libhostile Documentation

Release 1.0.0-161a575

Andrew Hutchings

May 12, 2017

Contents

Release 1.0.0-161a575

Date May 12, 2017

CHAPTER 1

About libhostile

> "What if I tossed small pox into a room filled with sprinters after filling their water cups with red bull."
>
> —Brian Aker

Libhostile is a lightweight library to be used with *LD_PRELOAD* which will intentially break certain libc calls at various intervals. Specifically designed to cause random failures in network functions and memory allocation.

The tool was originally designed by Data Differential as part of Gearman but is now being continued as an independent project maintained by Andrew Hutchings.

CHAPTER 2

Contents

Compiling libhostile

Simply download the source and run:

```
$ autoreconf -fi
$ ./configure
$ make
$ sudo make install
```

Using libhostile

libhostile adds a very thin layer over every call that it breaks. By default it will not break any calls, it needs to be told what calls you wish to be hostile to. There are two ways of doing this:

1. Using the *hostile.sh* tool
2. Manually as below

Manual hostility

You can manually invoke libhostile using `LD_PRELOAD` as follows:

```
$ LD_PRELOAD=/usr/local/lib/libhostile.so application
```

Where `application` is the application/service you wish to be hostile to.

To become hostile to libc functions you will need to set environment variables. Multiple environment variables can be used in the same run.

Each environment variable should be set to an integer representation of the frequency of failures. For example:

```
$ export HOSTILE_MALLOC=500
```

The above will tell libhostile to return an error for roughly 1/500 `malloc()` calls (there is a random element and a small grace period).

HOSTILE_ACCEPT
Become hostile to `accept()` calls

HOSTILE_ACCEPT4
Become hostile to `accept4()` calls

HOSTILE_CLOSE
Become hostile to `close()` calls

HOSTILE_CONNECT
Become hostile to `connect()` calls

HOSTILE_GETADDRINFO
Become hostile to `getaddrinfo()` calls

HOSTILE_MALLOC
Become hostile to `malloc()` calls

HOSTILE_PIPE
Become hostile to `pipe()` calls

HOSTILE_PIPE2
Become hostile to `pipe2()` calls

HOSTILE_POLL
Become hostile to `poll()` calls

HOSTILE_REALLOC
Become hostile to `realloc()` calls

HOSTILE_RECV
Become hostile to `recv()` calls

HOSTILE_SEND
Become hostile to `send()` calls

HOSTILE_SETSOCKOPT
Become hostile to `setsockopt()` calls

HOSTILE_SOCKET
Become hostile to `socket()` calls

HOSTILE_WRITE
Become hostile to `write()` calls

FAQ

1. Why do I not get the familar "Hostile Engaged" when trying to be hostile to certain binaries?

   This is because `LD_PRELOAD` will not work on binaries that do not have the same set-user-ID/set-group-ID as the libarary. This is from the documentation of `LD_PRELOAD`:

   > A whitespace-separated list of additional, user-specified, ELF shared libraries to be loaded before all others. This can be used to selectively override functions in other shared libraries. For set-

user-ID/set-group-ID ELF binaries, only libraries in the standard search directories that are also set-user-ID will be loaded.

hostile.sh

The `hostile.sh` script is a wrapper for `libhostile`. It will do the `LD_PRELOAD` for you as well as setup the required calls you want to be hostile with. It should be used as follows:

```
$ hostile.sh [options] <program> [program_options]
```

Where `<program>` is the application/service you wish to be hostile to and `[program_options]` are the options for it. For example, this would be hostile to `malloc()` calls with `curl`:

```
$ hostile.sh -m 500 curl -L http://www.google.com/
```

With all the below options the `frequency` is roughly how often the call should fail. So a frequency of 500 will fail roughly every 1/500 calls (there is a random element to it and a small grace period).

-a `frequency`
: Become hostile on `accept()` and `accept4()` calls.

-c `frequency`
: Become hostile on `connect()` calls

-e `frequency`
: Become hostile on `recv()` calls

-g `frequency`
: Become hostile on `getaddrinfo()` calls

-l `frequency`
: Become hostile on `poll()` calls

-m `frequency`
: Become hostile on `malloc()` calls

-o `frequency`
: Become hostile on `socket()` calls

-p `frequency`
: Become hostile on `pipe()` and `pipe2()` calls

-r `frequency`
: Become hostile on `realloc()` calls

-s `frequency`
: Become hostile on `send()` calls

-t `frequency`
: Become hostile on `setsockopt()` calls

-w `frequency`
: Become hostile on `write()` calls

-x `frequency`
: Become hostile on `close()` calls

- genindex
- search

Index

Symbols

E

H