

---

# Centos 7 and 8 Install Guide

March 20, 2025



## Contents

Login to server . . . . .	2
Get the Essentials . . . . .	2
Remove non-essentials . . . . .	3
For Centos 8, remove firewalld and install iptables . . . . .	3
Install Docker . . . . .	3
Install Docker Compose . . . . .	4
Allow inter-docker communication . . . . .	4
Pull software . . . . .	4
Configure Kwanza . . . . .	6
Add a certificate . . . . .	7
Test . . . . .	7
<b>Sirenia Analytics</b>	<b>7</b>
Configure Elastic Search . . . . .	11
Configure Nginx Proxy . . . . .	13
Test . . . . .	13
Restart Server . . . . .	14

This document provides a Centos 7 install guide. The guide can be followed for Ubuntu installation or serve as a starting point for installing on other Linux OS.

You should read the Deployment documentation beforehand, in order to understand the components and their roles.

### Login to server

```
1 ssh user@<server>
2 sudo su
3 #password
4 cat /etc/centos-release
5 #CentOS Linux release 7 eller 8
```

### Get the Essentials

```
1 sudo yum -y install epel-release
2 sudo yum install -y htop
3 sudo yum install -y nano
4 sudo yum install -y wget
```

```
5 sudo wget https://github.com/bcicen/ctop/releases/download/v0.7.3/ctop
   -0.7.3-linux-amd64 -O /usr/local/bin/ctop
6 sudo chmod +x /usr/local/bin/ctop
7 sudo yum install -y postgresql
```

### Remove non-essentials

```
1 systemctl stop rpcbind.service
2 systemctl disable rpcbind.service
3 systemctl stop rpcbind.socket
4 systemctl disable rpcbind.socket
```

### For Centos 8, remove firewalld and install iptables

```
1 sudo systemctl stop firewalld
2 sudo systemctl disable firewalld
3 sudo systemctl mask firewalld
4 sudo yum install -y iptables-services
5 sudo systemctl start iptables
6 sudo systemctl start ip6tables
7 sudo systemctl enable iptables
8 sudo systemctl enable ip6tables
```

### Install Docker

On the target machine

```
1 sudo yum install -y yum-utils device-mapper-persistent-data lvm2
2 sudo yum-config-manager --add-repo https://download.docker.com/linux/
   centos/docker-ce.repo
3 wget https://download.docker.com/linux/centos/7/x86_64/edge/Packages/
   containerd.io-1.2.6-3.3.el7.x86_64.rpm
4 yum install -y containerd.io-1.2.6-3.3.el7.x86_64.rpm
5 sudo yum install -y docker-ce docker-ce-cli containerd.io
6 sudo systemctl start docker
7 sudo docker run hello-world
8 sudo systemctl enable docker
9 sudo systemctl status docker
10 ctrl-c to stop
```

If target machine has no internet add http(s) proxy to docker

## Install Docker Compose

On the target machine

```
1 sudo curl -L "https://github.com/docker/compose/releases/download
  /1.27.4/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/
  docker-compose
2 sudo chmod +x /usr/local/bin/docker-compose
3 echo TMPDIR="/opt/compose-tmp" >> /etc/environment
4 mkdir -p /opt/compose-tmp
5 echo "export PATH=/usr/local/bin:$PATH" >> /root/.bashrc
6 source /root/.bashrc
7 docker-compose --version
8 #docker-compose version 1.27.4, build 40524192
```

## Allow inter-docker communication

```
1 sysctl net.bridge.bridge-nf-call-iptables=0
2 sysctl net.bridge.bridge-nf-call-arptables=0
3 sysctl net.bridge.bridge-nf-call-ip6tables=0
4 echo 'net.bridge.bridge-nf-call-iptables=0' >> /etc/sysctl.conf
5 echo 'net.bridge.bridge-nf-call-arptables=0' >> /etc/sysctl.conf
6 echo 'net.bridge.bridge-nf-call-ip6tables=0' >> /etc/sysctl.conf
```

## Pull software

On the target machine pull some Sirenia software

```
1 mkdir /root/deploy
2 cd /root/deploy
```

Create a docker-compose file for your specific setup.

```
1 nano docker-compose.yml
```

You could take a base in this example. You must change at least kwanza version, cuesta version and `#{HOSTNAME}` of your server. You MUST use all small letters in the fqdn. eg. some.sirenia.io

```
1 version: '3'
2
3 networks:
4   default:
5     ipam:
6       driver: default
7       config:
8         - subnet: "172.27.0.0/24"
9
10  services:
11    kwanza:
12      image: registry.sirenia.io/kwanza:v2.16.2
13      restart: unless-stopped
14      environment:
15        KWANZA_DATABASE: pg://postgres:postgres@postgres/kwanza
16        KWANZA_MINTLSVERSION: 1.2
17        KWANZA_CIPHERSUITES: "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
18          TLS_RSA_WITH_AES_128_GCM_SHA256
19          TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
20          TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256"
21        KWANZA_PREFERSERVERCIPHERSUITES: "True"
22        KWANZA_STRICTTRANSPORTSECURITY: "True"
23        KWANZA_CERT_SUBJECTS: "${HOSTNAME}"
24        KWANZA_CERT_DURATION: 87600h
25        KWANZA_CERT: "/cert/cert.pem"
26        KWANZA_KEY: "/cert/key.pem"
27        KWANZA_SALT: kwanzified
28        KWANZA_AUTH: jwt
29        KWANZA_MAXSTREAMSPERSUBSCRIBER: 102400
30        KWANZA_MAXAUTHTHROTTLEDKEYS: -1
31        KWANZA_MAXTHROTTLEDKEYS: -1
32      ports:
33        - "8000:8000"      # HTTP(S)
34        - "8001:8001"      # TCP (gRPC)
35        - "127.0.0.1:6060:6060"  # Profiling to host-only
36        - "127.0.0.1:8080:8080"  # Expvar to host-only
37      volumes:
38        - "/usr/local/etc/sirenia/cert:/cert"
39        - "/usr/local/etc/sirenia/kwanza/conf:/etc/sirenia/kwanza"
40      depends_on:
41        - postgres
```

```
40  cuesta:
41     image: registry.sirenia.io/cuesta:v1.14.17
42     restart: unless-stopped
43     environment:
44         CUESTA_CERT: "/cert/cert.pem"
45         CUESTA_KEY: "/cert/key.pem"
46         KWANZA_URL: "https://${HOSTNAME}:8000/v1"
47         KWANZA_STREAMURL: "wss://${HOSTNAME}:8000/v1/stream"
48     ports:
49         - "80:80"
50         - "443:443"
51     volumes:
52         - "/usr/local/etc/sirenia/cert:/cert"
53     depends_on:
54         - kwanza
55
56     postgres:
57         image: postgres:10
58         restart: always
59         ports:
60             - "127.0.0.1:5432:5432"
61         environment:
62             PGDATA: "/data"
63             POSTGRES_PASSWORD: "postgres"
64         volumes:
65             - "/root/postgresdata:/data"
```

## Configure Kwanza

```
1  mkdir -p /usr/local/etc/sirenia/kwanza/conf
2  cd /usr/local/etc/sirenia/kwanza/conf
3  nano .kwanza.yml
```

paste this

```
1  users:
2     john:
3         d224cfd091471383708424f3e494f8029b456b0e559fe82ee9adb5b66a7f1e55
4     martin:
5         d224cfd091471383708424f3e494f8029b456b0e559fe82ee9adb5b66a7f1e55
6     jonathan:
7         d224cfd091471383708424f3e494f8029b456b0e559fe82ee9adb5b66a7f1e55
```

Now pull some software from the repository and try to start the combined setup.

```
1 cd /root/deploy
2 docker login registry.sirenia.io
3 #dist-<username> / <password>
4 # ... Login Succeeded
5 docker-compose up
6 <ctrl-c> (stop again)
```

### Add a certificate

Kwanza will generate self-signed cert at startup. Alternatively copy valid cert for prod here `/usr/local/etc/sirenia/cert` It must be a valid x.509 certificate with a full trust chain to a CA in PEM format.

### Test

Ok, we are ready to test the complete setup

```
1 cd /root/deploy/
2 docker-compose stop
3 docker-compose up
```

Look for errors etc in the logs. Login to Cuesta

- `https://<FQDN>/`
- `user:john pass:1234`

If no errors show up, we are ready to go. Start the setup as background processes.

```
1 docker-compose stop
2 docker-compose up -d
```

## Sirenia Analytics

If you have acquired a license to the Data Driven Operational Intelligence solution Sirenia Analytics, follow the installation guide here. You can deploy this on the same server as Cuesta and Kwanza (assuming it is sized correctly), or on its own. If you install on a new server, you must first install docker and docker-compose as explained above.

Create a docker-compose file for your specific setup (or add to existing).

```
1 mkdir /root/deploy-elk
2 cd /root/deploy-elk
3 nano docker-compose.yml
```

You could take a base in this example. You must change at least versions and <FQDN> of your server.

```
1 version: '2'
2
3 networks:
4   default:
5     ipam:
6       driver: default
7       config:
8         - subnet: "172.28.0.0/24"
9
10  services:
11
12    nginx-proxy:
13      container_name: nginx-proxy
14      image: jwilder/nginx-proxy
15      ports:
16        - "81:443"
17      restart: always
18      #environment:
19      volumes:
20        - "/var/run/docker.sock:/tmp/docker.sock:ro"
21        - "./nginx-proxy/htpasswd:/etc/nginx/htpasswd"
22        - "/usr/local/etc/sirenia/cert:/etc/nginx/certs"
23
24    aripuana-stats:
25      image: registry.sirenia.io/aripuana:v1.5.1
26      restart: unless-stopped
27      environment:
28        ARIPUANA_MINTLSVERSION: 1.2
29        ARIPUANA_CIPHERSUITES: "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
30                                TLS_RSA_WITH_AES_128_GCM_SHA256
31                                TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
32                                TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256"
33        ARIPUANA_PREFERSERVERCIPHERSUITES: "True"
34        ARIPUANA_STRICTTRANSPORTSECURITY: "True"
35        ARIPUANA_CERT_SUBJECTS: "${HOSTNAME}"
```



```
33     ARIPUANA_CERT_DURATION: 87600h
34     ARIPUANA_CERT: "/cert/cert.pem"
35     ARIPUANA_KEY: "/cert/key.pem"
36     ARIPUANA_SALT: "fishy"
37     ARIPUANA_WRITERS: 1
38     ARIPUANA_PORT: 8083
39     ARIPUANA_LOGNAME: "stats.manatee"
40     ARIPUANA_OUTPUTDIR: "/data"
41     ports:
42     - "8082:8082"
43     - "8083:8083"
44     volumes:
45     - "/usr/local/etc/sirenia/cert:/cert"
46     - "./aripuana/data:/data"
47
48     aripuana-logs:
49     image: registry.sirenia.io/aripuana:v1.5.1
50     restart: unless-stopped
51     environment:
52     ARIPUANA_MINTLSVERSION: 1.2
53     ARIPUANA_CIPHERSUITES: "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
54     TLS_RSA_WITH_AES_128_GCM_SHA256
55     TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
56     TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256"
57     ARIPUANA_PREFERSERVERCIPHERSUITES: "True"
58     ARIPUANA_STRICTTRANSPORTSECURITY: "True"
59     ARIPUANA_CERT_SUBJECTS: "${HOSTNAME}"
60     ARIPUANA_CERT_DURATION: 87600h
61     ARIPUANA_CERT: "/cert/cert.pem"
62     ARIPUANA_KEY: "/cert/key.pem"
63     ARIPUANA_SALT: "fishy"
64     ARIPUANA_WRITERS: 1
65     ARIPUANA_PORT: 8085
66     ARIPUANA_LOGNAME: "all.manatee"
67     ARIPUANA_OUTPUTDIR: "/data"
68     ports:
69     - "8084:8084"
70     - "8085:8085"
71     volumes:
72     - "/usr/local/etc/sirenia/cert:/cert"
73     - "./aripuana/data:/data"
74
75     aripuana-perf:
```

```
73     image: registry.sirenia.io/aripuana:v1.5.1
74     restart: unless-stopped
75     environment:
76         ARIPUANA_MINTLSVERSION: 1.2
77         ARIPUANA_CIPHERSUITES: "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
78             TLS_RSA_WITH_AES_128_GCM_SHA256
79             TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
80             TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256"
81         ARIPUANA_PREFERSERVERCIPHERSUITES: "True"
82         ARIPUANA_STRICTTRANSPORTSECURITY: "True"
83         ARIPUANA_CERT_SUBJECTS: "${HOSTNAME}"
84         ARIPUANA_CERT_DURATION: 87600h
85         ARIPUANA_CERT: "/cert/cert.pem"
86         ARIPUANA_KEY: "/cert/key.pem"
87         ARIPUANA_SALT: "fishy"
88         ARIPUANA_WRITERS: 1
89         ARIPUANA_PORT: 8087
90         ARIPUANA_LOGNAME: "perf.manatee"
91         ARIPUANA_OUTPUTDIR: "/data"
92     ports:
93         - "8086:8086"
94         - "8087:8087"
95     volumes:
96         - "/usr/local/etc/sirenia/cert:/cert"
97         - "./aripuana/data:/data"
98
99     elk6:
100     container_name: elk6
101     environment:
102         ES_JAVA_OPTS: "-Xmx1500m -Xms1500m"
103         EL_JAVA_OPTS: "-Xmx256m -Xms256m"
104         VENDOR: Sirenia
105         ELASTICSEARCH_START: 1
106         LOGSTASH_START: 1
107         KIBANA_START: 1
108         VIRTUAL_HOST: "${HOSTNAME}" # will be fwd by nginx proxy
109         VIRTUAL_PORT: 5601 # will be fwd by nginx proxy
110         CERT_NAME: linked_for_nginx
111     image: registry.sirenia.io/sirenia-elk-7:7.2.0.1
112     restart: always
113     volumes:
114         - "./elk6/conf.d:/etc/logstash/conf.d/"
115         - "./aripuana/data:/etc/logstash/indata/"
```

```
113     - "./elk6/elk-data:/var/lib/elasticsearch/" #OBS: Required
        chown 991:991 elk6/elk-data/
114     expose:
115     - "5601"
116
117     #elk6-readonly:
118     #   container_name: elk6-readonly
119     #   environment:
120     #     VENDOR: Sirenia
121     #     KIBANA_START: 1
122     #     VIRTUAL_HOST: "ro-${HOSTNAME}" # will be fwd by nginx proxy
123     #     VIRTUAL_PORT: 5601 # will be fwd by nginx proxy
124     #     CERT_NAME: linked_for_nginx
125     #   image: registry.gitlab.com/sirenia/dist/analytics/sirenia-elk-7-
        readonly:7.2.0.6
126     #   restart: always
```

Make sym-links for cert for proxy use

```
1 cd /usr/local/etc/sirenia/cert
2 ln -s key.pem linked_for_nginx.key
3 ln -s cert.pem linked_for_nginx.crt
```

Pull the software and initialize folder structure.

```
1 cd /root/deploy-elk
2 docker-compose up
```

Wait for download of software and start-up of all dockers. Is expected til give errors, as the setup have not been configured yet.

```
1 ctrl-c to stop
```

## Configure Elastic Search

To configure Elastic do the following

```
1 chown 991:991 elk6/elk-data/
2 echo "vm.max_map_count=262144" >> /etc/sysctl.conf
3 sysctl -w vm.max_map_count=262144
4 cd elk6/conf.d
5 nano logstash-in-out.conf
```

Add this to the file

```
1 input {
2   file {
3     #All for debug
4     type => "all-manatee"
5     path => "/etc/logstash/indata/all.manatee*.log"
6     #start_position => "beginning"
7     start_position => "end"
8     codec => json
9   }
10  file {
11    #Stats for BI only
12    type => "bi-manatee"
13    path => "/etc/logstash/indata/stats.manatee*.log"
14    #start_position => "beginning"
15    start_position => "end"
16    codec => json
17  }
18  file {
19    #perf for perf only
20    type => "perf-manatee"
21    path => "/etc/logstash/indata/perf.manatee*.log"
22    #start_position => "beginning"
23    start_position => "end"
24    codec => json
25  }
26 }
27 filter {
28   #NOOP
29 }
30 output {
31   if [type] == "all-manatee" {
32     elasticsearch {
33       hosts => ["localhost"]
34       manage_template => false
35       index => "all-manatee-1"
36     }
37   }
38   if [type] == "bi-manatee" {
39     elasticsearch {
40       hosts => ["localhost"]
41       manage_template => false
```

```
42     index => "all-manatee-1"
43   }
44 }
45 if [type] == "perf-manatee" {
46   elasticsearch {
47     hosts => ["localhost"]
48     manage_template => false
49     index => "all-manatee-perf-1"
50   }
51 }
52 }
```

## Configure Nginx Proxy

To configure the Nginx Proxy do the following. Change user and password according to your desired setup

```
1 cd ../../nginx-proxy/htpasswd/
2 yum install -y httpd-tools
3 htpasswd -nb user password >> <FQDN>
```

## Test

Ok, we are ready to test the complete DDOI setup. Start all dockers

```
1 cd ../../
2 docker-compose up
```

Look for errors etc in the logs. Login to Sirenia Analytics

- `http://<FQDN>:81/`
- `user:user pass:password`

If no errors show up, we are ready to go. Start the setup as background processes. `ctrl-c` to stop

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

Ensure that the containers are running as expected

```
1 docker-compose ps
```

Should produce output showing five containers running un Up state.

1	Name	Command Ports	State
2	-----		
3	aripuana-logs	aripuana run 0.0.0.0:8084->8084/tcp, 0.0.0.0:8085->8085/tcp	Up
4	aripuana-perf	aripuana run 0.0.0.0:8086->8086/tcp, 0.0.0.0:8087->8087/tcp	Up
5	aripuana-stats	aripuana run 0.0.0.0:8082->8082/tcp, 0.0.0.0:8083->8083/tcp	Up
6	elk6	/usr/local/bin/start.sh 5601/tcp, 9200/tcp, 9300/tcp	Up 5044/tcp,
7	nginx-proxy	/app/docker-entrypoint.sh ... 0.0.0.0:81->443/tcp, 80/tcp	Up

## Restart Server

You should always finish an install procedure with a complete server restart, to test that all services starts after a complete host restart

```
1 reboot -n
```