Installing Bro

Contents

Prerequisites

Before installing Bro, you'll need to ensure that some dependencies are in place.

Required Dependencies

Bro requires the following libraries and tools to be installed before you begin:

- Libpcap (http://www.tcpdump.org)
- OpenSSL libraries (http://www.openssl.org)
- BIND8 library
- Libz
- Bash (for BroControl)
- Python 2.6 or greater (for BroControl)

To build Bro from source, the following additional dependencies are required:

- CMake 2.8 or greater (http://www.cmake.org)
- Make
- C/C++ compiler with C++11 support (GCC 4.8+ or Clang 3.3+)
- SWIG (http://www.swig.org)
- Bison (GNU Parser Generator)
- Flex (Fast Lexical Analyzer)
- Libpcap headers (http://www.tcpdump.org)
- OpenSSL headers (http://www.openssl.org)
- zlib headers
- Python

To install the required dependencies, you can use:

SEARCH

Search

- RPM/RedHat-based Linux:

```
sudo yum install cmake make gcc gcc-c++ flex bison libpcap-devel
```

- DEB/Debian-based Linux:

```
sudo apt-get install cmake make gcc g++ flex bison libpcap-dev l
```

- FreeBSD:

  Most required dependencies should come with a minimal FreeBSD install except for the following.

```
sudo pkg install bash cmake swig bison python py27-sqlite3
```

  For older versions of FreeBSD (especially FreeBSD 9.x), the system compiler is not new enough to compile Bro. For these systems, you will have to install a newer compiler using pkg; the `clang34` package should work.

  You will also have to define several environment variables on these older systems to use the new compiler and headers similar to this before calling configure:

```
export CC=clang34
export CXX=clang++34
export CXXFLAGS="-stdlib=libc++ -I${LOCALBASE}/include/c++/v1 -L
export LDFLAGS="-pthread"
```

- Mac OS X:

  Compiling source code on Macs requires first installing either Xcode or the "Command Line Tools" (which is a much smaller download). To check if either is installed, run the `xcode-select -p` command. If you see an error message, then neither is installed and you can then run `xcode-select --install` which will prompt you to either get Xcode (by clicking "Get Xcode") or to install the command line tools (by clicking "Install").

  OS X comes with all required dependencies except for CMake, SWIG, and OpenSSL (OpenSSL headers were removed in OS X 10.11, therefore OpenSSL must be installed manually for OS X versions 10.11 or newer). Distributions of these dependencies can likely be obtained from your preferred Mac OS X package management system (e.g. Homebrew, MacPorts, or Fink). Specifically for Homebrew, the `cmake`, `swig`, and `openssl` packages provide the required dependencies. For MacPorts, the `cmake`, `swig`, `swig-python`, and `openssl` packages provide the required dependencies.

Optional Dependencies

Bro can make use of some optional libraries and tools if they are found at build time:

- C++ Actor Framework (CAF) version 0.14 (http://actor-framework.org)
- LibGeoIP (for geolocating IP addresses)
- sendmail (enables Bro and BroControl to send mail)
- curl (used by a Bro script that implements active HTTP)
- gperftools (tcmalloc is used to improve memory and CPU usage)

- jemalloc (http://www.canonware.com/jemalloc/)
- PF_RING (Linux only, see Cluster Configuration)
- ipsumdump (for trace-summary; http://www.cs.ucla.edu/~kohler/ipsumdump)

LibGeoIP is probably the most interesting and can be installed on most platforms by following the instructions for installing libGeoIP and the GeoIP database.

Installing Bro

Bro can be downloaded in either pre-built binary package or source code forms.

Using Pre-Built Binary Release Packages

See the bro downloads page for currently supported/targeted platforms for binary releases and for installation instructions.

- Linux Packages

  Linux based binary installations are usually performed by adding information about the Bro packages to the respective system packaging tool. Then the usual system utilities such as `apt`, `dnf`, `yum`, or `zypper` are used to perform the installation.

The primary install prefix for binary packages is `/opt/bro`.

Installing from Source

Bro releases are bundled into source packages for convenience and are available on the bro downloads page.

Alternatively, the latest Bro development version can be obtained through git repositories hosted at `git.bro.org`. See our git development documentation for comprehensive information on Bro's use of git revision control, but the short story for downloading the full source code experience for Bro via git is:

```
git clone --recursive git://git.bro.org/bro
```

> **Note**
>
> If you choose to clone the `bro` repository non-recursively for a "minimal Bro experience", be aware that compiling it depends on several of the other submodules as well.

The typical way to build and install from source is (for more options, run `./configure --help`):

```
./configure
make
make install
```

If the `configure` script fails, then it is most likely because it either couldn't find a required dependency or it couldn't find a sufficiently new version of a dependency. Assuming that you already installed all required dependencies, then you may need to use one of the `--with-*` options that can be given to the `configure` script to help it locate a dependency.

The default installation path is `/usr/local/bro`, which would typically require root privileges when doing the `make install`. A different installation path can be chosen by specifying the `configure` script `--prefix` option. Note that `/usr` and `/opt/bro` are the standard prefixes for binary Bro packages to be installed, so those are typically not good choices unless you are creating such a package.

OpenBSD users, please see our FAQ if you are having problems installing Bro.

Depending on the Bro package you downloaded, there may be auxiliary tools and libraries available in the `aux/` directory. Some of them will be automatically built and installed along with Bro. There are `--disable-*` options that can be given to the configure script to turn off unwanted auxiliary projects that would otherwise be installed automatically. Finally, use `make install-aux` to install some of the other programs that are in the `aux/bro-aux` directory.

Finally, if you want to build the Bro documentation (not required, because all of the documentation for the latest Bro release is available on the Bro web site), there are instructions in `doc/README` in the source distribution.

Configure the Run-Time Environment

You may want to adjust your `PATH` environment variable according to the platform/shell/package you're using. For example:

Bourne-Shell Syntax:

```
export PATH=/usr/local/bro/bin:$PATH
```

C-Shell Syntax:

```
setenv PATH /usr/local/bro/bin:$PATH
```

Or substitute `/opt/bro/bin` instead if you installed from a binary package.