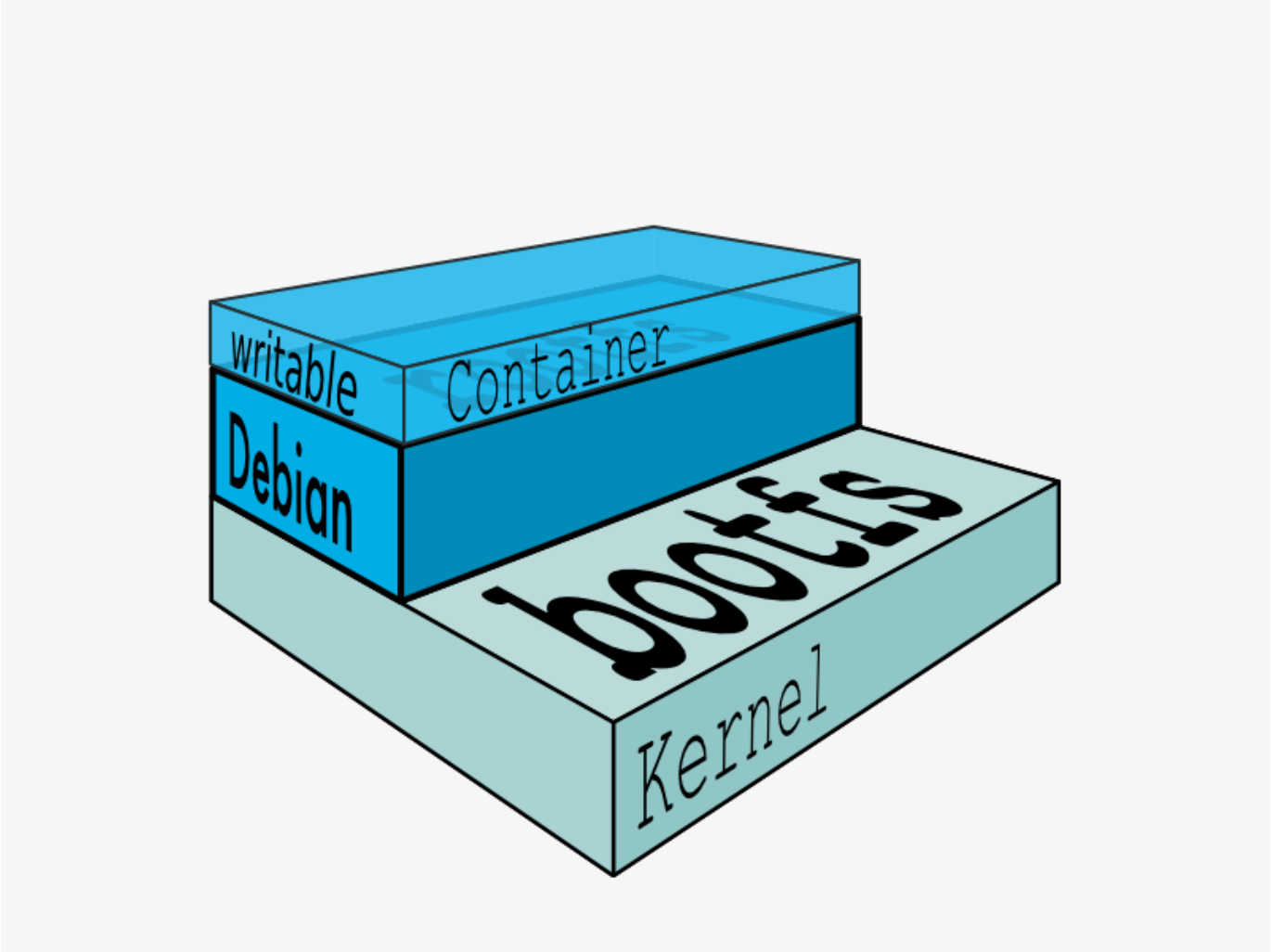




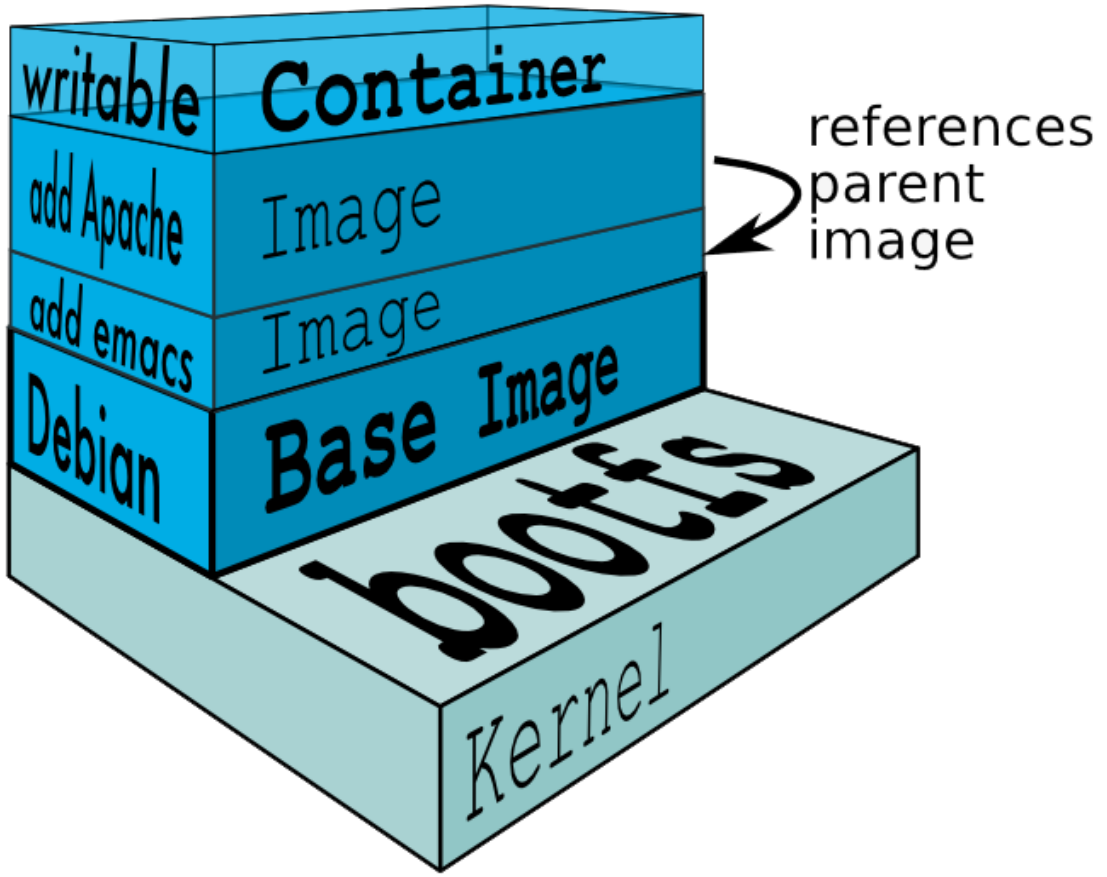
How it works



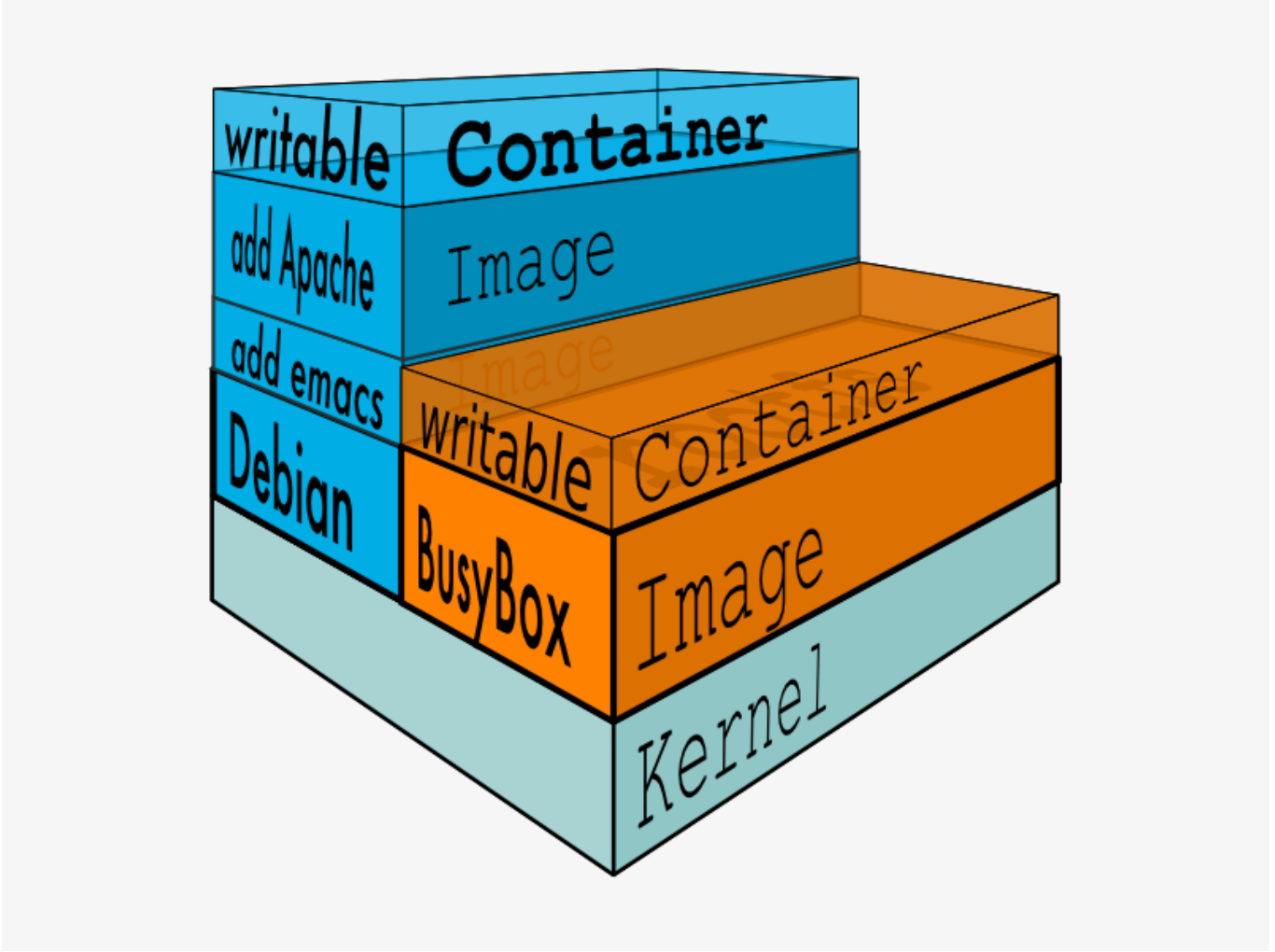
How it works



How it works


writable
Container
add Apache
Image
add emacs
Image
Debian
Base Image
bootfs
Kernel
references
parent
image

How it works


writable
Container
add Apache
Image
add emacs
Debian
writable
Container
BusyBox
Image
Kernel

Dockerfile

Dockerfile

```
1   FROM eboraas/apache:latest
2
3   LABEL maintainer="MaxymVlasov <m@il.com>"
4
5   # Add rewrite module to Apache
6   # Need for port forwarding for NodeJS
7   RUN a2enmod rewrite && service apache2 restart
8
9   # Change default Apache configuration
10  COPY 000-default.conf /etc/apache2/sites-available/000-default.conf
11  COPY ports.conf /etc/apache2/ports.conf
12
13  # Open port
14  EXPOSE 8080
15
16  # Run Apache server in background
17  CMD ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

Dockerfile

```
1   ARG VERSION=latest
2   FROM busybox:${VERSION}
3
4
5   ARG VERSION
6
7   RUN echo ${VERSION} > version
```

Dockerfile

```
1    ARG VERSION=latest
2    FROM busybox:${VERSION}
3
4    ENV VERSION previous
5    ARG VERSION
6
7    RUN echo ${VERSION} > version
```

Dockerfile

```
1   ARG VERSION=latest
2   FROM busybox:${VERSION}
3
4   ARG VERSION
5   ENV VERSION=${VERSION:-0}
6
7   RUN echo ${VERSION} > version
```

Dockerfile

```
1   ARG VERSION=latest
2   FROM busybox:${VERSION}
3
4   ARG VERSION
5   ENV VERSION=${VERSION:-0}
6
7   RUN echo ${VERSION} > version
8
9   ADD http://bit.ly/raw or binary file /app/file
10  COPY . /app
```

Dockerfile

```
1    FROM busybox:latest
2
3    WORKDIR /a
4    WORKDIR b
5    WORKDIR c
6
7    RUN pwd # Output: /a/b/c
```

Dockerfile

```
1   FROM busybox:latest
2
3   WORKDIR /a
4   WORKDIR b
5   WORKDIR c
6
7   RUN pwd # Output: /a/b/c
8
9   ENTRYPOINT ["ping"]
10  CMD ["8.8.8.8"]
```

Dockerfile

```
1    FROM busybox:latest
2
3    LABEL maintainer="Maksym Vlasov <m@il.com>"
```

Dockerfile

```
1    FROM busybox:latest
2
3    LABEL maintainer="Maksym Vlasov <m@il.com>"
4
5    EXPOSE 9000 9001
```

Dockerfile

```
1   FROM busybox:latest
2
3   LABEL maintainer="Maksym Vlasov <m@il.com>"
4
5   EXPOSE 9000 9001
6
7   VOLUME ["/data"]
```

Dockerfile

```
1   FROM busybox:latest
2
3   LABEL maintainer="Maksym Vlasov <m@il.com>"
4
5   EXPOSE 9000 9001
6
7   VOLUME ["/data"]
8
9   HEALTHCHECK --interval=30s \
10              --timeout=30s \
11              --start-period=0s \
12              --retries=3 \
13    CMD wget http://localhost/ || exit 1
```

Dockerfile

```
1    FROM busybox:latest
2
3    SHELL ["/bin/sh", "-c"]
```

Dockerfile

```
1  FROM busybox:latest
2
3  SHELL ["/bin/sh", "-c"]
4
5  RUN addgroup group && \
6      adduser -D -H user -G group
7
8  USER user:group
```

Dockerfile

```
1   FROM busybox:latest
2
3   SHELL ["/bin/sh", "-c"]
4
5   RUN addgroup group && \
6       adduser -D -H user -G group
7
8   USER user:group
9
10  STOPSIGNAL 9 # SIGKILL
```

Dockerfile

```
1   FROM busybox:latest
2
3   SHELL ["/bin/sh", "-c"]
4
5   RUN addgroup group && \
6       adduser -D -H user -G group
7
8   USER user:group
9
10  STOPSIGNAL 9 # SIGKILL
11
12  ONBUILD ADD . /app/src
13  ONBUILD RUN echo "based on busybox"
```

Docker-compose

Docker-compose.yaml

Docker-compose.yaml

```yaml
version: '3'

services:
  apache:
    image: maxymvlasov/apache-pdffiller
    ports:
      - "8080"
    volumes:
      - ./html:/var/www/
    environment:
      - VIRTUAL_HOST=0.0.0.0
      - VIRTUAL_PROTO=http
  nodejs:
    image: maxymvlasov/nodejs-pdffiller
    ports:
      - "8000:8000"
    environment:
      - "constraint:node==node-1"
  nginx:
    image: maxymvlasov/nginx-pdffiller
    ports:
      - "80:80"
    volumes:
      - /var/run/docker.sock:/tmp/docker.sock:ro
    environment:
      - VIRTUAL_PORT=8080
      - "constraint:node==node-1"
```

Docker-compose.yaml

```
1   version: '3'
2
3   services:
4     consul:
5       build: ./.dev_docker_compose/consul
6       network_mode: host
7       restart: always
8       volumes:
9       - consuldb:/consul/data
10
11  volumes:
12    consuldb:
13      external: true
```

Vagga

vagga.yaml

```
1   containers:
```

vagga.yaml

```
1   containers:
2     ubuntu-consul:
```

vagga.yaml

```
containers:
  ubuntu-consul:
    setup:
    - !Ubuntu xenial
    - !Install [unzip, wget, ca-certificates]
```

vagga.yaml

```
containers:
  ubuntu-consul:
    setup:
    - !Ubuntu xenial
    - !Install [unzip, wget, ca-certificates]
    - !Sh |
        cd /tmp
        wget https://releases.hashicorp.com/consul/0.8.3/consul_0.8.3_
        unzip consul_0.8.3_linux_amd64.zip
        cp consul /usr/bin/consul
```

vagga.yaml

```
containers:
  ubuntu-consul:
    setup:
    - !Ubuntu xenial
    - !Install [unzip, wget, ca-certificates]
    - !Sh |
        cd /tmp
        wget https://releases.hashicorp.com/consul/0.8.3/consul_0.8.3_
        unzip consul_0.8.3_linux_amd64.zip
        cp consul /usr/bin/consul
commands:
  consul-server: !Command
```

vagga.yaml

```
containers:
  ubuntu-consul:
    setup:
    - !Ubuntu xenial
    - !Install [unzip, wget, ca-certificates]
    - !Sh |
        cd /tmp
        wget https://releases.hashicorp.com/consul/0.8.3/consul_0.8.3_
        unzip consul_0.8.3_linux_amd64.zip
        cp consul /usr/bin/consul
commands:
  consul-server: !Command
    description: Start consul in server mode
    container: ubuntu-consul
```

vagga.yaml

```
containers:
  ubuntu-consul:
    setup:
    - !Ubuntu xenial
    - !Install [unzip, wget, ca-certificates]
    - !Sh |
        cd /tmp
        wget https://releases.hashicorp.com/consul/0.8.3/consul_0.8.3_
        unzip consul_0.8.3_linux_amd64.zip
        cp consul /usr/bin/consul
commands:
  consul-server: !Command
    description: Start consul in server mode
    container: ubuntu-consul
    run: |
        /usr/bin/consul agent -server -bootstrap-expect=1 \
                              -data-dir=/tmp/consul -log-level=debug \
                              -advertise=127.0.0.1
```

Dev-docker-compose

.generate.yaml

.generate.yaml

```yaml
version: '3'

services:
```

.generate.yaml

```
1   version: '3'
2
3   services:
10  volumes:
11    consuldb:
12      external: true
```

.generate.yaml

```
1   version: '3'
2
3   services:
4     # Key/Value storage with config for our app
5     consul: !include:consul.yaml |
6
7     # Search engine and document-based DB.
8     elasticsearch: !include:elasticsearch.yaml |
9
10  volumes:
11    consuldb:
12      external: true
```

Dev-... vs docker-compose

```yaml
version: '3'

services:
  # Add Key/Value to Consul.
  add_keys_to_consul: !include:add_keys_to_consul.yaml |

  # Key/Value storage with config for our app and other infrastructure.
  consul: !include:consul.yaml |

  # Search engine and document-based DB.
  elasticsearch: !include:elasticsearch.yaml |

  # Add logs from filesystem to Elasticsearch.
  filebeat: !include:filebeat.yaml |

  # Visualization tool for data in Elasticsearch.
  kibana: !include:kibana.yaml |

  # Queue with logs whish should write to DB.
  rabbit: !include:rabbit.yaml |

  # Consul services registrator.
  registrator: !include:registrator.yaml |


volumes:
  rabbitdb:
    external: true
  consuldb:
    external: true

networks:
  default:
    external:
      name: dev-compose-network
```

Dev-... vs docker-compose

```yaml
version: '3'

services:
  # Add Key/Value to Consul.
  add_keys_to_consul: !include:add_keys_to_consul.yaml |

  # Key/Value storage with config for our app and other infrastructure.
  consul: !include:consul.yaml |

  # Search engine and document-based DB.
  elasticsearch: !include:elasticsearch.yaml |

  # Add logs from filesystem to Elasticsearch.
  filebeat: !include:filebeat.yaml |

  # Visualization tool for data in Elasticsearch.
  kibana: !include:kibana.yaml |

  # Queue with logs whish should write to DB.
  rabbit: !include:rabbit.yaml |

  # Consul services registrator.
  registrator: !include:registrator.yaml |


volumes:
  rabbitdb:
    external: true
  consuldb:
    external: true

networks:
  default:
    external:
      name: dev-compose-network
```

```yaml
networks:
  default:
    external:
      name: dev-compose-network
services:
  add_keys_to_consul:
    build: ./.dev_docker_compose/yaml_files/add_keys_to_consul
    depends_on:
    - consul
    environment:
      CONSUL_SERVICE_NAME: dev-app
      REGION: us-east-1
    network_mode: host
    restart: on-failure
    volumes:
    - ./.consul_conf/:/.consul_conf/
  consul:
    build: ./.dev_docker_compose/yaml_files/consul
    network_mode: host
    restart: always
    volumes:
    - consuldb:/consul/data
  elasticsearch:
    deploy:
      resources:
        limits:
          memory: 1g
    environment:
    - cluster.name=ESCluster
    - bootstrap.memory_lock=true
    - ES_JAVA_OPTS=-Xms512m -Xmx512m
    - xpack.security.enabled=false
    - node.name=es_node1
    image: docker.elastic.co/elasticsearch/elasticsearch:5.4.0
    network_mode: host
    ports:
    - 9200:9200
    restart: always
    ulimits:
      memlock:
        hard: -1
        soft: -1
  filebeat:
    build: ./.dev_docker_compose/yaml_files/filebeat
    depends_on:
    - elasticsearch
    environment:
      CONSUL_SERVICE_NAME: dev-app
    network_mode: host
    restart: always
    volumes:
    - ./.elastic_logs/:/.elastic_logs/
  kibana:
    environment:
      ELASTICSEARCH_URL: http://localhost:9200
      SERVER_NAME: http://localhost:5601
      XPACK_MONITORING_UI_CONTAINER_ELASTICSEARCH_ENABLED: 'false'
    image: docker.elastic.co/kibana/kibana:5.4.0
    network_mode: host
    restart: always
  rabbit:
    domainname: rabbit.local
    hostname: rabbitmq
    image: pdffiller/rabbitmq:latest
    network_mode: host
    ports:
    - 4369:4369
    - 5671:5671
    - 5672:5672
    - 15671:15671
    - 15672:15672
    - 25672:25672
    restart: always
    volumes:
    - rabbitdb:/var/lib/rabbitmq
  registrator:
    command: consul://127.0.0.1:8500
    depends_on:
    - consul
    image: gliderlabs/registrator:latest
    network_mode: host
    restart: always
    volumes:
    - /var/run/docker.sock:/tmp/docker.sock
version: '3'
volumes:
  consuldb:
    external: true
  rabbitdb:
    external: true
```

Dev-... vs docker-compose

```yaml
version: '3'

services:
  # Add Key/Value to Consul.
  add_keys_to_consul: !include:add_keys_to_consul.yaml |

  # Key/Value storage with config for our app and other infrastructure.
  consul: !include:consul.yaml |

  # Search engine and document-based DB.
  elasticsearch: !include:elasticsearch.yaml |

  # Add logs from filesystem to Elasticsearch.
  filebeat: !include:filebeat.yaml |

  # Visualization tool for data in Elasticsearch.
  kibana: !include:kibana.yaml |

  # Queue with logs whish should write to DB.
  rabbit: !include:rabbit.yaml |

  # Consul services registrator.
  registrator: !include:registrator.yaml |


volumes:
  rabbitdb:
    external: true
  consuldb:
    external: true

networks:
  default:
    external:
      name: dev-compose-network
```

```yaml
version: '3'

services:
  # Add Key/Value to Consul.
  add_keys_to_consul: !include:add_keys_to_consul.yaml |

  # Key/Value storage with config for our app and other infrastructure.
  consul: !include:consul.yaml |

  # Search engine and document-based DB.
  elasticsearch: !include:elasticsearch.yaml |

  # Add logs from filesystem to Elasticsearch.
  filebeat: !include:filebeat.yaml |

  # Visualization tool for data in Elasticsearch.
  kibana: !include:kibana.yaml |

  # Queue with logs whish should write to DB.
  rabbit: !include:rabbit.yaml |

  # Consul services registrator.
  registrator: !include:registrator.yaml |


volumes:
  rabbitdb:
    external: true
  consuldb:
    external: true

networks:
  default:
    external:
      name: dev-compose-network
```

```yaml
networks:
  default:
    external:
      name: dev-compose-network
services:
  add_keys_to_consul:
    build: ./.dev_docker_compose/yaml_files/add_keys_to_consul
    depends_on:
    - consul
    environment:
      CONSUL_SERVICE_NAME: dev-app
      REGION: us-east-1
    network_mode: host
    restart: on-failure
    volumes:
    - ./.consul_conf/:/.consul_conf/
  consul:
    build: ./.dev_docker_compose/yaml_files/consul
    network_mode: host
    restart: always
    volumes:
    - consuldb:/consul/data
  elasticsearch:
    deploy:
      resources:
        limits:
          memory: 1g
    environment:
    - cluster.name=ESCluster
    - bootstrap.memory_lock=true
    - ES_JAVA_OPTS=-Xms512m -Xmx512m
    - xpack.security.enabled=false
    - node.name=es_node1
    image: docker.elastic.co/elasticsearch/elasticsearch:5.4.0
    network_mode: host
    ports:
    - 9200:9200
    restart: always
    ulimits:
      memlock:
        hard: -1
        soft: -1
  filebeat:
    build: ./.dev_docker_compose/yaml_files/filebeat
    depends_on:
    - elasticsearch
    environment:
      CONSUL_SERVICE_NAME: dev-app
    network_mode: host
    restart: always
    volumes:
    - ./.elastic_logs/:/.elastic_logs/
  kibana:
    environment:
      ELASTICSEARCH_URL: http://localhost:9200
      SERVER_NAME: http://localhost:5601
      XPACK_MONITORING_UI_CONTAINER_ELASTICSEARCH_ENABLED: 'false'
    image: docker.elastic.co/kibana/kibana:5.4.0
    network_mode: host
    restart: always
  rabbit:
    domainname: rabbit.local
    hostname: rabbitmq
    image: pdffiller/rabbitmq:latest
    network_mode: host
    ports:
    - 4369:4369
    - 5671:5671
    - 5672:5672
    - 15671:15671
    - 15672:15672
    - 25672:25672
    restart: always
    volumes:
    - rabbitdb:/var/lib/rabbitmq
  registrator:
    command: consul://127.0.0.1:8500
    depends_on:
    - consul
    image: gliderlabs/registrator:latest
    network_mode: host
    restart: always
    volumes:
    - /var/run/docker.sock:/tmp/docker.sock
version: '3'
volumes:
  consuldb:
    external: true
  rabbitdb:
    external: true
```

Structure

```
1    dev-docker-compose
2    ├── .generate.yaml
```

Structure

```
dev-docker-compose
├── .generate.yaml
├── Makefile
├── Makefile.dev.mk
```

Structure

```
dev-docker-compose
├── .generate.yaml
├── Makefile
├── Makefile.dev.mk
├── .dev_docker_compose
│   ├── Dockerfile
│   ├── loader.py
│   ├── RELEASE_NOTES.md
```

Structure

```
dev-docker-compose
├── .generate.yaml
├── Makefile
├── Makefile.dev.mk
├── .dev_docker_compose
│   ├── Dockerfile
│   ├── loader.py
│   ├── RELEASE_NOTES.md
│   └── yaml_files
│       ├── consul.yaml
│       └── elasticsearch.yaml
│
```

Structure

```
dev-docker-compose
├── .generate.yaml
├── Makefile
├── Makefile.dev.mk
├── .dev_docker_compose
│   ├── Dockerfile
│   ├── loader.py
│   ├── RELEASE_NOTES.md
│   └── yaml_files
│       ├── consul.yaml
│       └── elasticsearch.yaml
├── .consul_conf
│   ├── dev_conf.json
│   └── devops_conf.json
```

Structure

```
dev-docker-compose
├── .generate.yaml
├── Makefile
├── Makefile.dev.mk
├── .dev_docker_compose
│   ├── Dockerfile
│   ├── loader.py
│   ├── RELEASE_NOTES.md
│   └── yaml_files
│       ├── consul.yaml
│       └── elasticsearch.yaml
├── .consul_conf
│   ├── dev_conf.json
│   └── devops_conf.json
├── .elastic_logs
│   └── logs.json
```

Structure

```
dev-docker-compose
├── .generate.yaml
├── Makefile
├── Makefile.dev.mk
├── .dev_docker_compose
│   ├── Dockerfile
│   ├── loader.py
│   ├── RELEASE_NOTES.md
│   └── yaml_files
│       ├── consul.yaml
│       └── elasticsearch.yaml
├── .consul_conf
│   ├── dev_conf.json
│   └── devops_conf.json
├── .elastic_logs
│   └── logs.json
└── README.md
```

Questions?

fb.com/maxymvlasov
@maxymvlasov

bit.ly/msp-docker