The New QPKG

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author(s) | Comments |
| 0.1 | 13 May 2014 | Doro | Initial |
|  |  |  |  |

1. Problems to Solve 1

2. QPKG 2

Package Source Layout 2

Open With 3

3. Build QPKG on Ubuntu by qdk-ng 4

Commands 4

4. Package Management 4

Design Detail 5

5. QNAP Inter-Application Communication (QIAC) 6

Tool 6

Object: Core 7

Object: QnapNetwork 7

Object: QnapPower 8

Object: QnapDevice 8

6. Application Backup 8

# Problems to Solve

* Pack QPKG easily
  + Tool chain
* Package Management
  + Dependency
  + Backup or Migration between NAS, consider
    - Different NAS Hardware and version
    - File permissions and owner
* Inter-Application Communication

# QPKG

## Package Source Layout

Package source layout complies with Debian Package Rule, but replacing debian folder to *QNAP* folder. For example, the layout of application foobar looks like

foobar

├── etc

│   └── config

│   └── foobar.conf

├── QNAP

│   ├── changelog

│   ├── control

│   ├── rules

│   ├── foobar.config

│   ├── foobar.dirs

│   ├── foobar.init

│   ├── foobar.install

│   ├── foobar.links

│   ├── foobar.mime

│   ├── foobar.postinst

│   ├── foobar.postrm

│   ├── foobar.preinst

│   └── foobar.prerm

└── usr

├── bin

│   └── foobar.sh

├── lib

│   └── foobar.so

└── share

└── foobar

└── foobar.txt

The file purpose in QNAP describes in Chapter 4 & 5 of Debian New Maintainers' Guide, <https://www.debian.org/doc/manuals/maint-guide/>.

The etc and usr folders will be automatically copied to Application root folder in NAS, such as /share/CACHEDEV1\_DATA/.qpkg/foobar/. But etc/config/ is an exception that will be copied to /etc/config/ in NAS.

QNAP/control is the most important file describing the package. The fields are defined in Debian Policy Manual, <https://www.debian.org/doc/debian-policy>. The field, which prefix is “Q-”, is used and defined by QNAP. Following is an example of *control* file,

Source: foobar

Section: python

Vcs-Bzr: lp:~oem-solutions-group/dell/template-abba

Homepage: http://123.com.tw/

Priority: extra

Build-Depends: cdbs (>= 0.4.43),

debhelper (>= 6),

python,

python-support (>= 0.6.4),

python-distutils-extra (>= 2.10)

Maintainer: Doro Wu <dorowu@qnap.com>

Package: foobar

Architecture: all

Q-AppName: Foobar Application

Q-Firmware: qnap (>= 4.1), qnap (<< 4.2)

Recommends: python-xlsxwriter

Depends: python (<< 3.0), python (>= 2.6),

python-launchpadlib,

python-distutils-extra,

python-jinja2,

sshfs

Conflicts: lp-fish-init (<< 0.4)

Description: Utility scripts for OEM Dell enablement

UNKNOWN

## Open With

Task: Migrate mime-support on Ubuntu to NAS

Open files in NAS with NAS applications registered by LXE/Docker Containers. The rules, which file suffix opens with which application, comply with MIME format of Ubuntu desktop. /etc/config/mime.types shows the relation between MIME type and file suffix, for example,

text/html html htm shtml

text/plain txt srt

The application, for example foobar, specifies the action for dedicated MIME type. foobar has to put the command in the file /etc/config/mime/packages/foobar.xml, the file looks like,

<?xml version="1.0" encoding="UTF-8"?>

<mime-info xmlns="http://www.freedesktop.org/standards/shared-mime-info">

<mime-type type="application/vnd.sun.xml.draw">

<comment>OpenOffice.org 1.0 Drawing</comment>

<comment xml:lang="af">OpenOffice.org 1.0 Drawing</comment>

<comment xml:lang="zh-TW">OpenOffice.org 1.0 繪圖</comment>

<glob pattern="\*.sxd"/>

</mime-type>

</mime-info>

Alternatively foobar could write this rule in simple format if packing with qdk-ng. For example, create a file located QNAP/foobar.mime with the format <MIME Type>; <Command>, such as

application/x-dell-driver; dell-driver-installer '%s'

# Build QPKG on Ubuntu by qdk-ng

Task: Package qpkg-ng on Ubuntu 12.04 and 14.04

qdk-ng is an command-line tool to pack QPKG easily. It can be installed on Ubuntu 12.04 and 14.04 as following

sudo add-apt-repository ppa:fcwu-tw/ppa

sudo apt-get update

sudo apt-get install qpkg-ng

## Commands

qbuild-ng create <package\_name> [package\_folder]

qbuild-ng build [package\_folder]

“qbuild-ng create” will create dummy source package with name package\_name. If package\_folder is given, it creates QNAP folder in package\_folder. Else, package\_folder will be created.

“qbuild-ng build” will build the package to \*.qpkg.

# Package Management

Task: NAS tool qpt

Task: qscan-ng in qpkg-ng

QNAP Packing Tool (QPT), similar to apt on Ubuntu, is a command-line tool for handling package on QNAP NAS. QPT provides following command to manage package:

qpt-install <package\_name>

qpt-remove <package\_name>

qpt-search <package\_name>

qpt-update

qpt-list [keyword]

The repositories were listed in /etc/config/qdt/sources.list. For example,

qpkg <http://update.qnap.com/qts/> 4.2 main extra

With above example, the program would try to fetch package during executing qpt-update using the URL as following,

* [http://update.qnap.com/qts/dists/4.2/main/binary-${arch}/Packages.{bz2](http://update.qnap.com/qts/dists/4.2/main/binary-$%7barch%7d/Packages.%7bbz2), gz,}
* [http://update.qnap.com/qts/dists/4.2/extra/binary-${arch}/Packages.{bz2](http://update.qnap.com/qts/dists/4.2/extra/binary-$%7barch%7d/Packages.%7bbz2), gz,}

Where the packages looks like:

Package: account-plugin-aim

Priority: optional

Section: gnome

Installed-Size: 941

Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>

Architecture: amd64

Source: empathy

Version: 3.8.6-0ubuntu9

Depends: empathy (= 3.8.6-0ubuntu9), telepathy-haze, mcp-account-manager-uoa, unity-asset-pool (>> 0.8.24daily13.03.20.1)

Breaks: account-plugin-empathy

Filename: pool/main/e/empathy/account-plugin-aim\_3.8.6-0ubuntu9\_amd64.deb

Size: 8838

MD5sum: f7f709442600bb60bc5e230f905773ae

SHA1: 588d6aa693e4b9a9bda228360bc65dfab4527ed8

SHA256: 4a34e416bb37191d0b8e6855b27cdbf7cd63fec182ab415cc9e71b19cfe55e48

Description: Messaging account plugin for AIM

Homepage: http://wiki.gnome.org/Empathy

Description-md5: 1a2069e5dd5f4777061642b2d7c9a76a

Bugs: https://bugs.launchpad.net/ubuntu/+filebug

Origin: Ubuntu

Supported: 5y

Task: ubuntu-desktop, ubuntu-usb, edubuntu-desktop, edubuntu-usb

We need a new tool that works like qpkg-scanpackages to generate the Packages file. For more reference, refer to

* <https://www.debian.org/doc/manuals/repository-howto/repository-howto>
* <https://help.ubuntu.com/community/Repositories/Personal>

## Design Detail

Control files, where is QNAP/ of source package

/share/CACHEDEV1\_DATA/.qpkg/qpt/info/

/share/CACHEDEV1\_DATA/.qpkg/qpt/status

/share/CACHEDEV1\_DATA/.qpkg/qpt/lists/

# QNAP Inter-Application Communication (QIAC)

Task: qiac, qiac-core, qiac-qnapnetwork in NAS

QIAC is based on ZeroMQ (<http://zeromq.org/>), comparing to others Advanced Message Queuing Protocol implementation, such as RabbitMQ and ActiveMQ, which is lightweight and fastest[[1]](#footnote-1).

The application that compatible with QIAC would publish its API with file socket by ZeroMQ. For example, the application foobar opens its socket at

/var/run/qiac/public/foobar/socket

, and optionally register itself by */methods/Register* of */Core* object. Any privilege APIs would open at

/var/run/qiac/privilege/foobar/socket

The API provider should confirm UID is 0 and permission is 700 of this socket file. When providers register its interface by */Core* object, */Core* would validate the permission and path.

## Tool

QIAC provides a command-line tool, qiac, to easily access this interface as shown as following,

qiac / # list all objects

qiac /QnapNetwork/ # list interfaces of QnapNetwork

qiac /QnapNetwork/methods/GetHostname # call method GetHostname

qiac /QnapNetwork/methods/SetHostname ‘{“hostname”: “myhost”}’

qiac /QnapNetwork/signals/statusChanged # wait statusChanged signal

qiac -t 10 /QnapNetwork/signals/hostnameChanged # wait signal in 10 seconds

qiac /QnapNetwork/signals/ # wait all signals

## Object: Core

|  |  |  |  |
| --- | --- | --- | --- |
| Object | Interface | Arguments | Privilege |
| /Core | /methods/Register | {“object”: “str”} |  |
| /Core | /Help |  |  |
| /Core | /signals/objectsChanged |  |  |

qiac /Core

{

"/Core": {

"methods": [

{

"Register": {

"object": "str"

}

}

],

"signals": [

"objectsChanges"

]

}

}

### /Core/methods/Register

Arguments:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| object | str |  |

Return value:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| ret | int | * 0: OK * < 0: Error |

Example:

qiac /Core/Register ‘{“object”: “/QnapPower”}’

{“ret”: 0}

### /Core/signals/objectsChanged

Arguments:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
|  |  |  |

Return value:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| ret | int | * 0: OK * < 0: Error |
| object | str |  |
| event | str | “add” or “remove” |

Example:

qiac /Core/signals/objectsChange

{“ret”: 0, “object”: “/QnapPower”, “event”: “add”}

## Object: QnapNetwork

|  |  |  |
| --- | --- | --- |
| Object | Interface | Privilege |
| /QnapNetwork | /methods/GetHostname |  |
| /QnapNetwork | /methods/SetHostname | ● |
| /QnapNetwork | /signals/hostnameChanged |  |
| /QnapNetwork | /signals/statusChanged |  |

qiac /QnapNetwork/Help

{

"QnapNetwork": {

"methods": [

{

"GetHostname": {}

},

{

"SetHostname": {

"hostname": "str"

},

"privilege": true

}

],

"signals": [

"hostnameChanged",

"statusChanged"

]

}

}

### /QnapNetwork/methods/GetHostname

Arguments:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |

Return value:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| ret | int | * 0: OK * < 0: Error |
| hostname | str |  |

Example:

qiac /QnapNetwork/methods/GetHostname

{“ret”: 0, “hostname”: “MyNAS”}

### /QnapNetwork/methods/SetHostname

Arguments:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| hostname | str |  |

Return value:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| ret | int | * 0: OK * < 0: Error |

Example:

qiac /QnapNetwork/methods/SetHostname ‘{“hostname”: “MyNAS”}’

{“ret”: 0}

### /QnapNetwork/signals/hostnameChanged

Arguments:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |

Return value:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| ret | int | * 0: OK * < 0: Error |
| hostname | str |  |

Example:

qiac /QnapNetwork/signals/hostnameChanged

{“ret”: 0, “hostname”: “MyNAS”}

### /QnapNetwork/signals/statusChanged (TODO)

Address/Connection changed

Arguments:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |

Return value:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| ret | int | * 0: OK * < 0: Error |
| hostname | str |  |

Example:

qiac /QnapNetwork/signals/hostnameChanged

{“ret”: 0, “hostname”: “MyNAS”}

## Object: QnapPower

|  |  |  |  |
| --- | --- | --- | --- |
| Object | Interface | Arguments | Privilege |
| /QnapPower | /methods/RegisterPoweroffCallback |  |  |
| /QnapPower | /methods/RegisterSuspendCallback |  |  |
| /QnapPower | /methods/UnregisterPoweroffCallback |  |  |
| /QnapPower | /methods/UnregisterSuspendCallback |  |  |
| /QnapPower | /methods/ListPoweroffCallback |  |  |
| /QnapPower | /methods/ListSuspendCallback |  |  |
| /QnapPower | /signals/poweroff |  |  |
| /QnapNetwork | /signals/suspend |  |  |

### /QnapPower/methods/RegisterPoweroffCallback

Arguments:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| type | str | * “program” * “qiac” |
| path | str | **program**  Output progress to stdout. The program would be killed if no output in 30 seconds  Return   * 0: continues to poweroff procedure * < 0: revoke   **qiac**  With PULL/PUSH way, Callback server returns code and status as following format:  {“ret”: 1, “status”: “Do cleanup”}  Where ret is   * 0: continues to poweroff procedure * 1: status update * < 0: revoke |

Return value:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| ret | int | * 0: OK * < 0: Error |

Example:

qiac /QnapPower/methods/RegisterPoweroffCallback ‘{“type”: “program”, “path”: “/share/CACHEDEV1\_DATA/.qpkg/foobar/usr/bin/poweroff\_check”}’

cleanup

cleanup

echo $?

0

### /QnapPower/methods/RegisterSuspendCallback

As same as /QnapPower/methods/RegisterPoweroffCallback

## Object: QnapDevice

|  |  |  |  |
| --- | --- | --- | --- |
| Object | Interface | Arguments | Privilege |
| /QnapDevice | /signals/event |  |  |

### /QnaDevice/signals/event

Arguments:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |

Return value:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| ret | int | * 0: OK * < 0: Error |
| object | str |  |
| event | str | “add” or “remove” |

Example:

qiac /QnapDevice/signals/event

{“ret”: 0, “object”: “/block/sdf/sdf1”, “event”: “add”}

# Application Backup

Consider

* Different NAS Hardware and version
* File permissions and owner

Methods

* Application specific
  + (NAS/control) Q-BackupProgram: Program Path
  + (NAS/app.backup)
* Generic
  + Overlayfs + MTRR
  + (NAS/app.md5sum)
  + (NAS/app.backupWatchDirs)

NAS hardware and version should not affect backup result

File permission is simply replicated by -a of cp and rsync command

Via an owner map, do owner changing for each backup

1. <http://blog.x-aeon.com/2013/04/10/a-quick-message-queue-benchmark-activemq-rabbitmq-hornetq-qpid-apollo/> [↑](#footnote-ref-1)