The BlackArch Linux Guide

https://www.blackarch.org/

Contents

1 Introduction 3
1.1 Overview 3
1.2 What is BlackArch Linux? 3
1.3 History of BlackArch Linux 3
1.4 Supported platforms 3
1.5 Get involved 4

2 User Guide 5
2.1 Installation 5
2.1.1 Installing on top of ArchLinux 5
2.1.2 Installing packages 5
2.1.3 Installing packages from source 6
2.1.4 Basic Blackman usage 6
2.1.5 Installing from live-, netinstall- ISO or ArchLinux 7

3 Developer Guide 8
3.1 Arch's Build System and Repositories 8
3.2 Blackarch PKGBUILD standards 8
3.2.1 Groups 8
3.2.1.1 blackarch 8
3.2.1.2 blackarch-anti-forensic 9
3.2.1.3 blackarch-automation 9
3.2.1.4 blackarch-backdoor 9
3.2.1.5 blackarch-binary 9
3.2.1.6 blackarch-bluetooth 9
3.2.1.7 blackarch-code-audit 9
3.2.1.8 blackarch-cracker 9
3.2.1.9 blackarch-crypto 9
3.2.1.10 blackarch-database 10
3.2.1.11 blackarch-debugger 10
3.2.1.12 blackarch-decompiler 10
3.2.1.13 blackarch-defensive 10
3.2.1.14 blackarch-disassembler 10
3.2.1.15 blackarch-dos 10
3.2.1.16 blackarch-drone 10
3.2.1.17 blackarch-exploitation 10
3.2.1.18 blackarch-fingerprint 11
3.2.1.19 blackarch-firmware 11
3.2.1.20 blackarch-forensic 11
3.2.1.21 blackarch-fuzzer 11

3.2.1.22 blackarch-hardware 11
3.2.1.23 blackarch-honeypot 11
3.2.1.24 blackarch-keylogger 11
3.2.1.25 blackarch-malware 11
3.2.1.26 blackarch-misc 12
3.2.1.27 blackarch-mobile 12
3.2.1.28 blackarch-networking 12
3.2.1.29 blackarch-nfc 12
3.2.1.30 blackarch-packer 12
3.2.1.31 blackarch-proxy 12
3.2.1.32 blackarch-recon 12
3.2.1.33 blackarch-reversing 12
3.2.1.34 blackarch-scanner 13
3.2.1.35 blackarch-sniffer 13
3.2.1.36 blackarch-social 13
3.2.1.37 blackarch-spoof 13
3.2.1.38 blackarch-threat-model 13
3.2.1.39 blackarch-tunnel 13
3.2.1.40 blackarch-unpacker 13
3.2.1.41 blackarch-voip 13
3.2.1.42 blackarch-webapp 14
3.2.1.43 blackarch-windows 14
3.2.1.44 blackarch-wireless 14
3.3 Repository structure 14
3.3.1 Scripts 14
3.4 Contributing to repository 15
3.4.1 Required tutorials 16
3.4.2 Steps for contributing 16
3.4.3 Example 16
3.4.3.1 Fetch PKGBUILD 16
3.4.3.2 Clean up PKGBUILD 16
3.4.3.3 Adjust PKGBUILD 17
3.4.3.4 Build the package 17
3.4.3.5 Install and test the package 17
3.4.3.6 Add, commit and push package 17
3.4.3.7 Create a pull request 18
3.4.3.8 Adding a remote for upstream 18
3.4.4 Requests 18
3.4.5 General tips 18

4 Tools Guide **19**
4.1 Coming Soon 19

Chapter 1

Introduction

1.1 Overview

The BlackArch Linux guide is divided into several parts:

- Introduction - Provides a broad overview, introduction, and additional helpful project information
- User Guide - Everything a typical user needs to know to effectively use BlackArch
- Developer Guide - How to get started developing for and contributing to BlackArch
- Tool Guide - In-depth tool details along example usages (WIP)

1.2 What is BlackArch Linux?

BlackArch is a complete Linux distribution for penetration testers and security researchers. It is derived from ArchLinux and users can install BlackArch components individually or in groups directly on top of it.

The toolset is distributed as an Arch Linux unofficial user repository so you can install BlackArch on top of an existing Arch Linux installation. Packages may be installed individually or by category.

The constantly expanding repository currently includes over 1300 tools. All tools are thoroughly tested before being added to the codebase to maintain the quality of the repository.

1.3 History of BlackArch Linux

Coming soon...

1.4 Supported platforms

Coming soon...

1.5 Get involved

You can get in touch with the BlackArch team using the following avenues:

Website: `https://www.blackarch.org/`

Mail: team@blackarch.org

IRC: `irc://irc.freenode.net/blackarch`

Twitter: `https://twitter.com/blackarchlinux`

Github: `https://github.com/Blackarch/`

Chapter 2

User Guide

2.1 Installation

The following sections will show you how to setup the BlackArch repository and install packages. BlackArch supports both, installing from the repository using binary packages as well as compiling and installing from sources.

BlackArch is compatible with normal Arch installations. It acts as an unofficial user repository. If you want an ISO instead, see the Live ISO section.

2.1.1 Installing on top of ArchLinux

Run strap.sh as root and follow the instructions. See the following example.

```
curl -O https://blackarch.org/strap.sh
sha1sum strap.sh # should match: 6f152b79419491db92c1fdde3fad2d445f09aae3
sudo ./strap.sh
```

Now download a fresh copy of the master package list and synchronize packages:

```
sudo pacman -Syyu
```

2.1.2 Installing packages

You may now install tools from the blackarch repository.

1. To list all of the available tools, run

```
pacman -Sgg | grep blackarch | cut -d' ' -f2 | sort -u
```

2. To install all of the tools, run

```
pacman -S blackarch
```

3. To install a category of tools, run

```
pacman -S blackarch-<category>
```

4. To see the blackarch categories, run

```
pacman -Sg | grep blackarch
```

2.1.3 Installing packages from source

As part of an alternative method of installation, you can build the BlackArch packages from source. You can find the PKGBUILDs on github. To build the entire repo, you can use the Blackman tool.

- First, you have to install Blackman. If the BlackArch package repository is setup on your machine, you can install Blackman:

```
pacman -S blackman
```

- You can build and install Blackman from source:

```
mkdir blackman
cd blackman
wget https://raw2.github.com/BlackArch/blackarch/master/packages/blackman/PKGBUILD
# Make sure the PKGBUILD has not been maliciously tampered with.
makepkg -s
```

- Or you can install Blackman from the AUR:

```
<whatever AUR helper you use> -S blackman
```

2.1.4 Basic Blackman usage

Blackman is very simple to use, though the flags are different from what you would typically expect from something like pacman. Basic usage has been outlined below.

- Download, compile and install packages:

```
sudo blackman -i package
```

- Download, compile and install whole category:

```
sudo blackman -g group
```

- Download, compile and install all of the BlackArch tools:

```
sudo blackman -a
```

- To list the blackarch categories:

```
blackman -l
```

- To list category tools:

```
blackman -p category
```

2.1.5 Installing from live-, netinstall- ISO or ArchLinux

You can install BlackArch Linux from one of our live- or netinstall-ISOs.
See https://www.blackarch.org/download.html#iso. The following steps are required after the ISO boot up.

- Install blackarch-installer package:

```
sudo pacman -S blackarch-installer
```

- Run

```
sudo blackarch-install
```

Chapter 3

Developer Guide

3.1 Arch's Build System and Repositories

PKGBUILD files are build scripts. Each one tells makepkg(1) how to create a package. PKGBUILD files are written in Bash.

For more information, read (or skim through) the following:

- Arch Wiki: Creating Packages
- Arch Wiki: makepkg
- Arch Wiki: PKGBUILD
- Arch Wiki: Arch Packaging Standards

3.2 Blackarch PKGBUILD standards

For the sake of simplicity, our PKGBUILDs are similar to that of the AUR ones, with a few small differences outlined below. Every package must belong to blackarch at the minimum, there will also be a lot of crossover with multiple packages belonging to multiple groups.

3.2.1 Groups

To allow users to install a specific range of packages quickly and easily, packages have been separated into groups. Groups allow users to simply go ”pacman -S <group name>” in order to pull a lot of packages.

3.2.1.1 blackarch

The blackarch group is the base group that all packages must belong too. This allows users to install every package with ease.

What should be in here: Everything.

3.2.1.2 blackarch-anti-forensic

Packages that are used for countering forensic activities, including encryption, steganography, and anything that modifies files/file attributes. This all includes tools to work with anything in general that makes changes to a system for the purposes of hiding information.

Examples: luks, TrueCrypt, Timestomp, dd, ropeadope, secure-delete

3.2.1.3 blackarch-automation

Packages that are used for tool or workflow automation.

Examples: blueranger, tiger, wiffy

3.2.1.4 blackarch-backdoor

Packages that exploit or open backdoors on already vulnerable systems.

Examples: backdoor-factory, rrs, weevely

3.2.1.5 blackarch-binary

Packages that operate on binary files in some form.

Examples: binwally, packerid

3.2.1.6 blackarch-bluetooth

Packages that exploit anything concerning the Bluetooth standard (802.15.1).

Examples: ubertooth, tbear, redfang

3.2.1.7 blackarch-code-audit

Packages that audit existing source code for vulnerability analysis.

Examples: flawfinder, pscan

3.2.1.8 blackarch-cracker

Packages used for cracking cryptographic functions, ie hashes.

Examples: hashcat, john, crunch

3.2.1.9 blackarch-crypto

Packages that work with cryptography, with the exception of cracking.

Examples: ciphertest, xortool, sbd

3.2.1.10 blackarch-database

Packages that involve database exploitations on any level.

Examples: metacoretex, blindsql

3.2.1.11 blackarch-debugger

Packages that allow the user to view what a particular program is ”doing” in realtime.

Examples: radare2, shellnoob

3.2.1.12 blackarch-decompiler

Packages that attempt to reverse a compiled program into source code.

Examples: flasm, jd-gui

3.2.1.13 blackarch-defensive

Packages that are used to protect a user from malware & attacks from other users.

Examples: arpon, chkrootkit, sniffjoke

3.2.1.14 blackarch-disassembler

This is similar to blackarch-decompiler, and there will probably be a lot of programs that fall into both, however these packages produce assembly output rather than the raw source code.

Examples: inguma, radare2

3.2.1.15 blackarch-dos

Packages that use DoS (Denial of Service) attacks.

Examples: 42zip, nkiller2

3.2.1.16 blackarch-drone

Packages that are used for managing physically engineered drones.

Examples: meshdeck, skyjack

3.2.1.17 blackarch-exploitation

Packages that takes advantages of exploits in other programs or services.

Examples: armitage, metasploit, zarp

3.2.1.18 blackarch-fingerprint

Packages that exploit fingerprint biometric equipment.

Examples: dns-map, p0f, httprint

3.2.1.19 blackarch-firmware

Packages that exploit vulnerabilities in firmware

Examples: None yet, amend asap.

3.2.1.20 blackarch-forensic

Packages that are used to find data on physical disks or embedded memory.

Examples: aesfix, nfex, wyd

3.2.1.21 blackarch-fuzzer

Packages that use the fuzz testing principle, ie ”throwing” random inputs at the subject to see what happens.

Examples: msf, mdk3, wfuzz

3.2.1.22 blackarch-hardware

Packages that exploit or manage anything to do with physical hardware.

Examples: arduino, smali

3.2.1.23 blackarch-honeypot

Packages that act as ”honeypots”, ie programs that appear to be vulnerable services used to attract hackers into a trap.

Examples: artillery, bluepot, wifi-honey

3.2.1.24 blackarch-keylogger

Packages that record and retain keystrokes on another system.

Examples: None yet, amend asap.

3.2.1.25 blackarch-malware

Packages that count as any type of malicious software or malware detection.

Examples: malwaredetect, peepdf, yara

3.2.1.26 blackarch-misc

Packages that don't particularly fit into any categories.

Examples: oh-my-zsh-git, winexe, stompy

3.2.1.27 blackarch-mobile

Packages that manipulate mobile platforms.

Examples: android-sdk-platform-tools, android-udev-rules

3.2.1.28 blackarch-networking

Package that involve IP networking.

Examples: Anything pretty much

3.2.1.29 blackarch-nfc

Packages that use nfc (near-field communications).

Examples: nfcutils

3.2.1.30 blackarch-packer

Packages that operate on or invlove packers.

packers are programs that embed malware within other executables.

Examples: packerid

3.2.1.31 blackarch-proxy

Packages that acts as a proxy, ie redirecting traffic through another node on the internet.

Examples: burpsuite, ratproxy, sslnuke

3.2.1.32 blackarch-recon

Packages that actively seeks vulnerable exploits in the wild. More of an umbrella group for similar packages.

Examples: canri, dnsrecon, netmask

3.2.1.33 blackarch-reversing

This is an umbrella group for any decompiler, disassembler or any similar program.

Examples: capstone, radare2, zerowine

3.2.1.34 blackarch-scanner

Packages that scan selected systems for vulnerabilities.

Examples: scanssh, tiger, zmap

3.2.1.35 blackarch-sniffer

Packages that involve analyzing network traffic.

Examples: hexinject, pytactle, xspy

3.2.1.36 blackarch-social

Packages that primarily attack social networking sites.

Examples: jigsaw, websploit

3.2.1.37 blackarch-spoof

Packages that attempt to spoof the attacker such, in that the attacker doesn't show up as an attacker to the victim.

Examples: arpoison, lans, netcommander

3.2.1.38 blackarch-threat-model

Packages that would be used for reporting/recording the threat model outlined in a particular scenario.

Examples: magictree

3.2.1.39 blackarch-tunnel

Packages that are used to tunnel network traffic on a given network.

Examples: ctunnel, iodine, ptunnel

3.2.1.40 blackarch-unpacker

Packages that are used to extract pre-packed malware from an executable.

Examples: js-beautify

3.2.1.41 blackarch-voip

Packages that operate on voip programs and protocols.

Examples: iaxflood, rtp-flood, teardown

3.2.1.42 blackarch-webapp

Packages that operate on internet-facing applications.

Examples: metoscan, whatweb, zaproxy

3.2.1.43 blackarch-windows

This group is for any native Windows package that runs via wine.

Examples: 3proxy-win32, pwdump, winexe

3.2.1.44 blackarch-wireless

Packages that operates on wireless networks on any level.

Examples: airpwn, mdk3, wiffy

3.3 Repository structure

You can find the main BlackArch git repo here: https://github.com/BlackArch/blackarch. There are also several secondary repos here: https://github.com/BlackArch.

Within the main git repo, there are three important directories:

- docs - Documentation.
- packages - PKGBUILD files.
- scripts - Useful little scripts.

3.3.1 Scripts

Here is a reference for scripts in the `scripts/` directory:

- baaur - Soon, this will upload packages to the AUR.
- babuild - Build a package.
- bachroot - Manage a chroot for testing.
- baclean - Clean old .pkg.tar.xz files from the package repo.
- baconflict - Soon this will replace `scripts/conflicts`.
- bad-files - Find bad files in built packages.
- balock - Obtain or release the package repo lock.
- banotify - Notify IRC about package pushes.

- barelease - Release packages to the package repo.
- baright - Print the BlackArch copyright info.
- basign - Sign packages.
- basign-key - Sign a key.
- blackman - This behaves sort of like pacman but builds from git (not to be confused with nrz's Blackman).
- check-groups - Check groups.
- checkpkgs - Check packages for errors.
- conflicts - Check for file conflicts.
- dbmod - Modify a package database.
- depth-list - Create a list sorted by dependency depth.
- deptree - Create a dependency tree, listing only blackarch-provided packages.
- get-blackarch-deps - Get a list of blackarch dependencies for a package.
- get-official - Get official packages for release.
- list-loose-packages - List packages that are not in groups and are not dependencies for other packages.
- list-needed - List missing dependencies.
- list-removed - List packages that are in the package repo but not in git.
- list-tools - List tools.
- outdated - Look for packages that are out-dated in the package repo relative to the git repo.
- pkgmod - Modify a build package.
- pkgrel - Increment pkgrel in a package.
- prep - Clean up a PKGBUILD file's style and find errors.
- sitesync - Sync between a local copy of the package repo and a remote copy.
- size-hunt - Hunt for large packages.
- source-backup - Backup package source files.

3.4 Contributing to repository

This section shows you how to contribute to the BlackArch Linux project. We accept pull requests of all sizes, from tiny typo fixes to new packages.
For help, suggestions, or questions feel free to contact us.

Everyone is welcome to contribute. All contributions are appreciated.

3.4.1 Required tutorials

Please read the following tutorials before contributing:

- Arch Packaging Standards
- Creating Packages
- PKGBUILD
- Makepkg

3.4.2 Steps for contributing

In order to submit your changes to the BlackArchLinux project, follow these steps:

1. Fork the repository from `https://github.com/BlackArch/blackarch`
2. Hack the necessary files, (e.g. PKGBUILD, .patch files, etc).
3. Commit your changes.
4. Push your changes.
5. Ask us to merge in your changes, preferably through a pull request.

3.4.3 Example

The following example demonstrates submitting a new package to the BlackArch project. We use yaourt (you can use pacaur as well) to fetch a pre-existing PKGBUILD file for **nfsshell** from the AUR and adjust it according to our needs.

3.4.3.1 Fetch PKGBUILD

Fetch the *PKGBUILD* file using yaourt or pacaur:

```
user@blackarchlinux $ yaourt -G nfsshell
==> Download nfsshell sources
x LICENSE
x PKGBUILD
x gcc.patch
user@blackarchlinux $ cd nfsshell/
```

3.4.3.2 Clean up PKGBUILD

Clean up the *PKGBUILD* file and save some time:

```
  user@blackarchlinux nfsshell $ ./blackarch/scripts/prep PKGBUILD
  cleaning 'PKGBUILD'...
  expanding tabs...
  removing vim modeline...
  removing id comment...
  removing contributor and maintainer comments...
  squeezing extra blank lines...
  removing '|| return'...
  removing leading blank line...
  removing $pkgname...
  removing trailing whitespace...
```

3.4.3.3 Adjust PKGBUILD

Adjust the *PKGBUILD* file:

```
  user@blackarchlinux nfsshell $ vi PKGBUILD
```

3.4.3.4 Build the package

Build the package:

```
==> Making package: nfsshell 19980519-1 (Mon Dec  2 17:23:51 CET 2013)
==> Checking runtime dependencies...
==> Checking buildtime dependencies...
==> Retrieving sources...
-> Downloading nfsshell.tar.gz...
% Total    % Received % Xferd  Average Speed   Time    Time     Time
CurrentDload  Upload   Total   Spent    Left  Speed100 29213  100 29213    0
0  48150      0 --:--:-- --:--:-- --:--:-- 48206
-> Found gcc.patch
-> Found LICENSE
...
<lots of build process and compiler output here>
...
==> Leaving fakeroot environment.
==> Finished making: nfsshell 19980519-1 (Mon Dec  2 17:23:53 CET 2013)
```

3.4.3.5 Install and test the package

Install and test the package:

```
  user@blackarchlinux nfsshell $ pacman -U nfsshell-19980519-1-x86_64.pkg.tar.xz
  user@blackarchlinux nfsshell $ nfsshell # test it
```

3.4.3.6 Add, commit and push package

Add, commit and push the package

```
user@blackarchlinux ~/blackarchlinux/packages $ mv ~/nfsshell .
user@blackarchlinux ~/blackarchlinux/packages $ git commit -am nfsshell && git push
```

3.4.3.7 Create a pull request

Create a pull request on github.com

```
firefox https://github.com/<contributor>/blackarchlinux
```

3.4.3.8 Adding a remote for upstream

A smart thing to do if you're working upstream and on a fork is to pull your own fork and add the main ba repo as a remote

```
user@blackarchlinux ~/blackarchlinux $ git remote -v
origin <the url of your fork> (fetch)
origin <the url of your fork> (push)
user@blackarchlinux ~/blackarchlinux $ git remote add upstream https://github.com/blackarch/blac
user@blackarchlinux ~/blackarchlinux $ git remote -v
origin <the url of your fork> (fetch)
origin <the url of your fork> (push)
upstream https://github.com/blackarch/blackarch (fetch)
upstream https://github.com/blackarch/blackarch (push)
```

By default, git should push straight to origin, but make sure your git config is configured correctly. This won't be an issue unless you have commit rights as you won't be able to push upstream without them.

If you do have the ability to commit, you might have more success using *git@github.com:blackarch/blackarch.git* but it's up to you.

3.4.4 Requests

1. Don't add **Maintainer** or **Contributor** comments to *PKGBUILD* files. Add maintainer and contributor names to the AUTHORS section of BlackArch guide.
2. For the sake of consistency, please follow the general style of the other *PKGBUILD* files in the repo and use two-space indentation.

3.4.5 General tips

namcap can check packages for errors.

Chapter 4

Tools Guide

Coming soon...

4.1 Coming Soon

Coming soon...