AUR-Pakete

In diesem Kapitel geht es um das erstellen oder verändern von AUR-Paketen.

AUR-Paket anpassen inklusive Erstellung Patch

Da sich z.B. Bibliotheken geändert haben und dadurch die Abhängigkeiten nicht mehr auflösen lassen, muss ein existierendes AUR-Paket angepasst werden.

In diesem Beispiel geht es konkret um das Pakte urbackup2-client, welches wegen der Umstellung der wxgtk-Pakete (https://archlinux.org/news/wxwidgets-32-update-may-need-manual-intervention/) angepasst werden muss.

Dieses AUR-Paket wird über GitHub (https://github.com/funilrys/aur-urbackup2-client) verwaltet. Die Beschreibung der Änderungen am Paket sind aber auch allgemein gültig.

Zur GitHub-Anbindung nutze ich Visual Studio Code.

Git-Repository clonen

Wie das mit VS-Code geht ist hier beschrieben.

So sieht es dann aus:

hier folgt noch ein Bild...

.SRCINFO-Datei erstellen

```
$ makepkg --printsrcinfo > .SRCINFO
```

Patch erstellen

Zunächst "src"-Verzeichnis löschen, sofern es schon da ist.

Dann mit

```
makepkg --nobuild
```

das Source-File downloaden. Außerdem wird der Sourcecode entpackt.

Dann erstellt man sich von dem Verzeichnis, in dem eine Datei gepatcht werden muss zwei Kopien an einem anderen Ort.

Z.B. kopiert man den Ordner "src" nach "src_ori" und "src_neu".

Dann führt man die notwendigen Änderungen im Ordner "src_neu" durch. Im übergeordnetem Ordner führt man dann zunächst

```
diff --unified --recursive --text src_ori src_neu --color
```

aus. Das ist ein Testlauf der die gefunden Änderungen farblich im Terminal dargstellt.

Ist man zufrieden muss noch

```
diff --unified --recursive --text src_ori src_neu > neuerpatch.patch
```

ausführen, was die eigentliche Patchdatei mit dem Namen "neuerpatch.pach" erstellt.

Anschließend kopiert man die Datei "neuerpatch.pach" in das Verzeichnis "src_ori" und fürht dort folgendes aus:

```
patch --strip=1 < ../neuerpatch.patch
```

Die Änderungen sollten nun auch im Verzeichnis "src_ori" durchgeführt sein. Dieses sollte man überprüfen.

Wenn der Patch funktioniert wird dieser in das eigentliche Paket-Verzeichnis kopiert. Dort muss dann die PKGBUILD-Datei entsprechend angepasst. Dazu diese editieren. Hier ein Beispiel:

```
pkgname=urbackup2-client
pkgver=2.4.11
pkgrel=4
pkgdesc="Client Server backup system"
arch=('i686' 'x86_64' 'armv5' 'armv6h' 'armv6' 'armv7h' 'armv7' 'aarch64')
url="http://www.urbackup.org/"
license=("GPL")
makedepends=('gcc-libs' 'gcc' 'make')
depends=('wxwidgets' 'crypto++' 'zlib')
conflicts=('urbackup2-client-no-gui' 'urbackup-client-no-gui' 'urbackup-client')
```

```
source=(
    "https://www.urbackup.org/downloads/Client/${pkgver}/urbackup-client-${pkgver}.0.tar.gz"
    'btrfs_create_filesystem_snapshot'
    'btrfs_remove_filesystem_snapshot'
    'dattobd_create_filesystem_snapshot'
    'dattobd_remove_filesystem_snapshot'
    'defaults_client'
    'lvm_create_filesystem_snapshot'
    'lvm_remove_filesystem_snapshot'
    'md5-bytes.patch'
    'double_language.patch'
)
sha512sums=('a0cae5dad805d11e2764b2635dd37a4e24e0d029bae1eefc471f2b87262ac2448b3d123c2ffb5deb2eccab1baed0e31d3711e8837d66b93edf736fb113145ef1'

'416fb8f5f3687a3c369cc2b199d4c8b4170494f0a119566a91ac6a0c2f202dc5049804c10508b66ba657011b39be5ddd055091cd531a665b4398899f404086ca'

'860021ce5b8d92ff58e8286991162c7bab45493c3b9c87577a43764f6b416397448bb99b8fcb850c4c5853927cb0a8637792b75ff53ee7ee257da3f5d29ae3a7'

'fde5912b589a495dc03a26d174d7673ff746eed34d6b1ed64758b2dc2ec2ec53e02e6a28b04734a7112f16687b31d25123e99dbc69e9dcab48773675382ec582'

'a8b58bba1b8b0a6b70395f9fe4277eeee60a0ba534f4eddb999d719915c76b76facb54172e03b7b29b9f725a4d720e9b676b05e5081f7528570956e903fe59bd'

'238c286d451474a8721292f7e98b4f13600cb430c16a27ceb9551cc83705b8268a3f1202785fb5b61523f372b4e7e804fd20b7db62677621983d79a271aa106b'

'a2d4ba03ae15582d2cd74ff68c38ff0f90d75a6eb5c241f9a022b0652fa2dc9b184439f6bda9a9538645925f739503ee7b3fc7bb232589583cdeb6dc27d74e5c'

'9bdfefccdd9d6e37a77975324a7c417f3de2aa59e6da0bfde3c318b8c6f3d7f4629f3a41eebee548b9c572b8ed39640434cc08bd020d25362fddffc4426438de'

'34e25c868cf4572414fbc6c693877127152f9a97edf8865b4263a55cf16f71a5045ba96b1a9af8244ed49c35cab56e3fdb44348d191e9f85e2efb66392907132'

'fb6c43d725d4ea201bdf91b1482473f205834c14f24906041d5c6ef22b1e0f4cec16d5e765a3fda6fed4d3b98f621
```

```
7b190dd5d1e84051525205e22747c5eefac')

CFLAGS="-march=native -O2 -pipe -fstack-protector-strong"
CXXFLAGS="${CFLAGS} -ansi -std=gnu++11"
CPPFLAGS="${CPPFLAGS} -DNDEBUG"
MAKEFLAGS="-j$(nproc)"

build() {
    sed  -i '/\#include \"cryptopp_inc.h\"/a #include "assert.h"' "${srcdir}/urbackup-client-
${pkgver}.0/cryptoplugin/AESGCMDecryption.h"

    patch -d"${srcdir}/urbackup-client-${pkgver}.0" -p0 < "${srcdir}/md5-bytes.patch"
    patch --forward --strip=1 --input="${srcdir}/double_language.patch"

    cd "${srcdir}/urbackup-client-${pkgver}.0"
    ./configure --prefix=/usr --sbindir=/usr/bin --localstatedir=/var --sysconfdir=/etc --
enable-embedded-cryptopp
    make
}

package() {
    cd "${srcdir}/urbackup-client-${pkgver}.0"
    make DESTDIR="${pkgdir}" install
    sed -i 's/\/usr\/local\/sbin/\/usr\/bin/gi' urbackupclientbackend-debian.service
    install -Dm644 urbackupclientbackend-debian.service \
    "${pkgdir}"/usr/lib/systemd/system/urbackupclientbackend.service
    install -Dm644 docs/urbackupclientbackend.1 \
    "${pkgdir}"/usr/share/man/man1/urbackupclientbackend.1

    cd "${srcdir}"
    install -Dm644 defaults_client "${pkgdir}/etc/default/urbackupclient"
    install -Dm700 btrfs_create_filesystem_snapshot "${pkgdir}/usr/share/urbackup"
    install -Dm700 btrfs_remove_filesystem_snapshot "${pkgdir}/usr/share/urbackup"
    install -Dm700 lvm_create_filesystem_snapshot "${pkgdir}/usr/share/urbackup"
    install -Dm700 lvm_remove_filesystem_snapshot "${pkgdir}/usr/share/urbackup"
    install -Dm700 dattobd_create_filesystem_snapshot "${pkgdir}/usr/share/urbackup"
    install -Dm700 dattobd_remove_filesystem_snapshot "${pkgdir}/usr/share/urbackup"
}
# vim: ts=2
```

Bei diesem Paket habe ich den Patch "double_language.patch" ergänzt. Dazu musste diese in diesem Abschnitt am Ende hinzugefügt werden:

```
source=(
    "https://www.urbackup.org/downloads/Client/${pkgver}/urbackup-client-${pkgver}.0.tar.gz"
    'btrfs_create_filesystem_snapshot'
    'btrfs_remove_filesystem_snapshot'
    'dattobd_create_filesystem_snapshot'
    'dattobd_remove_filesystem_snapshot'
    'defaults_client'
    'lvm_create_filesystem_snapshot'
    'lvm_remove_filesystem_snapshot'
    'md5-bytes.patch'
    'double_language.patch'
)
```

Damit der Patch auch ausgeführt wird benötigt es noch die Zeile

```
patch --forward --strip=1 --input="${srcdir}/double_language.patch"
```

im Abschnitt

```
build() {
    sed  -i '/\#include \"cryptopp_inc.h\"/a #include "assert.h"' "${srcdir}/urbackup-client-
${pkgver}.0/cryptoplugin/AESGCMDecryption.h"

    patch -d"${srcdir}/urbackup-client-${pkgver}.0" -p0 < "${srcdir}/md5-bytes.patch"
    patch --forward --strip=1 --input="${srcdir}/double_language.patch"

    cd "${srcdir}/urbackup-client-${pkgver}.0"
    ./configure --prefix=/usr --sbindir=/usr/bin --localstatedir=/var --sysconfdir=/etc --
enable-embedded-cryptopp
    make
}
```

Offiziell sollen solche Patch-Files in einem eigenen "prepare()"-Abschnitt eingefügt werden. Da in diesem Fall aber bereits ein andere Patch im "build()"- Abschnitt existierte, habe ich ihn dort eingefügt.

Nach den Änderungen muss noch mit

```
updpkgsums
```

die Checksumme ergänzt werden.

Checksumme manuell erstellen

```
sha512sum file1.txt
```

Checksummen automatisiert erstellen (für alle Dateien)

```
updpkgsums
```

Links:

https://wiki.archlinux.org/title/Patching_packages

https://wiki.archlinux.de/title/Arch_Build_System

AUR-Paket erstellen

https://aaron-fischer.net/rxy

AUR-Pakete veröffentlichen

Möchte man für das AUR Pakete veröffentlichen , benötigt man zunächst folgende Dinge:

1. AUR-Account
2. SSH-Schlüsselpaar

Das anlegen eines Accounts ist selbsterklärend. Daher gehe ich hier nicht weiter darauf ein.

SSH-Schlüsselpaar erzeugen

Dieses wird von ChatGPT wie folgt beschrieben:

```
ssh-keygen -t ed25519 -C "deinname@deinrechner"
```

- `-t ed25519` → moderner, sicherer Schlüsseltyp
- `-C "Kommentar"` → nur ein Hinweis (meist Benutzername@Rechnername), optional

Dann kommt:

```
Enter file in which to save the key (/home/thomas/.ssh/id_ed25519):
```

- Standard ist `~/.ssh/id_ed25519`.
- Falls du schon einen Key dort hast, kannst du ENTER drücken (überschreibt nicht automatisch), oder einen neuen Namen vergeben wie `~/.ssh/id_ed25519_aur`.

Dann wirst du nach einer **Passphrase** gefragt – das ist optional, aber empfohlen. Wenn du keine willst, einfach ENTER drücken.

Danach liegen zwei Dateien in `~/.ssh/`:

- **privater Schlüssel**: `id_ed25519`
  (gut aufbewahren, darf nie hochgeladen werden)
- **öffentlicher Schlüssel**: `id_ed25519.pub`
  (den musst du ins AUR eintragen)

Inhalt der Datei anzeigen:

```
cat ~/.ssh/id_ed25519.pub
```

- Den kompletten Inhalt (eine Zeile, beginnt mit `ssh-ed25519 ...`) kopieren.

- Auf aur.archlinux.or einloggen → **My Account** → **SSH Public Keys** → einfügen → speichern.

Jetzt prüfen, ob die Verbindung klappt:

```
ssh -T aur@aur.archlinux.org
```

Wenn alles passt, kommt so etwas wie:

```
Welcome to the AUR, <AUR-Benutzername>! Interactive shell is disabled.
```

Paket anpassen und veröffentlichen

Zunächst auf der Ordnerstruktur im persönlichen Ordner ein geeignetes Verzeichnis anlagen und darein wechseln und ein Terminal öffnen.

Dann folgendes ausführen um den aktuelle Stand des Paketes in den Ordner zu laden. Hier am Beispiel des Pakets "cockpit-navigator":

```
git clone ssh://aur@aur.archlinux.org/cockpit-navigator.git
```

Es ensteht der Ordner "cockpit-navigator". Darein wechseln:

```
cd cockpit-navigator
```

Darin befindet sich u.a. die Datei PKGBUILD. Diese mit einem Editor nach Bedarf anpassen. U.a. muss sich bei jeder Änderung mindestens die Paketversion ändern.

Hat man alle Änderungen durchgeführt und die Datei gespeichert, muss man die Checksumme(n) anpassen. Dazu im Terminal folgenden Befehl ausführen:

```
updpkgsums
```

Dabei wird auch der Sourcecode runtergeladen, damit die Checksumme gebildet werden kann.

Dann sollte die Datei .SRCINFO neu erstellt werden:

```
makepkg --printsrcinfo > .SRCINFO
```

Nun sollte die Erstellung und Installation des Paketes getestet werden:

```
makepkg -si
```

Läuft alles fehlerfrei durch, müssen jetzt noch die Änderungen dokumentiert und hochgeladen werden:

```
git add PKGBUILD .SRCINFO
git commit -m "einen passenden Kommmentar hinterlegen"


git push origin master
```

Jetzt wird man noch mal nach dem Passwort zu seinem SSH-Schlüssel gefragt. Nach der Eingabe erfolgt das Hochladen der Änderung und die Veröffentlichung.