Production Deployment guideline

This document provide the steps for APK production deployment.

WSO2 APK can be configured through `values.yaml` file. Please refer to Customize Configurations for information on how to use a customized values file for APK deployment. When deploying WSO2 APK in a production environment, we strongly recommend following these guidelines.

Choose the correct deployment pattern

Please refer this document on how to choose the correct pattern for you environment.

Change the hostnames and vhosts

By default, APK uses wso2.com for its hostnames and vhosts for the gateway. You need to change these values to your own domain, which you plan to use for production. The following values.yaml values should be modified:

Configuration	Description
`wso2.apk.listener.hostname`	This configuration is used to specify the hostname for listening to API requests related to the APK system. It should be set to your desired domain for production.
`wso2.apk.dp.gateway.listener.hostname`	This configuration is used to specify the hostname for listening to API requests made by users deploying their APIs. It should be set to your desired domain for production.
`wso2.apk.dp.configdeployer.vhosts`	This configuration is utilized by the Config Deployer Service to create API Custom Resources (CRs) in response to user API creation requests. It should be set to the appropriate value for production use.

By modifying these configurations, you can ensure that APK operates with the correct hostnames and vhosts for your production environment.

For example if you want to deploy a production environment and you have a domain name example.com and you want to expose your API's through prod.gw.example.com and expose APK system APIs through prod.apk.example.com then

- wso2.apk.listener.hostname: 'prod.apk.example.com'
- wso2.apk.dp.gateway.listener.hostname: 'gw.example.com'
- wso2.apk.dp.configdeployer.vhosts: [{"hosts": ["gw.example.com"],"name":"prod","type":"production"}]

For further clarification on the keys, please refer to the description and default values here

Change certificates

The default APK deployment uses a self-signed certificate for APK components. Default APK configuration installs cert-manager in the cluster.

For a production environment, it is recommended to use CA-validated public certificates for internet-facing services. In APK, certificates are used for servers and listeners. Listeners are responsible for exposing services to the internet, while servers are not directly accessible from the internet. In a production environment, it's crucial to configure CA-validated `public` certificates for listeners. Non-public or self-signed certificates can be used for servers, as these server names are internal. Let's explore how to configure these certificates.

Listeners	Description and hostnames
Gateway listener	Listens for for API invocation requests. Hostname can be configures through values.yaml's `wso2.apk.dp.gateway.listener.dns` . Default value is [".*gw.wso2.com*","sandbox.gw.wso2.com","prod.gw.wso2.com"]
APK system api listener	Listens for for APK system related requests(Ex: API creation rest request). Hostname can be configures through values.yaml's `wso2.apk.listener.hostname` . Default value is "api.am.wso2.com"

Servers	Hostnames
Adapter server	`<helm-installation-name>-adapter-service.<namespace-name>.svc` , `<helm-installation-name>-adapter-service.<namespace-name>.svc.cluster.local`

Servers	Hostnames
Common controller server	`<helm-installation-name>-common-controller-service.<namespace-name>.svc` , `<helm-installation-name>-common-controller-service.<namespace-name>.svc.cluster.local`
Config deployer server	`<helm-installation-name>-config-ds-service.<namespace-name>.svc` , `<helm-installation-name>-config-ds-service.<namespace-name>.svc.cluster.local`
Enforcer server	`<helm-installation-name>-enforcer-service.<namespace-name>.svc` , `<helm-installation-name>-enforcer-service.<namespace-name>.svc.cluster.local`
Gateway server	`<helm-installation-name>-gateway-service.<namespace-name>.svc` , `<helm-installation-name>-gateway-service.<namespace-name>.svc.cluster.local`
Ratelimitter server	`<helm-installation-name>-ratelimiter-service.<namespace-name>.svc` , `<helm-installation-name>-ratelimiter-service.<namespace-name>.svc.cluster.local`

1. Use cert manager

By default, APK installs cert manager in your cluster and employs a SelfSigned issuer for certificate validations. To utilize cert manager for handling the certificates, you will need to create Issuers. Choose the type of Issuer you are going to use for listeners and servers, and create the Issuers in accordance with the cert-manager documentation document. You will need to create two issuers: one for listeners and one for servers. Once created, update the values.yaml configuration as follows.

```
certmanager:
  listeners:
    issuerName: "<issuer-name-created-for-listeners>"
    issuerKind: "ClusterIssuer" # or "Issuer" Refer to cert-manager's issuer doc
  servers:
    issuerName: "<issuer-name-created-for-servers>"
    issuerKind: "ClusterIssuer" # or "Issuer" Refer to cert-manager's issuer doc
```

2. Use the certificate files

Prerequisites

For all the components(Listeners and servers) prepare the following required files.

1. TLS certificate verified by a Ceriticate Authority (tls.crt)
2. Private key associated with the TLS certificate(tls.key)
3. Certificate Authority's (CA) root certificate(ca.crt)

For each component create a secret in the same namespace as APK is deployed with the following key-value pairs:

- tls.crt - Base64 encoded value of tls.crt file
- tls.key - Base64 encoded value of tls.key file
- ca.crt - Base64 encoded value of ca.crt file

You can use the following command to create the secret from the files

```
kubectl create secret generic <SECRET_NAME> --from-file=tls.crt=path/to/tls.crt --from-file=tls.key=path/to/tls.key --from-file=ca.crt=path/to/ca.crt -n <NAMESPACE>
```

- To update the gateway listener certificates, update the following values.yaml config

```
wso2:
 apk:
  dp:
   gateway:
    listener:
     secretName: <created-secret-name-for-gateway-listener>
```

- To update the APK system listener certificates, update the following values.yaml config

```
wso2:
 apk:
  listener:
   secretName: <created-secret-name-for-apk-system-listener>
```

- To update the APK system servers certificates, update the following values.yaml config

```
configs:
 tls:
  secretName: "<Name of the created secret>"
  certKeyFilename: "tls.key"
  certFilename: "tls.crt"
  certCAFilename: "ca.crt"
```

Servers and their `configs` location in the value.yaml

Servers	config location
Adapter server	wso2.apk.dp.adapter.configs.tls
Common controller server	wso2.apk.dp.configdeployer.deployment.configs.tls
Config deployer server	wso2.apk.dp.configdeployer.deployment.configs.tls
Enforcer server	wso2.apk.dp.gatewayRuntime.deployment.enforcer.configs.tls
Gateway server	wso2.apk.dp.gatewayRuntime.deployment.router.configs.tls
Ratelimitter server	wso2.apk.dp.ratelimiter.deployment.configs.tls

Remove default IDP

APK comes with a default IDP which is not production-ready. Disable the default IDP and use a production-ready IDP solution. Please follow these guidelines to setup the production ready IDP

Disable the default idp by changing the following value to `false` in values.yaml

```
idp:
 enabled: false
```

Use a production grade Redis

APK uses a built-in standalone Redis service which is not suitable for production usage. Please use a production grade Redis. You can update the following values to configure the Redis configuration in APK:

- wso2.apk.dp.redis.type
- wso2.apk.dp.redis.url
- wso2.apk.dp.redis.tls
- wso2.apk.dp.redis.auth.certificatesSecret

- wso2.apk.dp.redis.auth.secretKey
- wso2.apk.dp.redis.poolSize

Disable the default redis that comes with the APK deployment - redis.enabled: bool

Protect gateway admin port

APK uses EnvoyProxy in the router implementation. EnvoyProxy offers an administrator interface that can be used to query and modify different aspects of the server. In the production environment, we should disable or restrict access to this port. By default, APK exposes this interface through port `9000`. To disable external access to the port, you can set the following Helm value to `false`: `wso2.apk.dp.gatewayRuntime.deployment.router.adminInterfaceEnabled`

If admin port is enabled, it is critical that access to the administration interface is only allowed via a secure network. It is also critical that hosts that access the administration interface are only attached to the secure network (i.e., to avoid CSRF attacks). This involves setting up an appropriate firewall.