Micro Focus
Fortify Application Defender

Software Version: 18.20

On-Premises Installation Guide

Document Release Date: December 2018
Software Release Date: December 2018

Legal Notices

Micro Focus
The Lawn
22-30 Old Bath Road
Newbury, Berkshire RG14 1QN
UK

https://www.microfocus.com

Warranty

The only warranties for products and services of Micro Focus and its affiliates and licensors (“Micro Focus”) are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Except as specifically indicated otherwise, a valid license from Micro Focus is required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number
- Document Release Date, which changes each time the document is updated
- Software Release Date, which indicates the release date of this version of the software

To check for recent updates or to verify that you are using the most recent edition of a document, go to:

https://www.microfocus.com/support-and-services/documentation

Contents

Preface

Contacting Micro Focus Fortify Customer Support

If you have questions or comments about using this product, contact Micro Focus Fortify Customer Support using one of the following options.

To Manage Your Support Cases, Acquire Licenses, and Manage Your Account

https://softwaresupport.softwaregrp.com

To Call Support

1.844.260.7219

For More Information

For more information about Fortify software products:
https://software.microfocus.com/solutions/application-security

About the Documentation Set

The Fortify Software documentation set contains installation, user, and deployment guides for all Fortify Software products and components. In addition, you will find technical notes and release notes that describe new features, known issues, and last-minute updates. You can access the latest versions of these documents from the following Micro Focus Product Documentation website:

https://www.microfocus.com/support-and-services/documentation

Change Log

The following table lists changes made to this guide.

Software Release-Version	Change
18.20	Added: • Support for encrypting sensitive values in the appdefender.properties file • Agent Support for Tomcat 9 and IIS 10 Updated: • Docker engine and kernel support for 18.09 and later
17.1	Added: • Support for Docker version 17.05.0 • SMTP email server authentication
16.3	Added: • rsyslog_defender service to consume logs from all application services • A new script to print the status of all Application Defender containers • Support for Docker 1.12

Getting Started

This document provides administrators with instructions on installing and running Application Defender (App Defender) in either a single instance or clustered installation.

This section contains the following topics:

Intended Audience

The intended audience is anyone who wants to deploy an on-premises installation of Application Defender. A basic understanding of hardware and server management is assumed.

Overview of the On-premises Installation of Application Defender

There are two Fortify Fortify Application Defender installation options; the single instance is the minimum recommended deployment, and a clustered installation which provides higher availability, reliability, and scaling than a single instance installation.

This section contains the following topics:

Upgrade

If you're upgrading from a previous version of Application Defender On-Premises, the following instruction will guide you through the process:

Regenerate Docker-compose Files

1. Get the latest version of the Application Defender installation package from the Application Defender portal.
2. Update the `appdefender.properties` file to be used for updated or new properties. Please refer to "Clustered Installation " on page 22for additional details. Use latest version number in the

version field (for example, use version 18.20 if you're upgrading to version 18.20).

3. Execute generation script with the updated properties file. This will generate compose files to start Application Defender instance.

Upgrade Docker Images

There are two categories of Application Defender images:

- Infrastructures: All Infrastructure Docker images are updated as needed.
- Applications: All Application images are updated monthly.

1. Stop all Application Defender containers using the following command:

```
# docker stop $(docker ps -a -q)
```

2. Remove all Application Defender containers using the following command:

```
# docker rm -vf $(docker ps -a -q)
```

3. Start Application Defender instance using the installation steps:

   a) If rsyslog_defender is enabled for centralized logging, you will need to start rsyslog_defender container:

```
#docker-compose -f infrastructures.yml up -d db_migrations
#docker-compose -f applications.yml up -d rsyslog_defender
#docker-compose -f infrastructures.yml -f applications.yml up -d ui_
customer
#docker-compose -f infrastructures.yml -f applications.yml up -d
```

   b) If rsyslog_defender is not enabled for centralized logging:

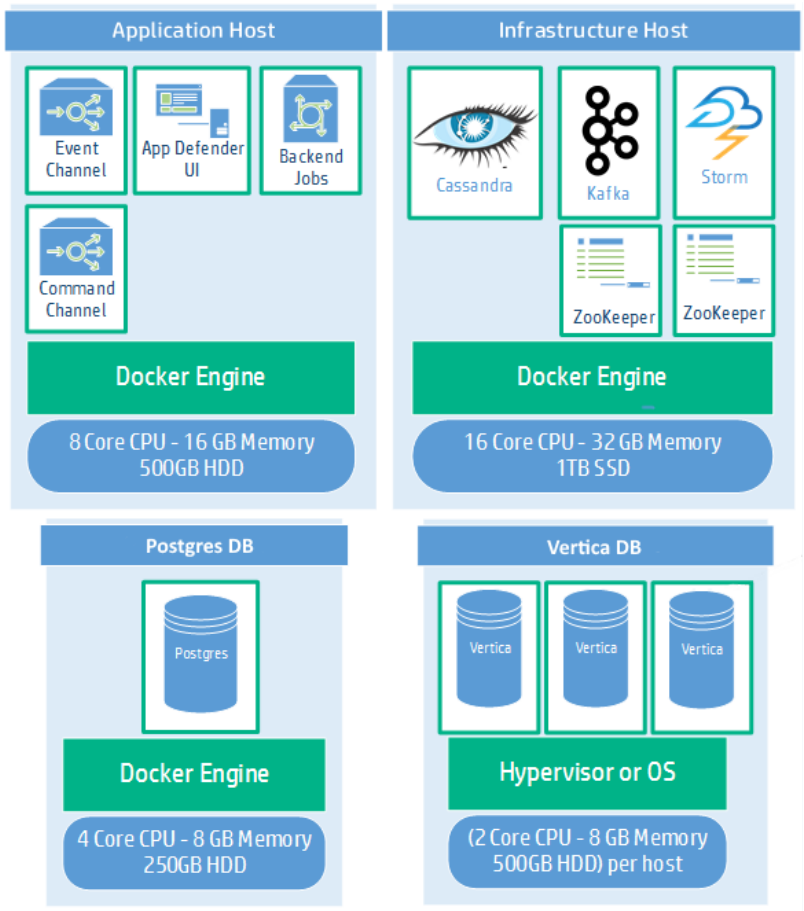
```
#docker-compose -f infrastructures.yml up -d db_migrations
#docker-compose -f infrastructures.yml -f applications.yml up -d ui_
customer
#docker-compose -f infrastructures.yml -f applications.yml up -d
```

4. Start the Storm UI to troubleshoot Storm topologies that have been submitted:

```
#docker-compose -f infrastructures.yml -f applications.yml -f
optional.yml up -d storm_ui
```

Overview: Single Instance Installation

The following diagram illustrates an Application Defender (App Defender) on-premises environment. The minimum recommended deployment consists of an application host, an infrastructure host, a postgres host, and three Vertica hosts.



Application Host Components

Component	Description
Event Channel	Secure communication channel between App Defender Agent and Service used by Agents to send events to the Service.
Application Defender UI	The website used by all App Defender users to access Protect, Manage, Messaging and Alerting functionality.
Backend Jobs	Component used to manage and schedule internal back-end jobs, such as reports.
Command Channel	Secure communication channel between the App Defender Agents and Service used for exchange of commands.

Component	Description
rsyslog_ defender	syslog container to consume logs from Application Defender app servives. This includes logs for the following services: • ui_customer • command_channel • backend_jobs • Edge

Infrastructure Host Components

Component	Description
Apache Cassandra	An open source distributed database used by App Defender to store intermediate data for alerting.
Apache Kafka	A Stateless Distributed Queue used for reports, events, and activity stream processing.
Apache Storm	A distributed real-time stream computation system. Application Defender currently uses Storm topologies for notifications, reporting, alerting, reconciliation, and writing events to Vertica.
Apache Zookeeper	A service for maintaining configuration information, naming, distributed synchronization, and group services used by Kafka and Storm.

Postgres Database Component

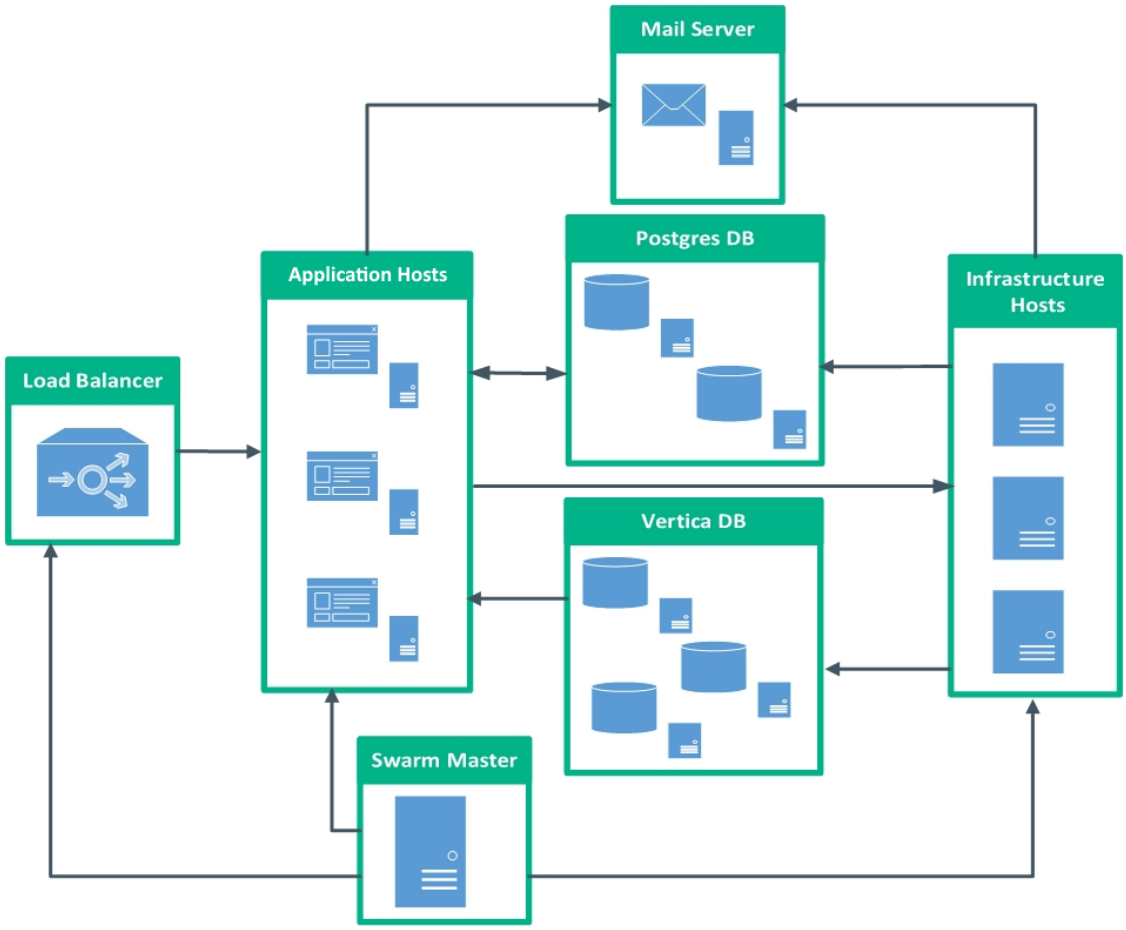
Component	Description
Postgres	An object-relational database that stores user data for Application Defender.

Vertica Database Component

Component	Description
Vertica	A columnar database that stores event data for Application Defender. Clustering the Vertica hosts makes it easier to scale up the entire system at a later date. Fortify recommends a minimum of three clustered Vertica hosts. A single instance is supported, but not recommended.

Overview: Clustered Installation

The following diagram illustrates the Clustered App Defender on-premises installation. Clustered installations require experience with clustering networks.



Clustered Install Components

Component	Description
Application Hosts	List of the nodes designated for application components. Node information includes the following properties: • Number assigned to the host (numeric range: **1-255**). • IP Address • Hostname
Infrastructure Hosts	List of the nodes designated for infrastructure components. Node information includes following properties: • Number assigned to the host (numeric range: **1-255**).

Clustered Install Components, continued

Component	Description
	• IP Address • Hostname
Load Balancer	Load Balancer host. The front end node which provides the interface for users and agents to interact with.
Mail Server	Application that sends and receives email from local users (people within the same domain) and remote senders.
Postgres Database	Database server, its primary function is to store data securely and allow for retrieval at the request of other software applications.
Swarm Master	Manages the resources for the entire cluster.

Hardware Requirements

An on-premises Installation of Application Defender requires the following hardware and a cluster of three or more Vertica instances :

While you can create an installation with a single Vertica instance, Fortify strongly recommends deploying a Vertica cluster of three or more instances. Because your data is not replicated when installing a single Vertica instance there is a risk that you could lose security event data. Moving from a single node deployment to a clustered Vertica deployment in the future will require manual data migration.

Component	CPU	Memory	Hard Drive
Application	8 cores	16 GB	500 GB HDD
Infrastructure	16 cores	32 GB	1 TB SSD
Postgres Database	4 cores	8 GB	Single Instance: 250 GB HDD Clustered: 500 GB HDD
Vertica (x3)	2 cores	8 GB	500 GB HDD per host

For additional Vertica requirements, see "Additional References" on page 40.

To begin the installation process, see "Single Instance Installation " on page 15.

To begin the installation process, see "Clustered Installation " on page 22.

Software Requirements

The following software requirements apply to both single and clustered installations, except where noted.

Network Connection

All of the Application Defender hosts (application, infrastructure, Postgres, and Vertica) need to communicate with each other. Communication ports on the Application Defender apps server should be open to allow all application servers access to the Application Defender service. Refer to the Overview of the On-premises Installation of Application Defender diagram for additional networking and service details.

Docker Hub

A Docker Hub account is needed to access Application Defender docker images. To gain access to the required Docker repositories, provide your Docker Hub account username to your Application Defender account team or Fortify technical support representative.

Firewall Rules

Firewalls on all machines should be configured to allow communication across hosts.

For additional port information, see Troubleshooting your Application Defender Installation.

SMTP Server (mail)

Application Defender sends an email notifications to each user in the system. Provide a proper reference to the SMTP server to be used by Application Defender. For more information, see "Advanced Installation Notes" on page 30.

Vertica Database Cluster

- Use Vertica documentation to install a Vertica cluster. See Additional References for links to the Vertica site.
- Firewall rules must allow application and infrastructure host access.

Note: Single node installations are not as reliable and require data migration to grow your App Defender installation into a cluster at a later date. This and other limitations of single node installations make them less suitable for use in production environments.

Postgres Database

- Fortify recommends that you use a Postgres container. To use a Postgres container to start Application Defender, see the Postgres Container section for alternate installation information.
- If you have an existing Postgres database, use the `appdefender.sql` file generated when using the generation script to create the "appdefender" database.

Linux Machines

Install the following software on your Linux machines:

- **RHEL 7 or CentOS 7:** Kernel version 3.10 or later
- **Docker-engine:**version 18.09 or later
- **Docker-compose:** version 1.7.0
- **Python:** version 2.7.11
- **Java:** Openjdk version 7 or 8

For additional information on Docker, Postgres or Vertica see Additional References.

Application Defender License

You should have received an email containing your license key and instructions on how to redeem the keys. If you have not received the email, contact Micro Focus Fortify support.

Additonal Settings for Clustered Installations

- Network time protocol (ntp) should be running on all nodes.
- Update `/etc/hosts` file with the required IP - host mapping on all nodes.

Application Defender Installation Package

The Application Defender installation package (`AppDefender_xx.xx.zip`) contains the following files:

File Name	Purpose
`CertGeneration.tar.gz`	Files needed to auto-generate java keystore files.
`generate-compose-yaml.py`	Use an existing database or use this script to create docker-compose files, haproxy template, and the database creation script.
`appdefender.properties` (sample)	Used as an argument with the `generate-compose-yaml.py` script to create different App Defender services.
`2018SecurityContent.zip`	Package used to populate the App Defender service with the latest security content.
Fortify Application Defender On-Premises Installation Guide	This document.
Vertica OEM license	An open license for Vertica that includes technical support.
ArcSight Enterprise Security Manager (ESM) content	ArcSight Enterprise Security Manager enables Application Logging, Application Protection-specific dashboards, and ESM use cases.

Installing Application Defender

Fortify offers two on-premises installation options. The Single Instance Installation allows you to start an instance of Micro FocusFortify Application Defender. It supports a specific number of users and agents determined by your hardware configuration. A Clustered Installation supports high availability (HA), continuous operations or zero downtime deployments, and replication. It scales to support any number of Application Defender agents.

Begin by selecting the installation that best matches your needs:

Single Instance Installation

The single instance on-premises installation is intended for users with a small number of agents or a proof-of-value installation.

Perform the following steps in order:

1. Copy the Installation Package to a Linux Machine

Copy the entire installation package to a folder in your `opt` directory (for example: `/opt/appdefender`).

Note: You must have read/write/execute privileges to install Application Defender.

2. Generate the Java Keystore

1. Run the `build-stores.sh` script.
2. Enter a server certificate option at the prompt.
   - Enter **1** for Self-signed Server Certificate Generation.
     Self-signed certificate scripts are used with trial or pilot installations.

     ```
     #export PATH=$PATH:$JAVA_HOME/bin
     #cd CertGeneration
     #chmod 755 build-stores.sh
     #chmod 755 server-root-self-signed.sh
     #sh build-stores.sh
     #<Press Enter>
     ```

   - Enter **2** if you have a server certificate signed by valid certificate authority (CA).
     Copy the signed server certificate (`server.crt`), server private key (`server.key`), CA intermediate Root cert (`server.int.crt`) , and CA Root cert (`server.root.crt`) into the third party folder.

     Note: Use the filenames provided in parentheses. Rename your files if necessary.

     ```
     #export PATH=$PATH:$JAVA_HOME/bin
     #cd CertGeneration
     #chmod 755 build-stores.sh
     #sh build-stores.sh
     #<Type 2><Press Enter>
     ```

3. Enter a passphrase for the keystore. This field requires a minimum of 6 characters.
4. Press **ENTER**.

Both options generate the three required files to start Application Defender service.

Files generated:

- keystore.jks
- truststore.jks
- itemstore.jks

3. Update the Application Defender Properties File

Update the `appdefender.properties` file with the required parameters as shown in the following example.

```
deploy:single
lb_host:10.100.100.100
apps_host:[['1','10.100.100.100','applications']]
infrastructure_host:[['1','10.100.100.101','infrastructure']]
apps_host_mac_address:00-aa-bb-cc-dd-ee
appdefender_registry:appdefender
defender_logs:/opt/defenderlogs
defender_data:/opt/defenderdata
initial_user_email:test@hpe.com
initial_user_first_name:Applicatioin
initial_user_last_name:Defender
initial_tenant_domain:defender.com
initial_tenant_name:SingleInstance
mail_from:single@defender.com
mail_host:mail.server.com
mail_port:25
mail_username:mailusername@hpe.com
mail_password:mailpassword
postgres_ip:10.100.100.121
postgres_dbname:appdefender
postgres_user:username
postgres_password:postgrespassword
vertica_ip:10.100.100.131
vertica_dbname:docker
vertica_user:username
vertica_password:verticapassword
keystore_path:/opt/hpad/serverkeys/keystore.jks
keystore_password:keystorepassword
truststore_path:/opt/hpad/serverkeys/truststore.jks
truststore_password:keystorepassword
itemstore_path:/opt/hpad/serverkeys/itemstore.jks
itemstore_password:keystorepassword
license_file_dir:/opt/hpad/license
haproxy_config_location:/opt/hpad/haproxy/haproxy.tmpl
docker_folder:/home/defender/onprem/docker
docker_version:1.12+
version:17.1
syslog:disable
```

Note: The `appdefender.properties` file is stored as clear text. Since the file includes user names and passwords, you can encrypt sensitive `appdefender.properties` data. For instructions, see "[Optional] Encrypting Sensitive Values " on page 20

4. Generate Compose Files, Environment Files, Template File, and Database Creation Script

Update the contents of the `appdefender.properties` file with the required details. Execute the `generate-compose-yaml.py` script with the `-h` parameter to display help contents and parameter definitions:

```
#chmod 755 generate-compose-yaml.py
#python generate-compose-yaml.py -h
#python generate-compose-yaml.py appdefender.properties
```

This script generates the following files in the appdefender directory, and these files start the Application Defender service.

Application Defender Directory Files

File	Definition
`applications.env`	Contains the environment variables used to start Fortify Application Defender components.
`applications.yml`	Contains the service description to start Fortify Application Defender application containers.
`create-db.sql`	Used to create the Fortify Application Defender database.
`haproxy.tmpl`	A temporary file used to create the haproxy configuration file.
`infrastructure.env`	Contains the environmental variables used to start Fortify Application Defender infrastructure components.
`infrastructures.yml`	Contains the service description to start Application Defender application containers.
`postgres.yml`	If a Postgres container is being used to start the Application Defender service, this file contains information used in bringing up the postgres container.
`optional.yml`	The `optional.yml` file contains service description for optional services like the storm_ui service.
`privacy_scripts.env`	An optional file created if you encrypted sensitive property values. Includes encryption key used to decrypt properties.
`privacy_scripts.yml`	An optional file created if you encrypted sensitive property values.

Application Defender Directory Files, continued

File	Definition
Shell script for individual hosts	In a single instance installation this script copies corresponding shell scripts to their respective hosts and runs them.

5. Log in to your Docker Hub Account

Use your Docker Hub credentials to log in to your account. Execute the following command on both application and infrastructure hosts:

```
#docker login
```

6. Start the Postgres Database

There are two ways to start the Postgres database, as a Standalone Postgres Database or the Postgres Container. Fortify recommends using the Postgres Container.

Select the option needed for your configuration:

Standalone Postgres Database

1. Copy the `create-db.sql` file to the database host.
2. Log in to the Postgres database server using valid credentials.
3. Execute `create-db.sql` to create Application Defender database.

Postgres Container

```
#docker-compose -f postgres.yml up -d
```

7. Execute Database Migrations to Create the Initial Schema

Copy `infrastructures.yml` and `infrastructures.env` files to your infrastructure host in a folder (for example: /opt/defender/). This container creates the database schemas required for App Defender.

Execute following command from infrastructure host to start database migrations:

```
#docker-compose -f infrastructures.yml up -d db_migrations
```

Starting the database migrations container also starts the Cassandra container.

8. [Optional] Start rsyslog_defender service

Start rsyslog_defender container to consume Application Defender serivice logs.

```
#docker-compose -f applications.yml up -d rsyslog_defender
```

9. Start the Customer User Interface (UI) to Create Initial Application Defender Version

Execute the following command from application host:

```
#docker-compose -f applications.yml up -d ui_customer
```

10. Delete the zookeeper folder.

The Zookeeper folder is located in the default AppDefender folder (specified in `appdefender.properties`).

11. Start Infrastructure Components

Execute the following command from infrastructure host:

```
#docker-compose -f infrastructures.yml up -d
```

12. Start All Application Components

Execute the following command from application host:

```
#docker-compose -f applications.yml up -d
```

13. Reset Password

Navigate to `https://<application_url>:8443` and reset the password.

[Optional] Encrypting Sensitive Values

By default, .yml files, .env files, and the `appdefender.properties` files are created in clear text. The `appdefender.properties` file inludes your Postgress and Vertica credentials and other sensitive data. If you want to encrypt the sensitive values stored in these files, follow these steps:

1. Complete privacy-scripts.env with sensitive properties and encryption key:

`VERTICA_USER=`

VERTICA_PASSWORD=

POSTGRES_USER=

POSTGRES_PASSWORD=

DB_KEY=

PROPERTIES_KEY=

`DB_KEY` and `PROPERTIES_KEY` are used for encrypting sensitive values. These values should contain only alphanumeric symbols and the following special characters: !.,><_

Possible key lengths: 16, 24, 32.

Important! Maintain keys in a safe place.

2. Run the docker container to encrypt the properties.

The encrypted properties will be displayed on the console. You should store them locally for later use.

```
docker-compose -f privacy-scripts.yml up
```

3. Fill appdefender.properties.

Insert encrypted values from Step 2 into:

```
postgres_user:
postgres_password:
vertica_user:
vertica_password:
db_key:
```

4. Create new property 'properties_key' and fill it with the value from `PROPERTIES_KEY` in `privacy-scripts.env`

5. If this is a new App Defender installation and not an upgrade, the other properties will be empty and should be completed (e.g. postgres_ip, postgres_dbname, ...)

6. Return to Step 3 of the Single Instance Installation to complete the process.

After completing the process, follow the Post Encryption Clean-Up steps below.

Post Encryption Clean-up

If you followed the procedure to encrypt sensitive values, you will need to remove sensitive information from your hard drive to complete the process.

1. Delete the disk properties encryption key.

2. Delete appdefender/`properties_encryption.env`.

3. Delete `properties_key` field from the `appdefender.properties` file. This field is no longer necessary as it is stored in RAM in running containers.

Redeployment: Encrypted Values Used

To redeploy an installation that includes encrypted values:

1. Add `properties_key` field to `appdefender.properties` file.
2. Regenerate docker-compose files (including appdefender/properties_encryption.env).
3. After redeployment, delete encryption keys again.

Clustered Installation

A clustered installation of Fortify Fortify Application Defenderrequires at least six nodes; three application nodes and three infrastructure nodes. The advantage of a clustered installation include:

- Fault Tolerance
- Replication
- Rolling restart for upgrades
- Horizontal scaling, which allows application components to handle more users and agents
- Faster event processing

Note: To install in cluster mode, installers must have experience configuring and managing clustered networks.

Application and Infrastructure Node Requirements

Nodes	Descriptions
Application Nodes (x3)	There must be at least three application nodes. Application nodes are used to start the following application components: • edge • command-channel • haproxy - The host with which agents and users interact. Designate one node for haproxy. • scheduler • ui-customer • rsyslog_defender
Infrastructure Nodes (x3)	Infrastructure nodes start components for event processing, alerting, and reporting, You must have at least three nodes for clustering. The generation script, `generate-compose-yaml.py`, generates the required file to configure the Docker daemon and start the Application Defender service from information saved in the `appdefender.properties` file. The following services run on infrastructure nodes: • Zookeeper • Kafka • Storm • Cassandra • Topologies

Application and Infrastructure Node Requirements , continued

Nodes	Descriptions
	• Database migrations

Configure an Application Defender Cluster

To configure an Application Defender cluster, complete the steps in this section in order.

1. ## Copy the Installation Package to a Linux Machine.

   Designate one node as the Swarm Master. The swarm master is used to control the cluster and deploy containers across the Swarm Cluster. Copy the entire installation package to any single folder (for example: `/opt/appdefender`).

   Note: You must have read/write/execute privileges to install Application Defender.

2. ## Generate the Java Keystore

   a. Run the `build-stores.sh` script.
   b. Enter a server certificate option at the prompt.
      - Enter **1** for Self-signed Server Certificate Generation.

        Self-signed certificate scripts are used with trial or pilot installations.

        ```
        #export PATH=$PATH:$JAVA_HOME/bin
        #cd CertGeneration
        #chmod 755 build-stores.sh
        #chmod 755 server-root-self-signed.sh
        #sh build-stores.sh
        #<Press Enter>
        ```

      - Enter **2** for Server Certificate Signed by valid Certificate Authority (CA).

        If you use certificates signed by a third-party CA, copy server certificate signed by CA (`server.crt`), server private key (`server.key`), CA intermediate Root cert (`server.int.crt`) and CA Root cert (`server.root.crt`) under third party folder.

        Note: Use the filenames provided in parentheses.

        ```
        #export PATH=$PATH:$JAVA_HOME/bin
        #cd CertGeneration
        #chmod 755 build-stores.sh
        ```

```
#sh build-stores.sh
#<Type 2><Press Enter>
```

c. Enter a passphrase for the keystore. This field requires a minimum of 6 characters.

d. Press **ENTER**.

Both options generate the three required files to start Application Defender service.

Files generated:

- keystore.jks
- truststore.jks
- itemstore.jks

3. Complete the Fortify Application Defender Properties File.

Update the `properties` file with the required parameters as shown in the following example.

```
deploy:cluster
lb_host:10.100.100.101
apps_host:[['1', '10.100.100.101', 'c02app01'],['2', '10.100.100.102', 'c02app02'], ['3', '10.100.100.103', 'c02app03']]
infrastructure_host:[['1', '10.100.100.111', 'c02infra01'], ['2', '10.100.100.112', 'c02infra02'], ['3', '10.100.100.113',
'c02infra03']]
apps_host_mac_address:00-aa-bb-cc-dd-ee
appdefender_registry:appdefender
defender_logs:/opt/defenderlogs
defender_data:/opt/defenderdata
initial_user_email:test@defender.com
initial_user_first_name:Applicatioin
initial_user_last_name:Defender
initial_tenant_domain:defender.com
initial_tenant_name:ClusterDefender
mail_from:cluster@defender.com
mail_host:mail.server.com
mail_port:25
mail_username:mailusername@hpe.com
mail_password:mailpassword
postgres_ip:10.100.100.121
postgres_dbname:appdefender
postgres_user:username
postgres_password:postgrespassword
vertica_ip:10.100.100.131
vertica_dbname:docker
vertica_user:dbadmin
vertica_password:verticapassword
keystore_path:/opt/hpad/serverkeys/keystore.jks
keystore_password:keystorepassword
truststore_path:/opt/hpad/serverkeys/truststore.jks
truststore_password:keystorepassword
itemstore_path:/opt/hpad/serverkeys/itemstore.jks
itemstore_password:keystorepassword
license_file_dir:/opt/hpad/license
haproxy_config_location:/opt/hpad/haproxy/haproxy.tmpl
docker_folder:/home/defender/onprem/docker
docker_version:1.12+
version:17.1
syslog:disable
```

Note: The `appdefender.properties` file is stored as clear text. Since the file includes user names and passwords, you can encrypt sensitive `appdefender.properties` data. For information, see "[Optional] Encrypting Sensitive Values " on page 28

4. Generate Compose Files, Environment Files, Template File, and Database Creation Script.

Update the contents of the `appdefender.properties` file with the required details. Execute the `generate-compose-yaml.py` script with the `-h` parameter to display help contents and parameter definitions:

```
#chmod 755 generate-compose-yaml.py
#python generate-compose-yaml.py -h
#python generate-compose-yaml.py appdefender.properties
```

This script generates the following files in the appdefender directory, and these files start the Application Defender service.

Application Defender Directory Files

File	Definition
`applications.env`	Contains the environment variables used to start Fortify Application Defender components.
`applications.yml`	Contains the service description to start Fortify Application Defender application containers.
`create-db.sql`	Use this file to create the Fortify Application Defender database. If you are installing a single instance deployment, copy and execute this file to Postgres host.
`haproxy.tmpl`	Copy this file to the application host.
`infrastructure.env`	Contains the environmental variables used to start Fortify Application Defender infrastructure components.
`infrastructure.yml`	Contains the service description to start Application Defender infrastructure containers.
`postgres.yml`	If a Postgres container is being used to start the Application Defender service, copy this file to the Postgres database host.
`privacy_ scripts.env`	An optional file created if you encrypted sensitive property values. Includes encryption key used to decrypt properties.
`privacy_ scripts.yml`	An optional file created if you encrypted sensitive property values.
Shell script for individual hosts	Clustered Installation - Additional shell scripts are generated under **hostShellScripts** folder. The scripts generated are created using the hostname provided in appdefender.properties. Copy and execute script on corresponding hosts.

5. Create Swarm Cluster

   The generation script creates a shell script that configures the Docker daemon and starts the consul and swarm container. Execute the generated shell scripts on their respective docker nodes.

Following is a summary of the commands executed by shell script on individual host to create and start Swarm Cluster.

For a link to Docker documentation, see Additional References.

6. Configure Swarm Cluster

   After creating your Swarm cluster, you will need to configure it. The following sections provide the information you will need to configure your Swarm cluster for use with Application Defender.

7. Configure Docker Daemon

   The generation script creates a shell script, based on the settings provided in `appdefender.properties` that configures the Docker daemon. Options configured using the shell script start the swarm cluster daemon. The following sample shows the configuration for one host: Pl

   Sample Configuration: docker.conf (folder : /etc/systemd/system/docker.service.d/docker.conf

   ```
   [Service]
   ExecStart=
   ExecStart=/usr/bin/docker daemon -H fd:// -H tcp://10.100.100.111 -H
   unix:///var/run/docker.sock --cluster-
   store=consul://10.100.100.111:8500 --cluster-
   advertise=10.100.100.111:2375 -g /var/lib/docker --label
   com.defender.server="infrastructure"
   ```

   After updating Docker configuration, the shell script reloads the configuration and restarts the Docker daemon.

8. Start the Discovery Backup

   The discovery backend is used to authenticate Swarm managers and nodes with the cluster. To start a Swarm cluster you need to set up a discovery backend. The docker.conf shell script used, also starts a docker container for consul.

   Sample Run Command:

   ```
   docker run -d --name=consul_dockernode01 -v /opt/newConsul/:/data -e
   constraint:node==dockernode01 -p 8300:8300 -p 8301:8301 -p
   8301:8301/udp -p 8302:8302 -p 8302:8302/udp -p 8400:8400 -p 8500:8500 -
   p 172.17.0.1:53:53/udp appdefender/consul -server -advertise
   10.100.100.111 -join 10.100.100.111 -bootstrapp;
   ```

9. Start Swarm Agent on Docker Node

   Once discovery back-end is running, a shell script starts the swarm agent on each Docker node.

   Sample Run Command:

   ```
   docker run -d --name=swarm_dockernode01 --addr=10.100.100.111:2375
   consul://10.100.100.111/swarm;
   ```

10. Swarm Manager Setup

Start the Swarm Manager with replication.

Sample Run Command:

```
docker run -d -p 10.100.100.111:3375:2375 --name=manager_dockernode01
swarm manage --replication --advertise 10.100.100.111:3375
consul://10.100.100.111:8500/swarm
```

11. Log in to your Docker Hub Account

    Use your Docker Hub credentials to log in to your account. Execute the following command on both application and infrastructure hosts:

    ```
    #docker login
    ```

    This command needs to be executed from the server where swarm manager is started and all compose files are stored.

12. Start the Postgres Database

    There are two ways to start the Postgres database, as a Standalone Postgres Database or the Postgres Container. Fortify recommends using the Postgres Container.

    Select the option needed for your configuration:

    Standalone Postgres Database

    Execute `create-db.sql` to create Application Defender database after login to Postgres Database server with valid credentials.

    Postgres Container

    Copy `postgres.yml` and `create-db.sql` to the Docker host where you plan to run the Postgres container and execute following command:

    ```
    #docker-compose -f postgres.yml up -d
    ```

13. Execute Database Migrations to Create the Initial Schema

    Run all the Application Defender related docker-compose commands after pointing to the designated port for Swarm Manager, for example, port 3375 in this case.

    ```
    #export DOCKER_HOST=<ip-address>:3375
    ```

    The `db_migrations` creates database schemas required for Application Defender and starts Cassandra cluster containers. Execute following command from swarm manager:

    ```
    #docker-compose -f infrastructures.yml up -d db_migrations
    ```

14. [Optional] Start rsyslog_defender service to consume Application Defender service logs:

    ```
    #docker-compose -f applications.yml up -d rsyslog_defender
    ```

15. Copy License, Keystore and haproxy.tmpl

    To distribute application containers across hosts.

a. Copy required license and keystore files on all the hosts designated for applications in the folders mentioned in `appdefender.properties`.
b. Copy the generated `haproxy.tmpl` file to the host designated as the Load Balancer (`lb_host`) under the folder configured in `appdefender.properties`.

16. Start the Customer User Interface (UI) to Create Initial Application Defender Version

    Execute the following command from the application host:

```
#docker-compose -f applications.yml up -d ui_customer
```

17. Start Infrastructure Components

    Execute the following command from the infrastructure host:

```
#docker-compose -f infrastructures.yml up -d
```

18. Start All Application Components

    Execute the following command from the application host:

```
#docker-compose -f applications.yml up -d
```

19. Reset Password

    Navigate to `https://<application_url>:8443` in your browser and reset the password for the first user using the reset password link.

[Optional] Encrypting Sensitive Values

By default, .yml files, .env files, and the `appdefender.properties` files are created in clear text. The `appdefender.properties` file inludes your Postgress and Vertica credentials and other sensitive data. If you want to encrypt the sensitive values stored in these files, follow these steps:

1. Complete privacy-scripts.env with sensitive properties and encryption key:

VERTICA_USER=

VERTICA_PASSWORD=

POSTGRES_USER=

POSTGRES_PASSWORD=

DB_KEY=

PROPERTIES_KEY=

`DB_KEY` and `PROPERTIES_KEY` are used for encrypting sensitive values. These values should contain only alphanumeric symbols and the following special characters: !.,><_

Possible key lengths: 16, 24, 32.

Important! Maintain keys in a safe place.

2. Run the docker container to encrypt the properties.

The encrypted properties will be displayed on the console. You should store them locally for later use.

```
docker-compose -f privacy-scripts.yml up
```

3. Fill appdefender.properties.

Insert encrypted values from Step 2 into:

```
postgres_user:
postgres_password:
vertica_user:
vertica_password:
db_key:
```

4. Create new property 'properties_key' and fill it with the value from `PROPERTIES_KEY` in `privacy-scripts.env`

5. If this is a new App Defender installation and not an upgrade, the other properties will be empty and should be completed (e.g. postgres_ip, postgres_dbname, ...).

6. Return to Step 3 of Configure an Application Defender Cluster to complete the process.

After completing the process, follow the Post Encryption Clean-Up steps below.

Post Encryption Clean-up

If you followed the procedure to encrypt sensitive values, you will need to remove sensitive information from your hard drive to complete the process.

1. Delete the properties encryption key.

2. Delete appdefender/`properties_encryption.env`.

3. Delete `properties_key` field from the `appdefender.properties` file. This field is no longer necessary as it is stored in RAM in running containers.

Redeployment: Encrypted Values Used

To redeploy an installation that includes encrypted values:

a. Add `properties_key` field to `appdefender.properties` file.
b. Regenerate docker-compose files (including appdefender/properties_encryption.env).
c. After redeployment, delete encryption keys again.

Advanced Installation Notes

The advanced installation notes are intended for experienced Application Defender users.

SMTP Email Server Authentication

If you want to access the SMTP email server using authentication, provide the appropriate values for mail_username and mail_password in the `appdefender.properties` file before you run the `generate-compose-yaml.py` script:

mail_username: `<abc@abc.com>`

mail_password: `<password>`

If you do not want to authenticate mail server leave these fields blank.

Java Keystore

This section contains the following topics:

All Application Defender communication takes place on a secure channel. To get this working, Application Defender needs three keystore files. Trial and pilot installations should use the Self-signed Server Certificate script. If you use certificates signed by a third-party use a Server Certificate Signed by Valid Certificate Authority.

Self-signed Server Certificate

The script provided in the package gives an option to create a self-signed server certificate chain and agent certificate chain to be used with Application Defender.

The included scripts:

`server-root-self-signed.sh` - This script generates the certificate chain for the Application Defender server. Execute this script only when creating a self-signed server certificate.

`build-stores.sh` - This script generates the agent certificate chain and the final java keystore files used for the Application Defender service. After executing this script, the following jks files are generated in the CertGeneration folder:

- `keystore.jks` - Contains the server certificate chain which includes the Intermediate ROOT certificate and ROOT certificate.
- `truststore.jks` - Contains trustedCertEntry for the Intermediate agent, ROOT agent, and server ROOT certificate.
- `itemstore.jks` - Contains the agent certificate chain , trustedCertEntry for ROOT certificate and trustedCertEntry for the ROOT agent.

Server Certificate Signed by Valid Certificate Authority

If you are using a certificate signed by a valid CA, copy the signing authority's ROOT certificate and Intermediate ROOT certificate to `CertGeneration>thirdparty` folder and rename the files if necessary:

- The server certificate should be named `server.crt` (example: `qa_appdefender_com.crt` renamed to `server.crt`)
- The server Private key should be named `server.key` (example: `qa_appdefender_com.crt` renamed to `server.key`)
- The CA Intermediate ROOT certificate should be named `server.int.crt` (example: `Digicert_int.crt` renamed to `server.int.crt`)
- The CA ROOT certificate should be named `server.root.crt` (example: `Digicert_root.crt` renamed to `server.root.crt`)

All Docker-Compose Files

The `generate-compose-yaml.py` script generates the docker-compose files, database creation scripts, and the proxy template file. These files are used to start the Application Defender service. This script uses the `appdefender.properties` file as input and generates the files needed to start the Application Defender service.

Update the `appdefender.properties` file

The following describes the `appdefender.properties` file settings that need to be updated:

Docker Compose Setting Field Descriptions

Setting	Description
`appdefender_registry`	The Docker registry that stores Application Defender images
`apps_host`	List of the nodes designated for application components. Node information includes following properties: • Number assigned to the host (numeric range: **1-255**).

Docker Compose Setting Field Descriptions, continued

Setting	Description
	• IP Address • Hostname
mail_username	Required for SMTP authentication. For more inforation, see page 29.
mail_password	Required for SMTP authentication.
`apps_host_mac_address`	The MAC address of the host machine running docker for the applications. Use the MAC address provided for the Fortify Application Defender license download.
`defender_data`	Directory on individual hosts to persist application defender data.
`defender_logs`	Directory on individual hosts to persist application defender logs.
`deploy`	Based on the kind of deployment, this setting should be either single or cluster.
`infrastructure_host`	List of the nodes designated for infrastructure components. Node information includes the following properties: • Number assigned to the host (numeric range: **1-255**). • IP Address • Hostname
`initial_tenant_domain`	Domain of the tenant, for example, microfocus.com.
`initial_tenant_name`	Name of the tenant, for example, Micro Focus.
`initial_user_email`	Email address of the first user in the system.
`initial_user_first_name`	Name of the first user in the system.
`initial_user_last_name`	Surname of the first user in the system.
`lb_host`	Load balancer host. The front end node which provides the interface for users and agents to interact with.

Docker Compose Setting Field Descriptions, continued

Setting	Description
`logs_data_dir`	Directory on host machine that contains the logs and data.
SMTP server	Application Defender needs an SMTP server to send emails to users: `mail_from` - A valid email address as the sender of all automated emails. `mail_host` - A valid mail host address. `mail_port` - Default port = **25** `mail_username:` - Required for SMTP authentication. For more information on SMTP Authentication, see Advanced Installation Notes. `mail_password:` - Required for SMTP authentication.

Postgres Container	A Docker container for the Postgres database is created unless the following properties are set to configure Application Defender to use an external Postgres database:
	`postgres_ip`
	`postgres_dbname`
	`postgres_user`
	`postgres_password`

Vertica Database	A Vertica database is required. Use the properties below to configure Application Defender to use the Vertica database:
	`vertica_ip`
	`vertica_dbname`
	`vertica_user`
	`vertica_password`

Keystore, Truststore, and Itemstore Configuration	Configure a keystore, truststore, and itemstore must be configured as per the Administrator Guide:
	`keystore_path`
	`keystore_password`
	`truststore_path`
	`truststore_password`
	`itemstore_path`

Docker Compose Setting Field Descriptions, continued

Setting	Description
	`itemstore_password`
License File Directory	Provide a path to license file directory. `license_file_dir`
`haproxy_config_location`	High Availability Proxy Configuration Location - Directory where `haproxy.tmpl` file is copied to the load balance host (`lb_host`). This is an optional field.
version	Version refers to the Application Defender docker containers hosted on the registry. This is an optional field.
`docker_folder`	Folder where the docker files are saved. This is an optional field.
docker_version	Version of docker used to install App Defender. This value should be "1.12+" unless you are installing a previous version. This establishes the docker start command as "dockerd -D" or " docker daemon -H".
Syslog: (enable/disable)	This setting redirects application logs to the rsyslog_defender container.

Vertica Database

A columnar database used by Application Defender to store event data.

Vertica Database Port Descriptions

Service Defined	Host Port Number	Container Port Number	Service Description
vertica	5433	Standalone	Used as a persistent data store for security and monitor events.

Postgres Database (Optional)

Use this option if you plan to use a Postgres container instead of the separate database installation.

Execute following command on the Postgres host:

```
#docker-compose -f postgres.yml up -d
```

Postgres Database Port Descriptions

Service Defined	Host Port Number	Container Port Number	Service Description
postgres	5432	5432	Relational database used by Application Defender to store application state, agent state, user data and agent binaries.

Infrastructure Virtual Machine Services

To start all the infrastructure components on infrastructure virtual machine, execute the following command:

```
#docker-compose -f infrastructure.yml up -d
```

Infrastructure Virtual Machine Port Descriptions

Service Defined	Host Port	Container Port Number	Service Description
cassandra	9042, 7000	9042, 7000, 7001, 7199, 9160	A highly available distributed database used by Application Defender to store intermediate data for alerting.
kafka	9092	9092	A stateless distributed queue used for event stream processing.
zookeeper	2181, 2888, 3888	2181, 2888, 3888	A service for maintaining configuration, naming, distributed synchronization, and group services used by Kafka.
storm_ nimbus	6627	6627	A distributed real-time stream processing technology. Application Defender currently uses five Storm topologies for notifications, reporting, alerting, and writing events to Vertica and Reconciliation.
storm_ supervisor		6700, 6701, 6702, 6703	Storm

Infrastructure Virtual Machine Port Descriptions, continued

Service Defined	Host Port	Container Port Number	Service Description
storm_ui	8080	8080	Provides a clean UI to check the status of storm processes.
topologies	n/a	n/a	n/a

Application Virtual Machine Services

To start all the application components on applications virtual machine, issue the following command:

```
#docker-compose -f applications.yml up -d
```

Application Virtual Machine Port Descriptions

Service Defined	Host Port Number	Container Port Number	Service Description
backend_ jobs		8080	Component used to schedule reports.
command_ channel	random	8080	Secure communication channel between Application Defender agents and service for exchange of commands.
consul		8300, 8400, 8500, 8600, 8301, 8302	
db_ migrations			Used to migrate existing Application Defender data in case of upgrades or create database schemas in case of new installations.
edge	random	4321	Secure communication channel between Application Defender Agent and Service used by Agents to send events to the service.
haproxy		1936, 8443,	Load balancer that provides access to different services

Application Virtual Machine Port Descriptions, continued

Service Defined	Host Port Number	Container Port Number	Service Description
		8444, 4321	provided by Application Defender.
registrator			
ui_ customer	random	8080	The website used by all Application Defender users to access Protect, Manage, Messaging and Alerting functionality.
Rsyslog_ defender	514, 1999	514, 1999	rsyslog_defender container used to consume Application Defender service logs.

Scaling Application Defender On-Premise Services

Docker Compose allows you to scale different services based on load and number of agents:

- To scale instances of Customer UI execute following command:

```
#docker-compose -f applications.yml scale ui_customer=2
```

- To scale instances of Command Channel:

```
#docker-compose -f applications.yml scale command_channel=2
```

- To scale process for backend jobs:

```
#docker-compose -f applications.yml scale backend_jobs=2
```

- To scale instances of Edge:

```
#docker-compose -f applications.yml scale edge=2
```

Application Defender System Hardening

Application Defender is a complex, multi-process solution with a big-data architecture. The distributed nature of the solution increases the attack surface, especially to malicious insiders. In addition to proper patch management policies, strict access controls, and secure server configurations, Fortify recommends the following to reduce your attack surface and increase security of your Applicaton Defender deployment:

change to:

- Protect the `appdefender.properties`, `applications.env`, and `infrastructures.env` files by restricting who can access them and read their contents. Fortify recommends at least file system level access controls to ensure only authenticated users with sufficient entitlement can access these files.
- The Application Defender installation provides a container with the Storm user interface to monitor storm processes as well as perform topology administration. Malicious users with access to the Storm UI can disable storm topologies and prevent event storage, analysis, or visualization in the Application Defender Server. Fortify recommends disabling storm_ui if you are not using it:

```
#docker stop storm_ui
```

- Application Defender has a three tier architecture:
  a. Application - Presentation tier
  b. Infrastructure - Logic tier
  c. Databases - Data tier

  Users and agents only interact with the application layer. Fortify recommends that you configure your firewall to only provide access to these machines.
- Please follow the instructions provided by Docker to secure your Docker daemon and secure Swarm Cluster deployment. For more information, see Additional References.

Logging Policy

The logging policy table provides information on each of the Application Defender services.

Svc #	Docker Image	Data Location	Log	Container Log Rotation Policy	Internal Daemon Rotation Policy
1	ui-customer		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
2	ui-internal		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
3	backend-jobs		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
4	command-channel		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
5	edge		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
6	topologies		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
7	db-migrations		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	

8	Zookeeper	$defender_ data/zookeeper	$defender_logs/zookeeper	max-size: "50m"max-file: "9"	autopurge.purgeInterval=24 autopurge.snapRetainCount=10
9	Kafka	defender_data/kafka	$defender_logs/kafka	max-size: "50m"max-file: "9"	log.retention.hours=168
10	Storm-nimbus		$defender_logs/storm_nimbus	max-size: "50m"max-file: "9"	100 MB 9 Files
11	Storm-supervisor		$defender_logs/storm_supervisor	max-size: "50m"max-file: "9"	100 MB 9 Files
12	Storm-ui		$defender_logs/storm_ui	max-size: "50m"max-file: "9"	100 MB 9 Files
13	Cassandra	$defender_ data/cassandra	$defender_logs/cassandra	max-size: "50m"max-file: "9"	20 MB 20 files
14	Consul	$defender_data/consul	Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
15	Registrator		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
16	haproxy		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
17	Swarm		Docker Container Folder e.g. /home/defender/docker/containers/<container_id>/	max-size: "50m"max-file: "9"	
18	Postgres			max-size: "50m"max-file: "9"	
19	Vertica			max-size: "50m"max-file: "9"	
20	Syslog			max-size: "50m"max-file: "9"	

Additional References

For assistance in configuring the recommended hardware components in your Application Defender on Premise installation see:

Software Component	Documentation URL
Docker Compose	https://docs.docker.com/compose/install/
Docker Control and configure with systemd	https://docs.docker.com/engine/admin/systemd/
Docker Engine	https://docs.docker.com/engine/installation/ubuntulinux/
Docker Hub Account	https://hub.docker.com/
Docker Protect the daemon socket	https://docs.docker.com/engine/security/https/
Docker Swarm Configuration	https://docs.docker.com/swarm/plan-for-production/
Docker Swarm for TLS	https://docs.docker.com/swarm/configure-tls/
Postgres	http://www.postgresql.org/docs/9.4/static/index.html
Vertica	Version 8.1.x: https://my.vertica.com/docs/7.1.x/HTML/#Authoring/InstallationGuide/Other/InstallationGuide.htm%3FTocPath%3DInstallation%2520Guide%7C_____0 https://my.vertica.com/docs/Hardware/HP_Vertica%20Planning%20Hardware%20Guide.pdf Version 9.1.x: https://www.vertica.com/documentation/vertica/9-1-x/

Send Documentation Feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this computer, click the link above and an email window opens with the following information in the subject line:

Feedback on On-Premises Installation Guide (Fortify Application Defender 18.20)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to FortifyDocTeam@microfocus.com.

We appreciate your feedback!