Peanut User Guide v1.0.0

Milestones	Comment
V1.0.0 Official Release	DMT @ 13 Jan 2023

Install

```
> pip3 install -i https://[USERNAME]:[PASSWORD]]@artifactory-cn.nevint.com/artifactory/api/pypi/nt3-pypi-virtual/simple peanut-ntxpkg
```

How to find USERNAME and PASSWORD? Log-in https://artifactory-cn.nevint.com/ui/login/ and visit https://artifactory-cn.nevint.com/ui/repos/tree/General/nt3-pypi-virtual, then click "Set Me Up" button and follow the instructions on the pop-up window.

Or checkout the latest source code from https://git.nevint.com/nt3-pet/peanut.

Usage

1. Introduce demo scope: **VDF** + **SOAFramework** + **Zone** + **srv1, srv2, srv3, ...**
2. Create and publish ntxpkg
   - 2.1 NVOS-VDF, SOAFramework and Zone
     - i. Support local and target, and extended properties.
     - ii. nvos-vdf==5.10.41
     - iii. zc-front-rear==1.0.2
     - iv. soa-framework==1.7.3
     - v. middleware==2.3.1 (for test purpose)
   - 2.2 Create 1-2 services ntxpkgs (local and target), and publish
     - i. Create package: tests/fixture/service, depend on middleware==2.3.1 (already published)

```
> cd tests/fixture/service
> peanut create    # Identify OS and Arch automatically.
> ls
```

```
4 service-1.0.1.ubuntu.x86_64.ntxpkg
5 > peanut publish service-1.0.1.ubuntu.x86_64.ntxpkg
```

ii. Create with non-existing dependence and failed

```
1 > cat ntxpkg.yml
2 name: service
3 type: component
4 version: 1.0.1
5 arch:
6   - x86_64
7 os:
8   - ubuntu
9   - microsys-auto
10
11 deployDependencies:
12     - middleware >= 2.3
13     - something-not-here == 1.0.0
14 > peanut create
15 ERROR:root:consistency check failed !!!   pkg something-not-here-
   1.0.0.ubuntu.x86_64.ntxpkg is not found in artifactory
```

iii. Create with unmatched version and failed

```
1 > cat ntxpkg
2 name: service
3 type: component
4 version: 1.0.1
5 arch:
6   - x86_64
7 os:
8   - ubuntu
9   - microsys-auto
10
11 deployDependencies:
12     - middleware == 3.3.1
13 > peanut create
14 ERROR:root:consistency check failed !!!   pkg middleware-
   3.3.1.ubuntu.x86_64.ntxpkg is not found in artifactory
```

iv. Create package with specified OS and Arch

```
> peanut create --os microsys-auto
> peanut create --arch aarch64
```

It is the package creator's responsibility to guarantee the correct OS and Arch are chosen.

2.3 VDF - A composite including dependence

Create VDF with unmatched dependence,

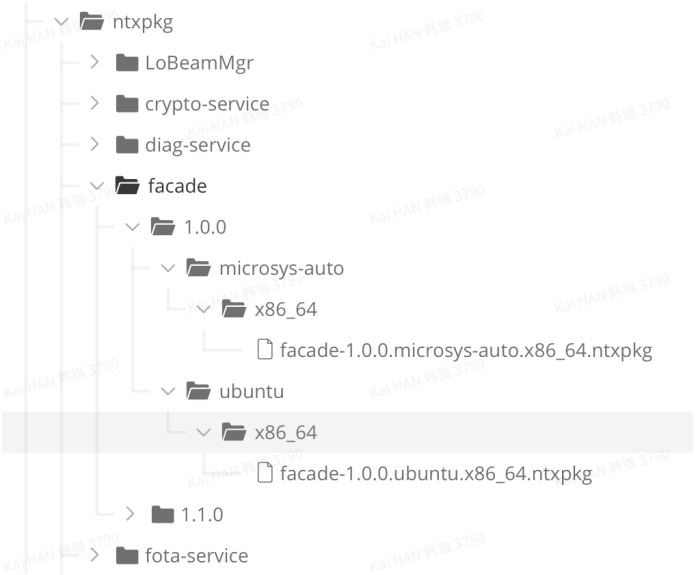
```
> cd vdf-pkg
> cat ntxpkg.yml
name: vdf
type: composite
version: 1.1.0
arch:
  - x86_64
  - aarch64
os:
  - ubuntu
  - microsys-auto

deployDependencies:
    - crypto-service == 3.2.1
    - facade == 1.1.0
    - diag-service == 1.2.6
    - fota-service == 1.3.1
    - soa-data-collection == 1.5.2
    - soa-remote-control == 1.1.9
    - soa-uds == 1.3.1
    - sota-config == 1.5.4
    - soa-remote-control-config == 2.4.5
    - sota-service == 1.3.1
    - tsync-service == 1.5.1
    - soa-deployment-config == 1.2.0
    - zc-front-rear == 1.0.2
> peanut create .
ERROR:root:consistency check failed !!!   pkg crypto-service-3.2.1.ubuntu.x86_64.ntxpkg is not found in artifactory
```

2.4 Explore what Artifactory looks like - layout, structure, properties

Layout, structure and extended properties

https://artifactory-cn.nevint.com/ui/repos/tree/General/nt3-devops-generic-local/build-system/ntxpkg/



3. Install

	app/service/middleware	VDF(nvos +middleware+ service + app)	Nio(vdf+cdf+adf+zone)
local	direct	direct	N/A
simulator	ssh/direct	QEMU	QEMU
bench	ssh/adb	ssh/flash tools(uds/serial)	ssh/flash tools

3.1 Local Installation

3.1.1 A service

```
1 > peanut install facade==1.1.0 --target local
```

ta rtarThen run the service,

```
1 > source ~/.ntxpkg/env/default/envsetup.sh
2 > ~/.ntxpkg/env/default/usr/bin/facade
```

Soa framework and facade service are installed.

3.1.2 VDF

```
1 > peanut install vdf==1.1.0 --target=local
```

Soa framework and ALL services belonging to VDF are installed (according to VDF's ntxpkg file). In this case, ZONE is also installed and runnable.

```
> ~/.ntxpkg/env/default/zone/launch.sh
```

3.2 Simulator Installation

3.2.1 Install A service on a NEW simulator

```
peanut install facade==1.1.0 --target qemu
```

The command executes following:

- Download facade, soa framework and nvos packages
- Start QEMU (qemu-system-x86_64) (2-3 minutes)
- Load kernel, rootfs, dtb
- Install facade on QEMU via ssh

Run service in QEMU,

```
> ssh -p 2222 root@127.0.0.1
-- SYSTEM INFORMATION BEGIN
-- IMAGE NAME:                nio-image-auto-vdc1
-- MACHINE:                   qemux86-64
-- BUILD TIME:                Mon Oct 10 01:16:48 UTC 2022
-- TARGET SYSTEM:             x86_64-ms-linux
-- GCC VERSION:               x86_64-ms-linux-gcc (GCC) 10.2.0
-- DISTRO NAME:               MicroSys Auto Linux BSP
-- DISTRO VERSION:            4.4.4
-- VERSION:                   NONE
-- REVISION:                  NONE
-- SYSTEM INFORMATION END
root> ls /
bin  boot  data  dev  envsetup.sh  etc        home  lib  lost+found  media
mnt  proc  run  sbin  sys  tmp  tools  usr  var
root> /usr/bin/facade
[830 830 2022-10-21 4:25:19.409508] [NrpcDeployment] # SOA Framework
Version: vmpu-soa.00.05.00
[830 830 2022-10-21 4:25:19.433205] [NrpcDeployment] # DEPLOYMENT_ROOT_PATH
is not set
```

```
[830 830 2022-10-21 4:25:19.441978] [NrpcGeneralServer] # add a sever of NrpcChannelType::kTOX
[TOX] [Oct 21 04:25:19.0472] --  tox_sock_udp_create: 666 heartbeat_interval[2000], heartbeat_timeout[10000].
root> /usr/bin/soa-uds  # currently not exist. Rerun it when finish section 3.2.3
root> exit
```

3.2.2 Install service with un-matched dependences and failed

Installing non-existing package leads to failure.

```
peanut install facade==1.3.0 --target qemu
requests.exceptions.HTTPError: 404 Client Error: Not Found for url: https://artifactory-cn.nevint.com/artifactory/nt3-devops-generic-local/build-system/ntxpkg/facade/1.3.0/microsys-auto/x86_64/facade-1.3.0.microsys-auto.x86_64.ntxpkg
```

3.2.3 Install VDF on existing simulator

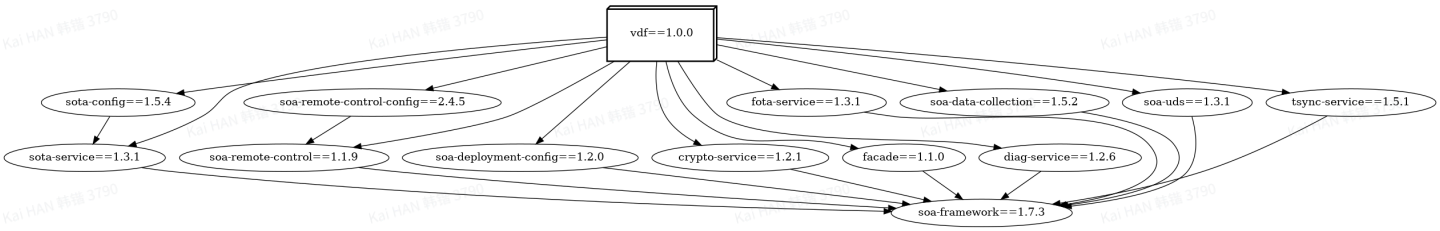
```
peanut install vdf==1.0.0 --target=qemu
```

The command executes following:

- Download vdf package
- Download ALL services packages
- Start QEMU (qemu-system-x86_64) (Skipped due to QEMU is running)
- Load kernel, rootfs, dtb (Skipped due to QEMU is running)
- Install all services on QEMU via ssh

4. Generate Dependence Digram - 2 mins

```
peanut dep-graph vdf==1.0.0
dot -Tpng -o vdf.1.0.0.png dependence_graph.dot
```


vdf==1.0.0
sota-config==1.5.4
soa-remote-control-config==2.4.5
fota-service==1.3.1
soa-data-collection==1.5.2
soa-uds==1.3.1
tsync-service==1.5.1
sota-service==1.3.1
soa-remote-control==1.1.9
soa-deployment-config==1.2.0
crypto-service==1.2.1
facade==1.1.0
diag-service==1.2.6
soa-framework==1.7.3