webosbrew / **webos-homebrew-channel** Public

Unofficial webOS TV homebrew store.

MIT License

71 stars **10** forks

Star Notifications

Code Issues 13 Pull requests 2 Actions Projects Wiki ···

main ▾ Go to file

Informatic 0.3.1 ··· ✓ 4 days ago **134**

View code

README.md

webos-homebrew-channel

Features

- Independent webOS package repository
  - Homebrew discovery, installation & updates
- (root) Root execution service that can be easily used by webOS homebrew developers without a need of separate privilege escalation handling (`luna://org.webosbrew.hbchannel.service/exec`)
- (root) Disable some telemetry
- (root) Startup user hooks (executable scripts present in `/var/lib/webosbrew/init.d` are run on bootup)
- (root) Remote access

  - SSH - public key authenticated (with default `alpine` password until authorized keys are provisioned)
  - Telnet - unauthenticated, use sparingly

- (root) Failsafe mode

  - In case a device crashes on boot only an emergency shell will be exposed via telnet. In order to disable it fix the original crash cause and remove `/var/luna/preferences/webosbrew_failsafe` flag file.

Installation

Automated (recommended)

Latest Homebrew Channel version is automatically installed when rooting a TV with https://rootmy.tv exploit.

Automated

Execute the following snippet on target TV using SSH or Telnet:

```
curl -L https://raw.githubusercontent.com/webosbrew/webos-homebrew-channel/ma

# Update startup script (assuming running as root)
cp /media/developer/apps/usr/palm/services/org.webosbrew.hbchannel.service/st
```

Manual

- Download latest release `.ipk`
- Install it using `ares-install` SDK command or using the following command directly on a TV:

  ```
  luna-send-pub -i 'luna://com.webos.appInstallService/dev/install' '{"id":
  ```

- (root) Elevate privileges by running:

  ```
  /media/developer/apps/usr/palm/services/org.webosbrew.hbchannel.service/e
  ```

- (root) Update startup script:

```
cp /media/developer/apps/usr/palm/services/org.webosbrew.hbchannel.servic
```

Development

Environment

Some libraries used by this project are submodules of this repository. Use following command when cloning:

```
git clone --recursive https://github.com/webosbrew/webos-homebrew-channel
```

All required development packages are distributed via npm. In order to install them in a local directory run:

```
npm install
```

Technology stack

Frontend is based on last development version of enyo. (dated january 2017) While this definitely is not the cool and jazzy latest technology, it provides us with a sensible UI toolkit for TV-based application that seems to work pretty well with versions of webOS as old as 1.x. We are currently using enyo built-in `enyo-dev` packager. This requires us to write code that'll be run directly on target platforms (no babel/transpilation of newer ES dialects - no arrow functions, no const, no promises, etc.). In the future we may migrate to some webpack-based solution around that.

Service is packaged using webpack & babel, thus, with enough shims and patches, we can write and use some modern ES features (like Promises, arrow functions, async functions, const, etc.) while targeting NodeJS 0.10 (used on earliest webOS versions)

Development TV setup

Configuring @webosose/ares-cli with Developer Mode App

This is partially based on: https://webostv.developer.lge.com/develop/app-test/using-devmode-app/

- Install Developer Mode app from Content Store
- Enable developer mode, enable keyserver

- Download TV's private key: `http://TV_IP:9991/webos_rsa`
- Configure the device using `ares-setup-device` (`-a` may need to be replaced with `-m` if device named `webos` is already configured)
    - `PASSPHRASE` is the 6-character passphrase printed on screen in developer mode app

```
ares-setup-device -a webos -i "username=prisoner" -i "privatekey=/path/to/dow
```

Configuring @webosose/ares-cli with rooted TV

- Enable sshd in Homebrew Channel app
- Generate ssh key on developer machine (`ssh-keygen`)
- Copy the public key (`id_rsa.pub`) to `/home/root/.ssh/authorized_keys` on TV
- Configure the device using `ares-setup-device` (`-a` may need to be replaced with `-m` if device named `webos` is already configured)

```
ares-setup-device -a webos -i "username=root" -i "privatekey=/path/to/id_rsa"
```

Note: @webosose/ares-cli doesn't need to be installed globally - you can use a package installed locally after `npm install` in this repo by just prefixing above commands with local path, like so: `node_modules/.bin/ares-setup-device ...`

Production build

```
rm -rf dist && npm run build -- --production && npm run build-service -- --en
```

Full application testing / deployment

```
rm -rf dist && npm run build && npm run build-service && npm run package && n
ssh root@10.0.0.2 -p 9922 /media/developer/apps/usr/palm/services/org.webosbr
```

Service testing

```
npm run build-service && \
    cat dist/services/service.js | ssh root@10.0.0.2 -p 9922 sh -c 'cat > /me
ssh root@10.0.0.2 -p 9922 /usr/bin/run-js-service -k -n /media/developer/apps
```

Update start-devmode.sh script

```
cp /media/developer/apps/usr/palm/services/org.webosbrew.hbchannel.service/st
```

Release engineering

```
npm version minor
git push origin main --tags
```

Releases 6

Release v0.2.1 Latest
on May 16

+ 5 releases

Contributors 7

Languages

● **JavaScript** 63.7% ● **TypeScript** 18.0% ● **Shell** 15.7% ● **Python** 1.3% ● **C** 0.9% ● **CSS** 0.3%
● **Makefile** 0.1%