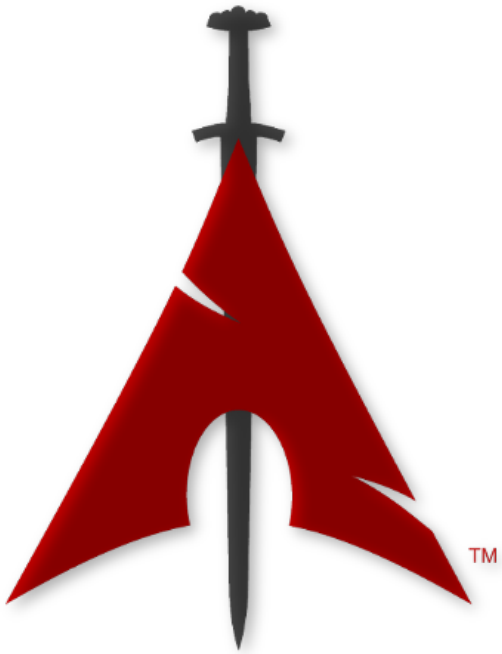

Guia BlackArch Linux

https://www.blackarch.org/

Sumário

Capítulo 1

Introdução

1.1 Resumo

O guia do BlackArch linux é dividido em algumas partes:

- Introdução - Apresenta uma visão geral e informações adicionais sobre o projeto
- Guia do Usuário - Tudo que o usuário precisa saber para usar o BlackArch de forma efetiva
- Guia do Desenvolvedor - Como começar a desenvolver e contribuir para o BlackArch
- Guia das Ferramentas - Detalhes e exemplos de uso das ferramentas

1.2 O que é o BlackArch Linux?

BlackArch é uma distribuição Linux para testes de penetração e pesquisadores de segurança.É uma derivação do ArchLinux e os usuários podem instalar componentes do BlackArch individualmente ou por grupos em cima da distribuição. As ferramentas são distribuídas através do Arch Linux unofficial user repository então você pode instalar o BlackArch em cima de uma instalação Arch Linux já existente.Os pacotes pode ser instalados individualmente ou por categorias.

O repositório continua aumentando e já possui 1300 ferramentas. Todas as ferramentas são testadas depois da adição de algum codigo mantendo a qualidade do repositório.

1.3 A historia do BlackArch Linux

Em breve...

1.4 Plataformas suportadas

Em breve...

1.5 Junte-se ao BlackArch

Você pode se comunicar com a equipe BlackArch através das formas abaixo:

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/

Capítulo 2

Guia do Usuário

2.1 Instalação

A seção seguinte mostra como configurar o repositório BlackArch e instalar os pacotes. O BlackArch suporta a instalação direta pelo repositório usando pacotes binários ou compilando, como também através de outras fontes. O BlackArch é compatível como toda instalação do Arch.A instalação é feita através de um repositório não oficial, se você quiser usar uma ISO olhe na seção Live ISO.

2.1.1 Instalando em cima do ArchLinux

Execute strap.sh como root e siga as instruções abaixo.

```
curl -O https://blackarch.org/strap.sh
sha1sum strap.sh # Deve ser igual a: 6f152b79419491db92c1fdde3fad2d445f09aae3
sudo ./strap.sh
```

Agora baixe uma copia atualizada da lista de pacotes e os sincronize.

```
sudo pacman -Syyu
```

2.1.2 Instalando os pacotes

Agora você pode instalar os pacotes do repositório do blackarch.

1. Para listar todas as ferramentas disponíveis execute

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

2. Para instalar todas a ferramentas execute

   ```
   pacman -S blackarch
   ```

3. Para instalar toda uma categoria execute

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

4. Para ver as categorias execute

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

2.1.3 Instalando os pacotes pelo codigo fonte

De maneira alternativa você pode criar os pacotes do BlackArch diretamente do codigo fonte.Você pode achar os PKGBUILDs no github.Para criar todo o repositório você pode usar a ferramenta Blackman.

- Primeramente, se você tem que instalar o Blackman, se o repositório do BlackArch está configurado na sua máquina, você pode instalar o Blackman:

```
pacman -S blackman
```

- Você pode construir e instalar o Blackman pelo código fonte:

```
mkdir blackman
cd blackman
wget https://raw2.github.com/BlackArch/blackarch/master/packages/blackman/PKGBUILD
# Tenha certeza que o PKGBUILD nao possui codigo malicioso.
makepkg -s
```

- Ou você pode instalar o Blackman pelo AUR::

```
<quaquer ajudante do AUR > -S blackman
```

2.1.4 Uso básico do Blackman

Blackman possui um uso simples, suas flags são um pouco diferentes do pacman.O básico será mostrado abaixo.

- Baixa, compila e instala o pacote:

```
sudo blackman -i pacote
```

- Baixa, compila e instala toda uma categoria:

```
sudo blackman -g grupo
```

- Baixa, compila e instala todas as ferramentas do BlackArch:

```
sudo blackman -a
```

- Lista todas as categorias de ferramentas:

```
blackman -l
```

- Lista uma categoria de ferramentas:

```
blackman -p categoria
```

2.1.5 Instalação pelo live-, netinstall- ISO ou ArchLinux

Você pode instalar o BlackArch através de um live- ou netinstall-ISOs.
Veja em https://www.blackarch.org/download.html#iso.Os passos a seguir são executados depois da inicialização da ISO.

- Instalando o pacote blackarch-installer:

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

- Execute

```
sudo blackarch-install
```

Capítulo 3

Guia do Desenvolvedor

3.1 Construir sistemas e repositórios Arch

Os PKGBUILD são scripts de construição.Cada um diz ao makepkg(1) como construir o pacote.Os arquivos PKGBUILD são escritos em Bash.

Para mais informações, ler abaixo:

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

3.2 Padrão PKGBUILD do Blackarch

Por sua simplicidade, os PKGBUILDs são similares aos do AUR, como algumas pequenas diferenças como mostrado a baixo.Cada pacote deve pertencer ao blackarch o minimo possível, e deve haver o máximo de crossover com vários pacotes percentes a vários grupos.

3.2.1 Grupos

Permite que os usuários instalem um específico tipo de pacote de forma rápida e fácil, os pacotes são separados por grupos.Os grupos permite que o usuário digite "pacman -S <group name>"e consiga vários pacotes.

3.2.1.1 blackarch

O grupo blackarch é a o grupo base onde todos os pacotes se encontram.Isto permite que o usuário instale todos os pacotes com facilidade.

O que deve estar aqui: Tudo.

3.2.1.2 blackarch-anti-forensic

Pacotes relacionados a o uso de contra forense, incluindo encriptação, estenografia, e tudo para modificar os atributos de arquivo/arquivos. Inclui todas as ferramentas para trabalhar com tudo em geral para modificar um sistema com o propósito de esconder uma informação.

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

3.2.1.3 blackarch-automation

Pacotes para uso de ferramentas ou automatização de tarefas.

Exemplo: blueranger, tiger, wiffy

3.2.1.4 blackarch-backdoor

Pacotes para explorar ou abrir blackdoors em sistemas vulneráveis.

Exemplo: backdoor-factory, rrs, weevely

3.2.1.5 blackarch-binary

Pacotes para modificar arquivos binários de algum modo.

Exemplo: binwally, packerid

3.2.1.6 blackarch-bluetooth

Pacotes para explorar tudo relacionado ao Bluetooth padrão (802.15.1).

Exemplo: ubertooth, tbear, redfang

3.2.1.7 blackarch-code-audit

Pacotes para analisar um codigo em busca de vulnerabilidades.

Exemplo: flawfinder, pscan

3.2.1.8 blackarch-cracker

Pacotes usados para quebrar criptografia.

Exemplo: hashcat, john, crunch

3.2.1.9 blackarch-crypto

Pacotes para trabalhar com criptografia sem ser sua quebra.

Exemplo: ciphertest, xortool, sbd

3.2.1.10 blackarch-database

Pacotes que envolvam exploração de banco de dados em algum nível.

Exemplo: metacoretex, blindsql

3.2.1.11 blackarch-debugger

Pacotes para permitir que o usuário veja o que um programa em particular esta "fazendo"em tempo real.

Exemplo: radare2, shellnoob

3.2.1.12 blackarch-decompiler

Pacotes voltados para converter um programa compilado em codigo fonte.

Exemplo: flasm, jd-gui

3.2.1.13 blackarch-defensive

Pacotes para proteger o usuário de malware & ataques de outros usuários.

Exemplos: arpon, chkrootkit, sniffjoke

3.2.1.14 blackarch-disassembler

Similar ao blackarch-decompiler, possui semelhança em alguns pacotes mas esses são mais voltados a produzir uma saída em assembly, não em codigo fonte.

Exemplo: inguma, radare2

3.2.1.15 blackarch-dos

Pacotes para o uso de ataques DoS (Denial of Service).

Exemplo: 42zip, nkiller2

3.2.1.16 blackarch-drone

Pacotes usados para o manejo físico de engenharia de drones.

Exemplo: meshdeck, skyjack

3.2.1.17 blackarch-exploitation

Pacotes para explorar outros programas ou serviços.

Exemplo: armitage, metasploit, zarp

3.2.1.18 blackarch-fingerprint

Pacotes para explorar equipamentos de biometria digital.

Exemplo: dns-map, p0f, httprint

3.2.1.19 blackarch-firmware

Pacotes para explorar vulnerabilidades no firmware

Exemplo: None yet, amend asap.

3.2.1.20 blackarch-forensic

Pacotes usados para procurar informação em discos físicos ou memoria.

Exemplo: aesfix, nfex, wyd

3.2.1.21 blackarch-fuzzer

Pacotes que usam o principio do teste fuzz, "jogando"números aleatórios com o objetivo de ver o que acontece.

Exemplo: msf, mdk3, wfuzz

3.2.1.22 blackarch-hardware

Pacotes que explorar ou manejam hardware.

Exemplo: arduino, smali

3.2.1.23 blackarch-honeypot

Pacotes que agem como "honeypots"(pote de mel), programas para simular uma vulnerabilidade e cria uma armadilha para os hackers.

Exemplo: artillery, bluepot, wifi-honey

3.2.1.24 blackarch-keylogger

Pacotes para gravar o digito em outros sistemas.

Exemplo: None yet, amend asap.

3.2.1.25 blackarch-malware

Pacotes para qualquer tipo de software malicioso ou detecção de malware.

Exemplo: malwaredetect, peepdf, yara

3.2.1.26 blackarch-misc

Pacotes com nada em particular para encaixar em uma categoria.

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

3.2.1.27 blackarch-mobile

Pacotes para manipulação de sistemas moveis.

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

3.2.1.28 blackarch-networking

Pacotes que involvem uma rede IP.

Exemplo: Praticamente tudo

3.2.1.29 blackarch-nfc

Pacotes que usam nfc (near-field communications).

Exemplo: nfcutils

3.2.1.30 blackarch-packer

Pacotes que operam ou envolvem packers.

packers são programas que juntam malware com outros executáveis.

Exemplo: packerid

3.2.1.31 blackarch-proxy

Pacotes que agem com proxy, como redirecionar o tráfego para outro nó na internet.

Exemplo: burpsuite, ratproxy, sslnuke

3.2.1.32 blackarch-recon

Pacotes que procuram exploits.Mais no grupo umbrella para pacotes similares.

Exemplo: canri, dnsrecon, netmask

3.2.1.33 blackarch-reversing

Este é um grupo umbrella para todo decompilador, disassembler ou programas similares.

Exemplo: capstone, radare2, zerowine

3.2.1.34 blackarch-scanner

Pacotes que scaneiam um sistema em busca de vulnerabilidades.

Exemplo: scanssh, tiger, zmap

3.2.1.35 blackarch-sniffer

Pacotes que envolvem analise de tráfego de rede.

Exemplo: hexinject, pytactle, xspy

3.2.1.36 blackarch-social

Pacotes para ataque em redes sociais.

Exemplo: jigsaw, websploit

3.2.1.37 blackarch-spoof

Pacotes para falsificar um ataque, onde um ataque não é mostrado como uma ameaça a vitima.

Exemplo: arpoison, lans, netcommander

3.2.1.38 blackarch-threat-model

Pacotes que são usados para reportar/gravar um modelo de ameaça em um cenário particular.

Exemplo: magictree

3.2.1.39 blackarch-tunnel

Pacotes que são usados para criar tuneis no tráfego de rede.

Exemplo: ctunnel, iodine, ptunnel

3.2.1.40 blackarch-unpacker

Pacotes que são usados para extrair um malware de um executável.

Exemplo: js-beautify

3.2.1.41 blackarch-voip

Pacotes que operam em programas e protocolos voip.

Exemplo: iaxflood, rtp-flood, teardown

3.2.1.42 blackarch-webapp

Pacotes que operam em aplicações na internet.

Exemplo: metoscan, whatweb, zaproxy

3.2.1.43 blackarch-windows

Este grupo é para todo pacote nativo para windows que é executado através do wine.

Exemplo: 3proxy-win32, pwdump, winexe

3.2.1.44 blackarch-wireless

Pacotes que operam em rede wireless em algum nível.

Exemplo: airpwn, mdk3, wiffy

3.3 Estrutura do repositório

Você pode achar o repositório git BlackArch principal aqui: https://github.com/BlackArch/blackarch. Temos também um secundário repositório aqui: https://github.com/BlackArch.

O repositório git principal, contém três diretórios importantes:

- docs - Documentação.
- packages - Arquivo PKGBUILD.
- scripts - Pequenos scripts.

3.3.1 Scripts

Aqui há uma referencia para o diretório de `scripts/` scripts:

- baaur - Brevemente, irá subir os pacotes para o AUR.
- babuild - Constrói os pacotes.
- bachroot - Manipula o chroot para testes.
- baclean - Limpa velhos arquivos .pkg.tar.xz do repositório.
- baconflict - Irá substiuir o `scripts/conflicts`.
- bad-files - Acha arquivos ruins nos pacotes de construção.
- balock - Obtém ou atualiza pacotes travados.
- banotify - Notifica IRC sobre pushes de pacotes.

- barelease - Atualiza pacotes no repositórios de pacotes.
- baright - Imprime a informação de copyright do BlackArch.
- basign - Assina pacotes.
- basign-key - Assina as chaves.
- blackman - Funciona de forma análoga ao pacman builds construindo pacotes do git(não confundir com o Blackman).
- check-groups - Confere grupos.
- checkpkgs - Confere erro em pacotes.
- conflicts - Confere conflitos em pacotes.
- dbmod - Modifica o banco de dados de pacotes.
- depth-list - Cria uma lista de dependência.
- deptree - Cria uma lista de dependência, somente dos pacotes do BlackArch.
- get-blackarch-deps - Pega uma lista de dependências de pacotes do BlackArch.
- get-official - Pega atualizações oficiais do BlackArch.
- list-loose-packages - Lista os pacotes que estão ou não estão em um grupo que dependem de outros pacotes.
- list-needed - Lista dependências perdidas.
- list-removed - Lista os pacotes que estão no repositório mas não no git.
- list-tools - Lista as ferramentas.
- outdated - Olhar por pacotes desatualizados no repositório comparado ao git.
- pkgmod - Modifica pacotes de construção.
- pkgrel - Adiciona pkgrel em um pacote.
- prep - Limpa arquivos PKGBUILD e procura erros.
- sitesync - Sincroniza entre os pacotes locais e os pacotes no repositório.
- size-hunt - Procura pacotes pesados/grandes.
- source-backup - Realiza backup do codigo fonte dos arquivos.

3.4 Contribuindo para o repositório

Essa seção mostra como você pode contribuir com o projeto BlackArch Linux. Nós aceitamos pull requests de todos os tamanhos, de pequenos consertos até novos pacotes.
Para mais ajuda, sugestões ou questões sinta-se livre para nos contatar.

Todo mundo que queira contribuir será bem vindo.Toda contribuição é bem vinda.

3.4.1 Tutorial de requerimento

Por favor leia este tutorial antes de contribuir:

- Arch Packaging Standards
- Creating Packages
- PKGBUILD
- Makepkg

3.4.2 Passos para contribuir

Quando submeter suas mudanças para o projeto BlackArchLinux, siga estes passos:

1. Fork o repositório em `https://github.com/BlackArch/blackarch`
2. Hackei os arquivos necessários, (e.g. PKGBUILD, .patch files, etc).
3. Commit as mudanças.
4. Push suas mudanças.
5. Nos peça para dar merge em suas mudanças, de preferencialmente um pull request.

3.4.3 Exemplo

O exemplo a seguir demonstra como enviar um novo pacotes para o projeto BlackArch. Nós usamos yaourt (you can use pacaur as well) para buscar arquivos PKGBUILD de **nfsshell** para AUR e modificamos de acordo com a necessidade.

3.4.3.1 Buscando PKGBUILD

Busque o arquivo *PKGBUILD* usando o yaourt ou pacaur:

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

3.4.3.2 Limpar o PKGBUILD

Limpe o arquivo *PKGBUILD* e salve:

```
user@blackarchlinux nfsshell $ ./blarckarch/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 Modificações PKGBUILD

Modifique o arquivo *PKGBUILD*:

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

3.4.3.4 Construa o pacote

Construindo o pacote:

```
==> 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 Instalando e testando os pacotes

Intale e teste os pacotes:

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

3.4.3.6 Adiciona, commit e push o pacote

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

3.4.3.7 Crie um pull request

Create a pull request on github.com

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

3.4.3.8 Adicione remotamente

Se melhor a se fazer se você trabalha remotamente é realizar o fork e o seu pull request para adicionar no repositório principal.

```
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
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)
```

Por padrão o git deve realizar o push para a origem, mas tenha certeza que seu git está configurado corretamente.Isto não deve causar problemas ao menos que você tenhas alguns direitos de commit com não habilitar o push sem o mesmo.

Se você tem a habilidade de commitar, é mais fácil usar o *git@github.com:blackarch/blackarch.git* mas ai é com você.

3.4.4 Requisitações

1. Não adicione **Maintainer** ou **Contributor** comentarios nos arquivos *PKGBUILD*. Adicione o nome do mantedor ou contribuinte na seção de AUTHORS do guia BlackArch.
2. Procure ser consistente, por favor siga o estilo dos outros arquivos *PKGBUILD* no repositório e use dois espaços para identação.

3.4.5 Dicas gerais

namcap para conferir os pacotes com erros.

Capítulo 4

Guia das ferramentas

Em breve...

4.1 Em breve

Em breve...