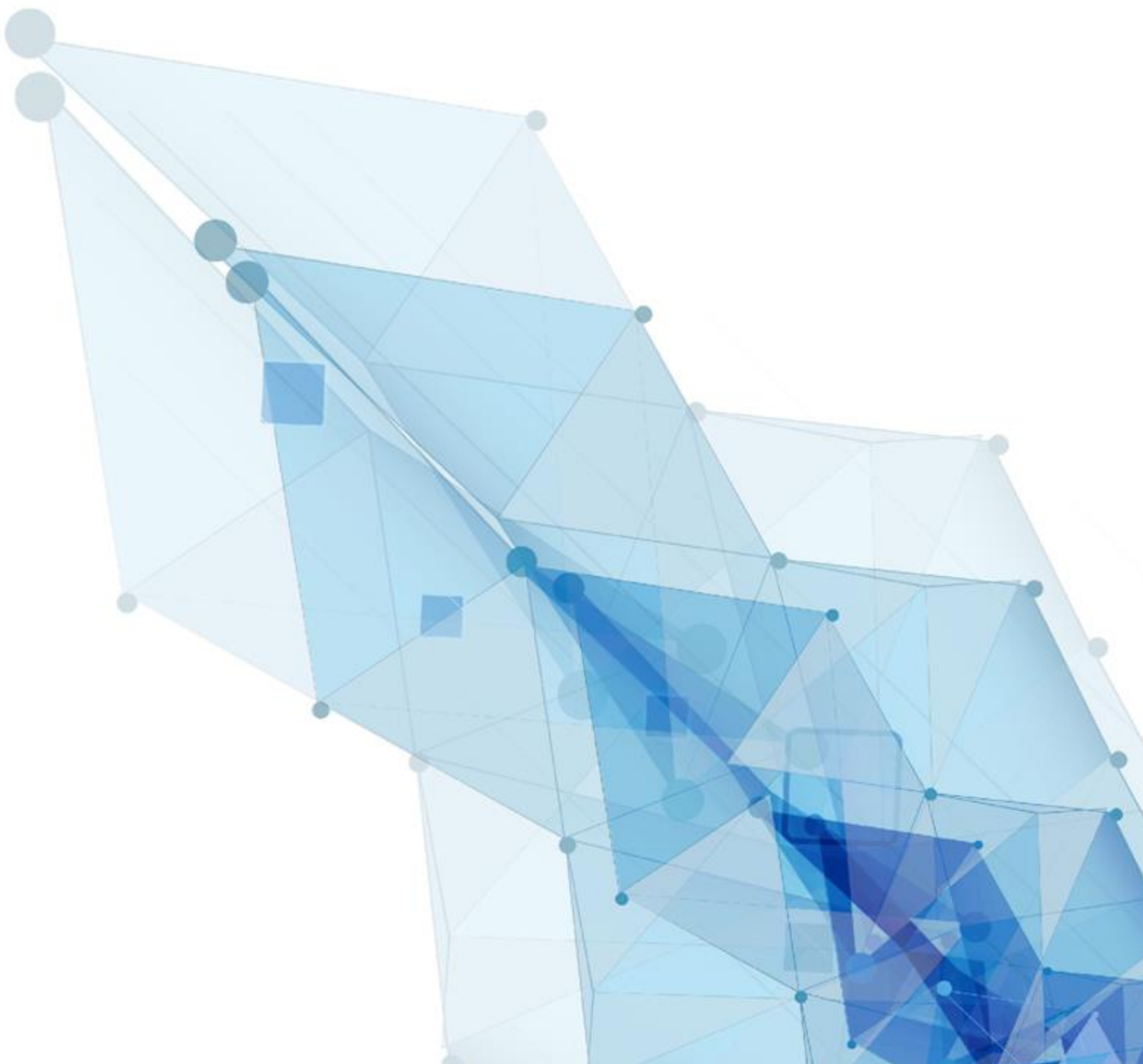
iDNA Applications v3 Migration Guide



Migration Overview

Why Migrate?

As CentOS is nearing it's end-of-life, panagenda provides iDNA for Applications (short: IFA or iDNA) on a new platform: Alma Linux.
iDNA versions 3.0.0 and above will not be compatible with the outdated CentOS appliance. To continue benefiting from the latest and greatest features of iDNA and keeping the base OS updated and secured, all customers will need to migrate to version 3.

What will be migrated?

- Database (including sensor configuration and data)
- User ID, password file and names.nsf
- panagenda license file and DXL files
- general configuration files

Pre-requisites

- New OVA template deployed, and basic setup tasks must have been performed on new appliance: hostname, DNS, partitioning, … (see the Guide to Partitioning in Alma)
- /opt/panagenda/pgdata should at least be the same size as "disk space used" on the same partition on the old appliance, with some additional space for migration files.
    - Rule of thumb: [new pgdata partition size] = [old pgdata partition used]+30%
- /opt/panagenda/appdata should be the same size as on CentOS (DXL files)
- Old IFA installation should be upgraded to >= v2.4.0

Execution Notes

This guide contains code snippets that are meant to be executed on the two appliances. They are marked in recognizable font and coloring and the white text is meant to be copy and pasted:

```
# description of the task to be performed
command to be copy and pasted
```

Prepare New Appliance

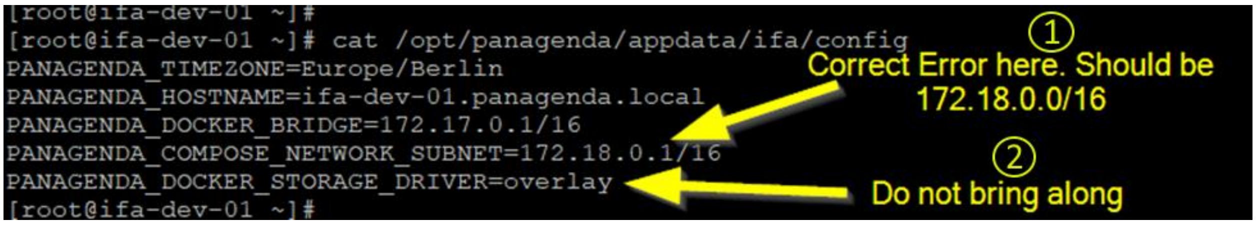
The new appliance will have the same fundamental container configuration that existed on the original iDNA Applications appliance.

```
# Old Appliance: check for custom config
cat /opt/panagenda/appdata/ifa/config
```

This file may not exist, please create it on the new appl if necessary. Attention to host name change!

```
# New Appliance: update custom config
vim /opt/panagenda/appdata/ifa/config
```

You might encounter a mistake here that we previously had in our default config. Docker ignored it in the past, but now flags it and IFA won't start. Please check for this and correct!



```
[root@ifa-dev-01 ~]#
[root@ifa-dev-01 ~]# cat /opt/panagenda/appdata/ifa/config
PANAGENDA_TIMEZONE=Europe/Berlin
PANAGENDA_HOSTNAME=ifa-dev-01.panagenda.local
PANAGENDA_DOCKER_BRIDGE=172.17.0.1/16
PANAGENDA_COMPOSE_NETWORK_SUBNET=172.18.0.1/16
PANAGENDA_DOCKER_STORAGE_DRIVER=overlay
[root@ifa-dev-01 ~]#
```

1. These network config parameters are optional and will not be there in most customer installations. Our default subnets are 192.168.237.0/24 and 192.168.238.0/24 and they are defined in the default config file (/opt/panagenda/appdata/ifa/config). Should this subnet config exist in customer environments though, we need to correct the subnet format and bring the two parameters along (DOCKER_BRIDGE + NETWORK_SUBNET)

2. There might also be a parameter PANAGENDA_DOCKER_STORAGE_DRIVER present. This parameter **SHOULD NOT BE SET** in the config of the new appliance. Just leave the whole line out or remove it if present on the old appliance. The new appliance has a much newer docker version with a new storage driver for performance and OS compatibility.

After updating the config file on the new appliance, use the following command to trigger iDNA to rebuild its container and network configuration.

```
# New Appliance: bring config changes into effect
ifa customize
```

Please check if all containers are started successfully (docker ps). At this point you might see some containers in the state “unhealthy” or continuously restarting themselves. This is normal in this unconfigured appliance state and no cause for concern.

Next you can check the Postgres container to verify if the subnet settings have been applied successfully.

```
# New Appliance: check postgres container
docker inspect panagenda_postgres
```

Near the end of the result, you should see something like the output below. In it, the subnet config and the hosts individual IP come together (x.x.x.100 = postgres in IFA)
`"IPv4Address": "172.18.0.100"` or `"IPv4Address": "192.168.238.100"`

As the last step, let’s stop all containers but the database.

```
# Stop all containers, except postgres
docker stop panagenda_cron panagenda_data_miner panagenda_nginx panagenda_pac
panagenda_node panagenda_dda panagenda_idna panagenda_metabase -t 30
```

Prepare Old Appliance

The following commands are the steps to prepare the old iDNA appliance for migration.

```
# create database user for the migration
docker exec -u postgres panagenda_postgres psql postgres -c "CREATE ROLE
migrator WITH SUPERUSER LOGIN REPLICATION PASSWORD 'panagenda'"
```

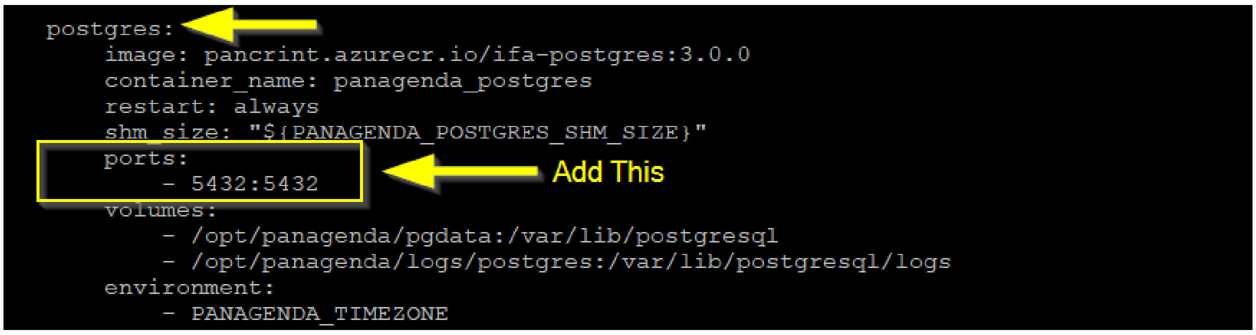
```
# execute function to remove temporary data that does not need to be migrated
docker exec -u postgres panagenda_postgres psql panagenda -c "SELECT
idna_dwh.purge_historical_data(true, true, true)"
```

```
# remove functions that are incompatible with new Postgres v15
docker exec -u postgres panagenda_postgres psql panagenda -c "DROP AGGREGATE IF
EXISTS idna_dwh.array_concat_agg(anyarray); DROP AGGREGATE IF EXISTS
idna_dwh.array_concat_agg(anycompatiblearray); DROP AGGREGATE IF EXISTS
cqx_data.array_concat_agg(anyarray); DROP AGGREGATE IF EXISTS
cqx_data.array_concat_agg(anycompatiblearray);"
```

The Postgres database normally only listens to a local port inside the appliance. For the migration, we need to make the database accessible from the outside network.

```
# open postgres port to the outside world
vim /opt/panagenda/appdata/ifa/docker-compose.yml
```

Tip: copy/paste “ports” block from node container configuration and change the numbers to 5432:5432 (yml files can be picky when it comes to exact indentation and use of space vs tab)

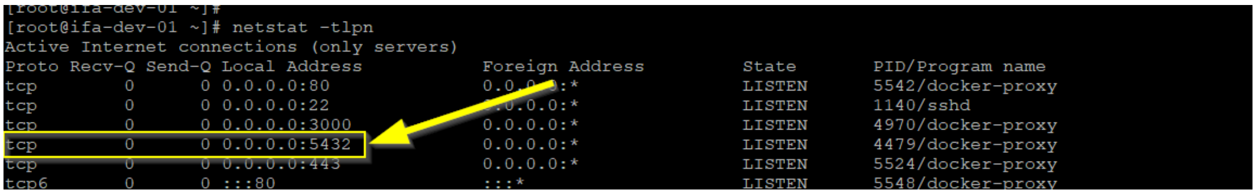


```
  postgres:
      image: pancrint.azurecr.io/ifa-postgres:3.0.0
      container_name: panagenda_postgres
      restart: always
      shm_size: "${PANAGENDA_POSTGRES_SHM_SIZE}"
      ports:
          - 5432:5432
      volumes:
          - /opt/panagenda/pgdata:/var/lib/postgresql
          - /opt/panagenda/logs/postgres:/var/lib/postgresql/logs
      environment:
          - PANAGENDA_TIMEZONE
```

```
# apply docker-compose changes
ifa start
```

```
# check if listening properly
netstat -tlpn
```

This should show 0 0.0.0.0:5432 in state LISTEN



```
[root@ifa-dev-01 ~]#
[root@ifa-dev-01 ~]# netstat -tlpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      5542/docker-proxy
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1140/sshd
tcp        0      0 0.0.0.0:3000            0.0.0.0:*               LISTEN      4970/docker-proxy
tcp        0      0 0.0.0.0:5432            0.0.0.0:*               LISTEN      4479/docker-proxy
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN      5524/docker-proxy
tcp6       0      0 :::80                   :::*                    LISTEN      5548/docker-proxy
```

As the last step, let’s stop all containers but the database.

```
# Stop all containers, except postgres
docker stop panagenda_cron panagenda_data_miner panagenda_nginx panagenda_pac
panagenda_node panagenda_dda panagenda_idna panagenda_metabase -t 30
```

Database Migration

On New appliance

Execute the statements in this chapter from a command line via VNC if possible! (SSH sessions might be interrupted)

```
# Execute the DB migration script
/opt/panagenda/appdata/system/db-migration.sh
```

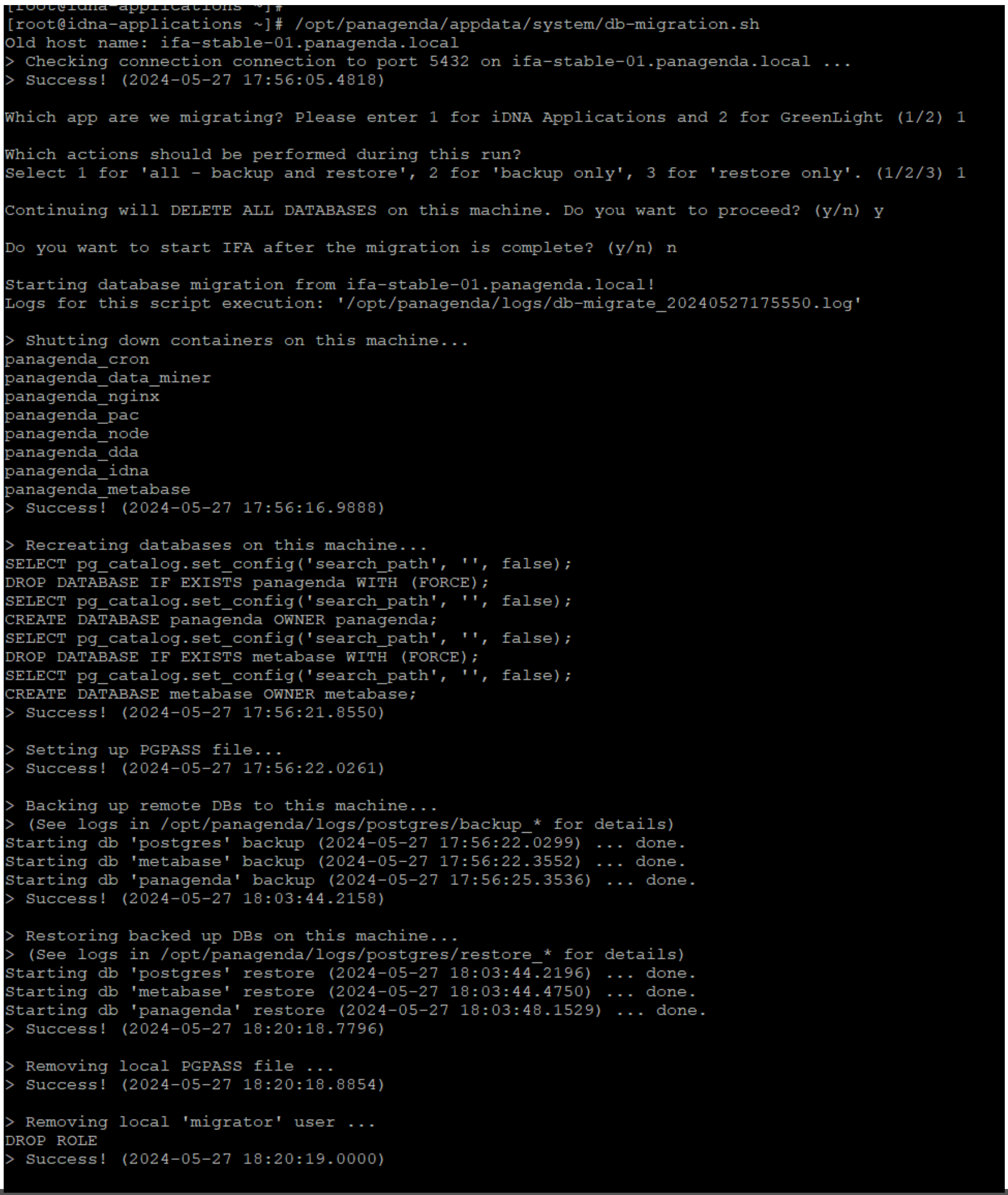
Some interactions with the script are required:

- Host name of the old appliance
- IFA or GL migration (1 should be used to perform an IFA migration)
- Backup+Restore or only one of the two (1 should be used per default)
- Acknowledgment that all DBs on the new server will be deleted

Depending on the data on the old appliance, this process may take several hours.

The combined Backup & Restore option allows you to basically fire and forget it with no interactions required between the most long-running processes.

While this process is going on, it's safe to continue with the Config Migration.



```
[root@idna-applications ~]# /opt/panagenda/appdata/system/db-migration.sh
Old host name: ifa-stable-01.panagenda.local
> Checking connection connection to port 5432 on ifa-stable-01.panagenda.local ...
> Success! (2024-05-27 17:56:05.4818)

Which app are we migrating? Please enter 1 for iDNA Applications and 2 for GreenLight (1/2) 1

Which actions should be performed during this run?
Select 1 for 'all - backup and restore', 2 for 'backup only', 3 for 'restore only'. (1/2/3) 1

Continuing will DELETE ALL DATABASES on this machine. Do you want to proceed? (y/n) y

Do you want to start IFA after the migration is complete? (y/n) n

Starting database migration from ifa-stable-01.panagenda.local!
Logs for this script execution: '/opt/panagenda/logs/db-migrate_20240527175550.log'

> Shutting down containers on this machine...
panagenda_cron
panagenda_data_miner
panagenda_nginx
panagenda_pac
panagenda_node
panagenda_dda
panagenda_idna
panagenda_metabase
> Success! (2024-05-27 17:56:16.9888)

> Recreating databases on this machine...
SELECT pg_catalog.set_config('search_path', '', false);
DROP DATABASE IF EXISTS panagenda WITH (FORCE);
SELECT pg_catalog.set_config('search_path', '', false);
CREATE DATABASE panagenda OWNER panagenda;
SELECT pg_catalog.set_config('search_path', '', false);
DROP DATABASE IF EXISTS metabase WITH (FORCE);
SELECT pg_catalog.set_config('search_path', '', false);
CREATE DATABASE metabase OWNER metabase;
> Success! (2024-05-27 17:56:21.8550)

> Setting up PGPASS file...
> Success! (2024-05-27 17:56:22.0261)

> Backing up remote DBs to this machine...
> (See logs in /opt/panagenda/logs/postgres/backup_* for details)
Starting db 'postgres' backup (2024-05-27 17:56:22.0299) ... done.
Starting db 'metabase' backup (2024-05-27 17:56:22.3552) ... done.
Starting db 'panagenda' backup (2024-05-27 17:56:25.3536) ... done.
> Success! (2024-05-27 18:03:44.2158)

> Restoring backed up DBs on this machine...
> (See logs in /opt/panagenda/logs/postgres/restore_* for details)
Starting db 'postgres' restore (2024-05-27 18:03:44.2196) ... done.
Starting db 'metabase' restore (2024-05-27 18:03:44.4750) ... done.
Starting db 'panagenda' restore (2024-05-27 18:03:48.1529) ... done.
> Success! (2024-05-27 18:20:18.7796)

> Removing local PGPASS file ...
> Success! (2024-05-27 18:20:18.8854)

> Removing local 'migrator' user ...
DROP ROLE
> Success! (2024-05-27 18:20:19.0000)
```

Config Migration

On Old Appliance: Create Appdata Backup

```
# got to volumes folder
cd /opt/panagenda/appdata/volumes/
```

Note: check the “cacerts” file in the idna folder. If it’s a symlink to something else on the old appliance, it makes sense to copy it over into a normal file in the “idna” folder and resolve it on the target appliance. It will be moved along with the other files in the next command.

```
# pack an archive with everything except dxl files
tar -czvf /root/appdata.tar.gz idna nginx ./notesdata/names.nsf
```

On New Appliance: Restore Appdata Backup

The following commands are the steps to restore the backup files in the new appliance.

```
# go to volumes folder
cd /opt/panagenda/appdata/volumes/
```

```
# rename default names.nsf
mv ./notesdata/names.nsf ./notesdata/names.orig_nsf
```

```
# copy the file over (replace OLDHOST with hostname/ip of the old appliance)
scp OLDHOST:/root/appdata.tar.gz /root/
```

```
# extract files (will be placed in correct folders automatically)
tar -xvzf /root/appdata.tar.gz
```

```
# copy dxl files over (replace OLDHOST with hostname/ip of the old appliance)
scp OLDHOST:/opt/panagenda/appdata/volumes/dxl/* ./dxl/
```

```
# clean-up (optional)
rm -f /root/appdata.tar.gz
```

Start IFA (On New Appliance)

```
# restart all IFA containers
ifa start
```