

Annobin

The ANNOBIN plugin
(Annobin)
Version 10.0

Nick Clifton

This manual describes the ANNOBIN plugin and the `annocheck` program, and how you can use them to determine what security features were used when a program was built.

Copyright © 2018 - 2021 Red Hat

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

Table of Contents

1	What is Binary Annotation ?	1
2	How to add Binary Annotations to your application.	3
3	How to examine the information stored in the binary.	9
3.1	Encoding Protocol and Producer Versions.....	9
3.2	Encoding Stack Protections.....	10
3.3	Encoding Position Independence.....	10
3.4	Encoding Optimization and Debugging Levels.....	11
3.5	Encoding Control Flow Protection.....	11
3.6	Encoding the Size of Enumerations.....	12
3.7	Encoding Instrumentation Options.....	12
4	Analysing binary files.	13
4.1	The builder checker.....	14
4.2	The Hardened security checker.....	15
4.2.1	The bind-now test.....	16
4.2.2	The gnu-stack test.....	17
4.2.3	The writeable-got test.....	17
4.2.4	These tests need extended documentation.....	18
4.2.5	How to waive the results of annocheck tests.....	21
4.3	The annobin note displayer.....	21
4.4	The section size recorder.....	22
4.5	How long did the check take ?.....	22
5	Configuring annobin and annocheck	25
6	How to use the information stored in the binary.	27
6.1	The built-by script.....	27
6.2	The check-abi script.....	29
6.3	The hardened script.....	30
6.4	The run-on-binaries-in script.....	32
Appendix A GNU Free Documentation License		35

1 What is Binary Annotation ?

Binary Annotation is a method for recording information about an application inside the application itself. It is an implementation of the **Watermark** specification defined here: <https://fedoraproject.org/wiki/Toolchain/Watermark>

Although mainly focused on recording security information, the system can be used to record any kind of data, even data not related to the application. One of the main goals of the system however is the ability to specify the address range over which a given piece of information is valid. So for example it is possible to specify that all of a program was compiled with the `-O2` option except for one special function which was compiled with `-O0` instead.

The range information is useful because it allows third parties to examine the binary and find out if its construction was consistent. IE that there are no gaps in the recorded information, and no special cases where a required feature was not active.

The system works by adding special sections to the application containing individual pieces of information along with an address range for which the information is valid. (Some effort has gone into the storing this information in a reasonably compact format).

The information is generated by a plugin that is attached to the compiler. The plugin extracts information from the internals of compiler and records them in the object file(s) being produced.

Note - the plugin method is just one way of generating the information. Any interested party can create and add information to the object file, providing that they follow the **Watermark** specification.

The information can be extracted from files via the use of tools like `readelf` and `objdump`. The `annobin` package itself includes a program called `annocheck` which can also examine this information. Details on this program can be found elsewhere in this documentation.

2 How to add Binary Annotations to your application.

Normally the option to enable the recording of binary annotation notes is enabled automatically by the build system, so no user intervention is required. On Fedora and RHEL based systems this is handled by the ‘`redhat-rpm-config`’ package.

Currently the binary annotations are generated by a plugin to the compiler (GCC, clang or llvm). This does mean that files that are not compiled by any of these compilers will not gain any annotations, although there is an optional assembler switch to add some basic notes if none are present in the input files.

If the build system being used does not automatically enable the ‘`annobin`’ plugin then it can be specifically added to the compiler command line by adding the `-fplugin=annobin` (for gcc) or `-fplugin=annobin-for-clang` (for clang) or `-fplugin=annobin-for-llvm` (for LLVM) option. It may also be necessary to tell the compiler where to find the plugin by adding the `-ipluginidir=` option, although this should only be necessary if the plugin is installed in an unusual place.

If it is desired to disable the recording of binary annotations then the `-fplugin-arg-annobin-disable` (for gcc) or `-Xclang -plugin-arg-annobin-disable` (for clang or llvm) can be used. Note - these options must be placed *after* the `-fplugin=annobin` option.

On Fedora and RHEL systems the plugin can be disabled entirely for all compilations in a package by adding `%undefine _annotated_build` to the spec file.

The plugin accepts a small selection of command line arguments, all accessed by passing `-fplugin-arg-annobin-<option>` (for gcc) or `-Xclang -plugin-arg-annobin-<option>` (for clang or llvm) on the command line. These options must be placed on the command line after the plugin itself is mentioned. Note - not all versions of the plugin accept all of these options. The options are:

<code>disable</code>	
<code>enable</code>	Either disable or enable the plugin. The default is for the plugin to be enabled.
<code>help</code>	Display a list of supported options on the standard output. This is in addition to whatever else the plugin has been instructed to do.
<code>version</code>	Display the version of the plugin on the standard output. This is in addition to whatever else the plugin has been instructed to do.
<code>verbose</code>	Report the actions that the plugin is taking. If invoked for a second time on the command line the plugin will be very verbose.

function-verbose

Report the generation of function specific notes. This indicates that the named function was compiled with different options from those that were globally enabled.

stack-size-notes**no-stack-size-notes**

Do, or do not, record information about the stack requirements of functions in the executable. This feature is disabled by default as these notes can take up a lot of extra room if the executable contains a lot of functions.

stack-threshold=N

If stack size requirements are being recorded then this option sets the minimum value to record. Functions which require less than N bytes of static stack space will not have their requirements recorded. If not set, then N defaults to 1024.

global-file-syms**no-global-file-syms**

If enabled the **global-file-syms** option will create globally visible, unique symbols to mark the start and end of the compiled code. This can be desirable if a program consists of multiple source files with the same name, or if it links to a library that was built with source files of the same name as the program itself. The disadvantage of this feature however is that the unique names are based upon the time of the build, so repeated builds of the same source will have different symbol names inside it. This breaks the functionality of the build-id system which is meant to identify similar builds created at different times. This feature is disabled by default, and if enabled can be disabled again via the **no-global-file-syms** option.

attach**no-attach**

When gcc compiles code with the **-ffunction-sections** option active it will place each function into its own section. When the annobin **attach** option is active the plugin will attempt to attach the function section to a group containing the notes and relocations for the function. In that way, if the linker decides to discard the function, it will also know that it should discard the notes and relocations as well.

The default is **attach**, but this can be disabled via the **no-attach** option. Note however that if both **attach** and **link-order** are disabled then note generation for function sections will not work properly.

`link-order`

`no-link-order`

As an alternative to using section groups and a special assembler directive the plugin can use a feature of the ELF `SHF_LINK_ORDER` flag which tells the linker that it should discard a section if the section it is linked to is also being discarded. This behaviour is enabled by the `link-order` option.

`rename`

Adds an extra prefix to the symbol names generated by the `annobin` plugin. This allows the plugin to be run twice on the same executable, which can be useful for debugging and build testing.

`active-checks`

`no-active-checks`

The `annobin` plugin will normally generate a warning message if it detects that the `-D_FORTIFY_SOURCE=2` has not been provided on the command line and `-flto` has been enabled. This is because LTO compilation hides preprocessor options, so information about them cannot be passed on to the `annocheck` tool.

The `active-checks` option changes the warning message into an error message, just as if `-Werror` had been specified.

The `no-active-checks` option disables the warning message entirely.

Note - in the future the `annobin` plugin might be extended to produce warning messages for other missing command line options.

`dynamic-notes`

`no-dynamic-notes`

`static-notes`

`no-static-notes`

These options are deprecated.

`ppc64-nops`

`no-ppc64-nops`

This option either enables or disables the insertion of NOP instructions in the some of the code sections of PowerPC64 binaries. This is necessary to avoid problems with the `elflint` program which will complain about binaries built without this option enabled. The option is enabled by default, but since it does increase the size of compiled programs by a small amount, the `no-ppc64-nops` is provided in order to turn it off.

The plugins record information appropriate to the compiler that is running them. So the `gcc` plugin records information about the following options:

```

-D_FORTIFY_SOURCE=[2|3]
-D_GLIBCXX_ASSERTIONS
-O
-Wall
-fPIC
-fPIE
-fcf-protection
-finstrument_functions
-flto
-fomit-frame-pointer
-fprofile
-fprofile-arcs
-fsanitize
-fshort-enums
-fstack-clash-protection
-fstack-protector
-g
-mbranch-protection (AArch64)
-mstack-realign (i386)
-mtls-size (PowerPC)

```

The Clang plugin records information on the following command line options:

```

-O
-Wall
-fPIC
-fPIE
-fcf-protection-branch
-fcf-protection-return
-fsanitize=cfi-cast-strict
-fsanitize=safe-stack
-fspeculative-load-hardening
-fstack-protector-strong

```

Note - if LTO compilation is enabled (-flto) then any data recorded by the Clang plugin is ignored when the object file is recompiled by the LLVM backend. Hence when using LTO and Clang it is best to enable the LLVM plugin.

The LLVM plugin records information on the following command line options:

```

-D_FORTIFY_SOURCE=[2|3]
-O

```

```
-Wall  
-flto  
-fPIC  
-fPIE  
-fcf-protection-branch  
-fcf-protection-return  
-fsanitize=safe-stack  
-fstack-protector-strong  
-g
```


3 How to examine the information stored in the binary.

The information is stored in the ELF Note format in a special section called `.gnu.build.attributes`. The `readelf` program from the `binutils` package can extract and display these notes when the `--notes` option is provided. (Adding the `--wide` option is also helpful). Here is an example of the output:

```

Displaying notes found in: .gnu.build.attributes
  Owner                Data size Description
  GA$<version>3p3      0x00000010 OPEN    Applies to region from 0x8a0 to 0x8c
  GA$<tool>gcc 7.2.1 20170915 0x00000000 OPEN    Applies to region from 0x8a0 to 0x8c
  GA*GOW:0x452b        0x00000000 OPEN    Applies to region from 0x8a0 to 0x8c
  GA*<stack prot>strong 0x00000000 OPEN    Applies to region from 0x8a0 to 0x8c
  GA*GOW:0x412b        0x00000010 func    Applies to region from 0x8c0 to 0x8c

```

This shows various different pieces of information, including the fact that the notes were produced using version 3 of the specification, and version 3 of the plugin. The binary was built by `gcc` version 7.2.1 and the `-fstack-protector-strong` option was enabled on the command line. The program was compiled with `-O2` enabled except the `baz()` function which was compiled with `-O0` instead.

The most complicated part of the notes is the owner field. This is used to encode the type of note as well as its value and possibly extra data as well. The format of the field is explained in detail in the Watermark specification, but it basically consists of the letters ‘G’ and ‘A’ followed by an encoding character (one of ‘*’ ‘!’ ‘+’) and then a type character and finally the value.

The notes are always four byte aligned, even on 64-bit systems. This does mean that consumers of the notes may have to read 8-byte wide values from 4-byte aligned addresses, and that producers of the notes may have to generate unaligned relocs when creating them.

Most of the notes have a reasonably self explanatory name and value. The exception are the `version` and `GOW` notes, which are included in the table below.

3.1 Encoding Protocol and Producer Versions

The `version` note encodes the version of the Watermark specification used and the version of the tool used to generate the notes. Typically the protocol version will be 3 and the plugin version will be 9. It also encodes the tool used to generate the notes as a single character. The following characters are used:

- L The notes have been produced by the Clang plugin.
- V The notes have been produced by the LLVM plugin.
- a The notes have been produced by the assembler.

- c** The notes have been produced by the gcc plugin for the `.text.cold` section.
- e** The notes have been produced by the gcc plugin for the `.text.exit` section.
- g** The notes have been produced by the gcc plugin when running in LTO mode.
- h** The notes have been produced by the gcc plugin for the `.text.hot` section.
- l** The notes have been produced by the linker.
- p** The notes have been produced by the gcc plugin.
- s** The notes have been produced by the gcc plugin for the `.text.startup` section.

3.2 Encoding Stack Protections

The stack protection note (value 2) encodes the setting of the `-fstack-protector` option. Possible values are:

- 0** Not compiled with any setting of `-fstack-protector` (or the setting is unknown).
- 1** Compiled with just `-fstack-protector`.
- 2** Compiled with `-fstack-protector-all`.
- 3** Compiled with `-fstack-protector-strong`.
- 4** Compiled with `-fstack-protector-explicit`.

3.3 Encoding Position Independence

The `Position Independence Status` note encodes the setting of the `-fpic/-fpie` used when compiling the program. The value of the note can be

- 0** Static code, ie neither `pic` nor `pie`.
- 1** Compiled with `-fpic`.
- 2** Compiled with `-fPIC`.
- 3** Compiled with `-fpie`.
- 4** Compiled with `-fPIE`

If both `pic` and `pie` have been specified on the command line then `pie` takes the precedence in the encoding.

3.4 Encoding Optimization and Debugging Levels

The `GOW` note encodes the optimization level (`-O`) and debugging level (`-g`) used when compiling a binary. In order to save space this is stored as a bit field with the bits having the following meanings:

- `bits 0 - 2` The debug type, ie DBX, DWARF, VMS or XCOFF. As specified by the `-gstabs`, `-gdwarf`, `-gvms` and `-gxcoff` options.
- `bit 3` Set if GNU extensions to the debug type have been enabled.
- `bits 4 - 5` The debug info level ie TERSE, NORMAL or VERBOSE as set by the `-g<level>` option.
- `bits 6 - 8` The DWARF version, if DWARF is being generated. Set by the `-gdwarf-<version>` option.
- `bits 9 - 10` The optimization level as set by the `-O<number>` option. Levels above 3 are treated as if they were 3.
- `bit 11` Set if the optimize-for-size option (`-Os`) is enabled.
- `bit 12` Set if the inaccurate-but-fast optimization option (`-Ofast`) has been enabled.
- `bit 13` Set if the optimize-with-debugging option (`-Og`) has been enabled.
- `bit 14` Set if the enable most warnings option (`-Wall`) has been enabled.
- `bit 15` Set if the format security warning option (`-Wformat-security`) has been enabled.
- `bit 16` Set if LTO compilation has been enabled.
- `bit 17` Set if LTO compilation has been disabled. This bit is here so that tools can detect notes created by earlier versions of annobin which did not set any bits higher than 15.

The other bits are not currently used and should be set to zero so they can be used in future extensions to the specification.

3.5 Encoding Control Flow Protection

Records the setting of the `-cf-protection` option. This is a bit mask using the following bits, based upon the definition of the `enum cf_protection_level` from gcc's `flag-types.h` header file:

- `bit 0` Branches are protected. (ie `-fcf-protection=branch`).
- `bit 1` Returns are protected. (ie `-fcf-protection=return`).

bit 2 If set, this indicates that the other bits were explicitly set by an option on the gcc command line. Otherwise those bits were implicitly set by either other options or the backend concerned.

If both bits 0 and 1 are set then this implies the `-fcf-protection=full` option, and if neither are set then this implies the `-fcf-protection=none` option.

Note - in order to avoid storing a value of 0 in the note (which can be confused with a NUL-byte to indicate the end of a string), the value stored is biased by 1.

3.6 Encoding the Size of Enumerations

Record the value of the `-fshort-enums` option. Possible values are:

true The `-fshort-enums` option has been enabled.

false The `-fshort-enums` option has not been enabled.

3.7 Encoding Instrumentation Options

Records the enablement of various code instrumentation options. Note - this note is only produced if one or more of these options are enabled.

The note encodes four values, separate by the forward slash (/) character. These values are:

sanitization

Enabled via a plethora of `-fsanitize=...` options these tell gcc to add extra code to help with various different types of error checking features.

function instrumentation

Enabled via gcc's `-finstrument-functions` option, this adds special function calls at the entry and exit point of every normal function.

profiling

Enabled via gcc's `-p` or `-pg` options, this adds instrumentation to the compiled code that generates output suitable for analysis via the `prof` or `gprof` programs.

arc profiling

Enabled via gcc's `-fprofile-arc` option, or one of the meta-profiling options, this option adds code to record how many times every branch and function call is executed.

Each value represents a setting of an internal gcc flag variable. The exact meaning of the values is specific to gcc, but any non-zero number means that the feature has been enabled in some way.

4 Analysing binary files.

```

annockeck
  [-help]
  [-help-tool]
  [-version]
  [-verbose]
  [-quiet]
  [-ignore-unknown]
  [-report-unknown]
  [-debug-rpm=file]
  [-dwarf-dir=dir]
  [-prefix=text]
  [-enable-tool]
  [-disable-tool]
  [-tool-option]
  file...

```

The **annockeck** program can analyse binary files and report information about them. It is designed to be modular, with a set of self-contained tools providing the checking functionality. Currently the following tools are implemented:

The **annockeck** program is able to scan inside rpm files and libraries. It will automatically recurse into any directories that are specified on the command line. In addition **annockeck** knows how to find debug information held in separate debug files, and it will search for these whenever it needs the resources that they contain.

New tools can be added to the **annockeck** framework by creating a new source file and including it in the **Makefile** used to build **annockeck**. The modular nature of **annockeck** means that nothing else needs to be updated.

New tools must fill out a **struct checker** structure (defined in **annockeck.h**) and they must define a constructor function that calls **annockeck_add_checker** to register their presence at program start-up.

The **annockeck** program supports some generic command line options that are used regardless of which tools are enabled.

```

--debug-rpm=file
    Look in file for separate dwarf debug information.

--dwarf-dir=dir
    Look in dir for separate dwarf debug information files.

--help
    Displays the generic annobin usage information and then exits.

--help-tool
    Display the usage information for tool and then exits.

--report-unknown
--ignore-unknown
    These options have two separate effects (and should really be
    separated into different options). If enabled, unknown file types

```

are reported when they are encountered. This includes non-ELF format files, block devices and so on. Directories are not considered to be unknown and are automatically descended.

The second effect is how symbolic links are handled. If reporting is enabled then they are treated as unknown and reported. If reporting is disabled then they are followed, if possible. Otherwise they are reported as being unresolvable.

The default setting depends upon the file being processed. For rpm files the default is to ignore unknowns, since these often contain non-executable files, and dangling symbolic links. For other file types, including directories, the default is to report unknown files.

- `--prefix=text`
Include *text* in the output description.
- `--quiet` Do not print anything, just return an exit status.
- `--verbose`
Produce informational messages whilst working. Repeat for more information.
- `--version`
Report the version of the tool and then exit.
- `--enable-tool`
Enable *tool*. Most tools are disabled by default and so need to be enabled via this option before they will act.
- `--disable-tool`
Disable *tool*. Normally used to disable the hardening checker, which is enabled by default.
- `--tool-option`
Pass *option* on to *tool*.

Any other command line options will be passed to the tools in turn in order to give them a chance to claim and process them.

4.1 The builder checker.

```

annocheck
  [-disable-hardened]
  --enable-builtby
  [-all]
  [-tool=name]
  [-nottool=name]
  file...

```

The *built-by* tool is disabled by default, but it can be enabled by the command line option `--enable-builtby`. The tool checks the specified files to see if any information is stored about how the file was built.

Since the hardening checker is enabled by default it may also be useful to add the `--disable-hardened` option to the command line.

The tool supports a few command line options to customise its behaviour:

`--all` Report all builder identification strings. The tool has several different heuristics for determining the builder. By default it will report the information return by the first successful heuristic. If the `--all` option is enabled then all successful results will be returned.

`--tool=name` This option can be used to restrict the output to only those files which were built by a specific tool. This can be useful when scanning a directory full of files searching for those built by a particular compiler.

`--nottool=NAME` This option can be used to restrict the output to only those files which were not built by a specific tool. This can be useful when scanning a directory full of files searching for those that were not built by a particular compiler.

4.2 The Hardened security checker.

```

annocheck
  [-skip-all]
  [-skip-bind-now]
  [-skip-branch-protection]
  [-skip-cf-protection]
  [-skip-dynamic-segment]
  [-skip-dynamic-tags]
  [-skip-entry]
  [-skip-fortify]
  [-skip-future]
  [-skip-glibcxx-assertions]
  [-skip-gnu-relro]
  [-skip-gnu-stack]
  [-skip-lto]
  [-skip-optimization]
  [-skip-pic]
  [-skip-pie]
  [-skip-production]
  [-skip-property-note]
  [-skip-run-path]
  [-skip-rwx-seg]
  [-skip-short-enum]
  [-skip-stack-clash]
  [-skip-stack-prot]
  [-skip-stack-realign]
  [-skip-textrel]

```

```

[-skip-threads]
[-skip-warnings]
[-skip-writeable-got]
[-test-name]
[-test-all]
[-test-future]
[-profile-el7]
[-profile-el9]
[-profile-rawhide]
[-ignore-gaps]
[-fixed-format-messages]
[-disable-colour]
[-enable-colour]
[-disable-hardened]
[-enable-hardened]
[-full-filenames]
[-base-filenames]
file...

```

The *hardened* tool checks that the specified files were compiled with specific security hardening features enabled. The features that are tested can be specified via command line options, but the default is to test for all of them.

New tests can be added to the *hardened* checker by adding an entry in the *tests* array defined in *hardened.c* and then creating the necessary code to support the test. There is more information on this process in this blog: <https://developers.redhat.com/articles/2021/07/15/build-your-own-tool-search-code-sequences-binary-files>

Currently the *hardened* tool can run the following tests:

4.2.1 The bind-now test

```

Summary: An attacker could intercept calls to shared library functions
Fix By:  Add -Wl,-z,now to final link command line
Waive If: No shared libraries used

```

```

Example: FAIL: bind-now test because not linked with -Wl,-z,now

```

This test checks that lazy binding is not enabled in the binary. Lazy binding can be used to delay resolving the links between an application and any shared libraries that it uses:

```

https://www.airs.com/blog/archives/41

```

Using lazy binding provides a faster start-up for an application since this resolving process is not performed until a function call is made to a specific library. But it is also a security vulnerability since an attacker could replace the binding with a link to their own code. Hence for security purposes immediate binding rather than lazy binding should be used.

The type of binding is selected via a linker command line option, and on a compiler command line the secure version usually looks like `-Wl,-z,now`. The lazy binding option is `-Wl,-z,lazy` although some linkers are

configured to use lazy binding by default, in which case just the absence of the `-Wl,-z,now` option is enough to trigger this test.

Whilst important, this test can be ignored if the binary does not use any shared libraries. The test can be disabled via the `--skip-bind-now` option and re-enabled by the `--test-bind-now` option.

4.2.2 The gnu-stack test

Summary: An attacker could place code on the stack and then run it
 Fix By: Updating assembler sources and/or linker script
 Waive If: The application really really needs to be able to dynamically create and execute code

Example: FAIL: the gnu-stack test because the .stack section has incorrect permissions

Example: FAIL: the gnu-stack test because the .note.GNU-stack section has execute permission

Example: FAIL: the gnu-stack test because the GNU stack segment has execute permission

Example: FAIL: the gnu-stack test because the GNU stack segment does not have both r

Example: FAIL: the gnu-stack test because no .note.GNU-stack section found

Example: MAYB: the gnu-stack test because multiple stack sections detected

This test checks that it is not possible to place code onto the stack and then execute it. Normally the stack just holds data and addresses, but never instructions. A favourite tactic of attackers however is to discover a buffer overrun bug that addresses the stack and then place instructions there before forcing the processor to execute them.

The test actually checks several different parts of a binary file in order to determine that its stack is safe, which is why there are several different potential failure messages.

Most applications will have a section inserted into them by the compiler called *.note.GNU-stack*. The section has no contents, but the read, write, and execute attributes of the section reflect the needs of the application's stack.

Ordinary compiled code should never see this problem, but the test failure can be triggered by programs built from assembler sources or linked with a custom made linker map. To fix the problem either the linker map needs to be updated to ensure that the stack section is not executable or the assembler sources need to be extended to a note that the stack is not executable:

```
.section .note.GNU-stack,"",%progbits
```

If necessary the test can be disabled via the `--skip-gnu-stack` option and re-enabled via the `--test-gnu-stack` option.

4.2.3 The writeable-got test

Summary: An attacker could intercept and redirect shared library function calls

Fix By: Link with `-Wl,--secure-plt`
 Waive If: No shared libraries are used

Example: FAIL `writeable-got` test because the `GOT/PLT` relocs are writeable

This test checks that the instructions to set up the `GOT` and `PLT` tables in a dynamic executable cannot be altered by an outside source.

Dynamic executables use two tables to help them connect to shared libraries. These tables - the `GOT` and the `PLT` - are set up when the program runs, based upon instructions held in special sections in the file. If these sections are writeable then an attacker could change their contents and thus cause the program to call the wrong functions in the shared libraries.

Under normal circumstances this test should never fail. If it does then something unusual is going on. One possible cure is to add the `-Wl,--secure-plt` option to the final link command line.

If necessary the test can be disabled via the `--skip-writeable-got` option and re-enabled via the `--test-writeable-got` option.

4.2.4 These tests need extended documentation

No RWX segments.

No program segment should have all three of the read, write and execute permission bits set. Disabled by `--skip-rwx-seg`.

No text relocations

There should be no relocations against executable code. Disabled by `--skip-textrel`.

Correct runpaths

The runpath information used to locate shared libraries at runtime must only include directories rooted at `/usr`. Disabled by `--skip-run-path`.

Missing annobin data

The program must have been compiled with annobin notes enabled. Disabled by `--ignore-gaps`.

Strong stack protection

The program must have been compiled with the `-fstack-protector-strong` option enabled, and with `-D_FORTIFY_SOURCE=[2|3]` specified. It must also have been compiled at least optimization level 2. Disabled by `--skip-stack-prot`.

Dynamic data present

Dynamic executables must have a dynamic segment. Disabled by `--skip-dynamic-segment`.

Position Independent compilation

Shared libraries must have been compiled with `-fPIC` or `-fPIE` but not `-static`. This check can be disabled by `--skip-pic`.

Dynamic executables must have been compiled with `-fPIE` and linked with `-pie`. This check can be disabled by `--skip-pie`.

Safe exceptions

Program which use exception handling must have been compiled with `-fexceptions` enabled and with `-D_GLIBCXX_ASSERTIONS` specified. Disabled by `--skip-threads` and/or `--skip-glibcxx-assertions`.

Stack Clash protection

If available the `-fstack-clash-protection` must have been used. Disabled by `--skip-stack-clash`.

Control Flow protection

If available the `-fcf-protection=full` option must have been used. Disabled by `--skip-cf-protection`. If this option is disabled then the check for GNU Property notes will also be disabled.

Branch protection

For *AArch64* binaries the `-mbranch-protection` option, if available, must have either not. Disabled by `--skip-branch-protection`.

Stack realignment

For *i686* binaries, the `-mstackrealign` option must have been specified. Disabled by `--skip-stack-realign`.

Source fortification

The program must have been compiled with the `-D_FORTIFY_SOURCE=[2|3]` command line option specified. Disabled by `--skip-fortify`.

Optimization

The program must have been compiled with at least `-O2` optimization enabled. Disabled by `--skip-optimization`.

Link Time Optimization

The program must have been compiled with link time optimization (`-flto`) enabled. Currently this is a soft check, so failing this test is not considered a reason to fail the overall run. Disabled by `--skip-lto`.

Read only relocations

The program must not have any relocations that are held in a writeable section. Disabled by `--skip-gnu-relro`.

GNU Property Note

For *x86_64*, *AArch64* and *PowerPC* binaries, check that a correctly formatted GNU Property note is present. The contents of the notes are architecture specific. Disabled by `--skip-property-note`.

Enum Size Check that the program makes consistent use of the `-fshort-enum` option.

Production Ready Compiler

Check that the program was built by a production-ready compiler. Disabled by `--skip-production`.

The tool does support a couple of other command line options as well:

`--skip-all`

Disable all tests. Not really useful unless followed by...

`--test-name`

Enable test *name*.

`--test-future`

Report *future fail* tests. These are tests for security features which are not yet implemented, but are planned for the future. The `--skip-future` option can be used to restore the default behaviour of skipping these tests.

`--profile-e19`

`--profile-rawhide`

`--profile-e17`

Rather than enabling and disabling specific tests a selection can be chosen via a profile option. The `--profile-e19` option will select the tests suitable for *RHEL-9* binaries. The `--profile-rawhide` option will select tests suitable for *Fedora rawhide* binaries and the `--profile-e17` option will select tests suitable for *RHEL-7* binaries.

Other profiles may be added in the future.

`--disable-hardened`

Disable the tool.

`--enable-hardened`

Enable the tool if it was previously disabled. The option is also the default.

`--ignore-gaps`

Do not complain about gaps in the note data.

`--fixed-format-messages`

Display messages in a fixed, machine parseable format. The format is:

```
Hardened: <result>: test: <test-name> file: <file-name>
```

Where `<result>` is *PASS* or *FAIL* and `<test-name>` is the name of the test, which is the same as the name used in the `--test-<test-name>` option. The `<filename>` is the name of the input file, but with any special characters replaced so that it always fits on one line.

Here is an example:

```
Hardened: FAIL: test: pie file: a.out.
```

```
--disable-colour
--enable-colour
--disable-color
--enable-color
```

Do not use colour to enhance FAIL, MAYB and WARN messages. By default annocheck will add colour to these messages so that they stand out when displayed by a terminal emulator. This option can be used in order to turn this feature off. The feature can be re-enabled with `--enable-colour`. The American spelling of color is also supported.

```
--full-filenames
--base-filenames
```

Use the full pathname for files. Useful when recursing into directories. By default this feature is disabled in normal mode and enabled in `verbose` mode. This option and its inverse `--base-filenames` can be used to set a fixed choice.

4.2.5 How to waive the results of annocheck tests

Now that `annocheck` is being used by the builders for Fedora and RHEL packages it is possible that certain tests may need to be waived for certain packages. This can be done on a per-package basis by editing the contents of the `rpminspect.yaml` file and adding an entry like this:

```
annocheck:
  - hardened: --skip-property-note --ignore-unknown --verbose
```

This example shows how the *property note* test can be ignored. Note that doing this overrides the default options that are passed to `annocheck` by the `rpminspect` framework, which is why the `--ignore-unknown` and `--verbose` options are also included in the example.

4.3 The annobin note displayer

```
annobin
  [-disable-hardened]
  -enable-notes
  file...
```

The `notes` tool displays the contents of any annobin notes inside the specified files. It groups the notes by address range, which can help locate missing details.

The `notes` tool is disabled by default, but it can be enabled by the command line option `--enable-notes`. Since the hardening checker is enabled by default it may also be useful to add the `--disable-hardened` option to the command line.

4.4 The section size recorder

```

annocheck
  [-disable-hardened]
  [-size-sec=name]
  [-size-sec-flags=!WAX]
  [-size-seg-flags=!WRX]
  [-size-human]
  file...

```

The *section-size* tool records the size of named sections within a list of files and then reports the accumulated size at the end. Since it is part of the **annocheck** framework, it is able to handle directories and rpms files as well as ordinary binary files.

The **--size-sec=*name*** option enables the tool and tells it to record the size of section *name*. The option can be repeated multiple times to record the sizes of multiple sections. It may also be useful to add the **--disable-hardened** option to the command line as otherwise the security hardening will be run at the same time.

Instead of searching for named sections, it is also possible to search for sections with specific flags. The **--size-sec-flags=<flags>** option will search for any section that has all of the specified <flags> set. Currently only *W*, *A* and *X* are recognised as flags, indicating that the section must have the *Write*, *Alloc* or *Execute* flags set respectively. If the *!* exclamation mark character is present then it negates the meaning of the following flags. Thus **--section-sec-flags=W** option will search for any writeable section whereas the **--size-sec-flags=W!A** option will search only for sections that are writeable but not allocated.

Instead of searching for sections by flags it is also possible to search for segments by flags using the **--size-seg-flags=<flags>** option. The flags recognised for segments are *W* for writeable, *R* for readable and *X* for executable. Again the *!* character can be used to invert the meaning of the flags that follow it.

If the **--verbose** option is enabled, then the tool will also report the size of the named section(s) in each file it encounters. If the **--size-human** option is enabled then sizes will be rounded down to the nearest byte, kibibyte, mebibyte or gibibyte, as appropriate.

4.5 How long did the check take ?

```

annocheck
  -enable-timing
  file...
  [-sec]
  [-usec]
  [-nsec]

```

The *timing* tool reports on the time taken by other tools to scan the list of files. The tool is disabled by default, but it can be enabled by the command line option `--enable-timing`.

By default the tool will report times in microseconds, but you can change this to reporting in seconds with the `--sec` or in nanoseconds with the `--nsec`. The default can be restored with the `--usec` option.

5 Configuring annobin and annocheck

When building annobin and annocheck from the sources there are a few configure options available to customise the build:

`--with-debuginfod`

`debuginfod` is a web service that indexes ELF/DWARF debugging resources by build-id and serves them over HTTP.

By default the `annocheck` program will be built and linked with the `debuginfod` client library `libdebuginfod` if it is present at build time. The `--with-debuginfod` configure option can be used to force the linking against the library even if the runtime `debuginfod` program cannot be found. Alternatively the `--without-debuginfod` can be used to force annobin to be built without `libdebuginfod` support, even if it is present on the build system.

`debuginfod` is packaged with `elfutils`, starting with version 0.178. You can get the latest version from ['https://sourceware.org/elfutils/'](https://sourceware.org/elfutils/).

`--with-gmp=PATH`

The `--with-gmp=PATH` option can be used to specify an alternative path to the gmp libraries, if necessary.

`--without-libelf`

The `annocheck` program uses `libelf` to read ELF binaries. By default the configure system will detect if the library is installed and if not, then it will disable the building of `annocheck` and the running of the tests. (Since they use `annocheck`). This behaviour can be overridden by the `--without-libelf` option which forces the build to assume that `libelf` is absent even if it would normally be detected.

`--without-tests`

Disable running the testsuite after building the various binaries.

`--with-clang`

Enable the building of the annobin plugin for the Clang compiler.

`--with-llvm`

Enable the building of the annobin plugin for the LLVM compiler backend. This is separate from the Clang plugin and can be used with any language that uses LLVM as a backend compiler.

`--without-gcc-plugin`

Do not build the gcc plugin.

`--without-docs`

Do not build the documentation.

--enable-maintainer-mode

This enables the regeneration of the `Makefile` and `configure` files when building the `annobin` sources.

6 How to use the information stored in the binary.

The `annobin` package includes some example scripts that demonstrate how the binary information can be used.

NOTE: These scripts are now redundant, their functionality having been subsumed into the `annockeck` program. However they are still useful as examples of how the `annobin` data can be consumed, so they are still included in the `annobin` sources.

The scripts are:

6.1 The `built-by` script

```
built-by
  [-help]
  [-version]
  [-verbose]
  [-quiet]
  [-silent]
  [-ignore]
  [-readelf=path]
  [-tmpdir=dir]
  [-tool=name]
  [-nottool=name]
  [-before=date]
  [-after=date]
  [-minver=version]
  [-maxver=version]
  [-]
  file...
```

The `built-by` script reports the name and version of the tool used to build the specified file(s). This script also demonstrates how information can be extracted from other other locations in the file, not just the binary annotation notes.

The script can also be used to filter files, only reporting those built by a specific tool, or a specific version of a tool, or even by a version of a tool that was built between a range of dates.

The options available are:

```
'--help'
'-h'      Displays the usage of the script and then exits.

'--version'
'-v'      Displays the version of the script.

'--verbose'
'-V'      Enables verbose mode, causing the script to detail each action
it takes.
```

- '--quiet'
'-q' Do not include the name of script in the out generated by the script.
- '--silent'
'-s' Produce no output. Just return an exit status.
- '--ignore'
Do not report file types that do not contain any builder information.
- '--tool=*name*'
Only report binaries built by *name*. The *name* is only an ordinary string, not a regular expression.
- '--nottool=*name*'
Skip any binary build by *name*. The *name* is only an ordinary string, not a regular expression.
- '--before=*date*'
Only report binaries built by a tool that was created before *date*. *date* has the format *YYYYMMDD*.
- '--after=*date*'
Only report binaries built by a tool that was created after *date*. When combined with the `--before` option can be used to restrict output to files which were built by tools created in a specific date range.
- '--minver=*version*'
Only report binaries built by a tool whose version is *version* or higher. The *version* string should be in the form *V.V.V*, for example *6.2.1*.
- '--maxver=*version*'
Only report binaries built by a tool whose version is *version* or lower. Can be combined with the `--minver` option to restrict output to those binaries created by tools within a specific version range.
- '--tmpdir=*dir*'
'-t=*dir*' Directory to use to store temporary files.
- '--readelf=*path*'
'-r=*path*' Use the specified program to read the notes from the files.
- '--'
Stop accumulating command line options. This allows the script to be run on files whose names starts with a dash.

6.2 The check-abi script

```

check-abi
  [-help]
  [-version]
  [-verbose]
  [-quiet]
  [-silent]
  [-inconsistencies]
  [-ignore-unknown]
  [-ignore-ABI|enum|FORTIFY|stack-prot]
  [-readelf=path]
  [-tmpdir=dir]
  [-]
  file...

```

The `check-abi` script reports any potential ABI conflicts in the files specified. This includes the use of the `-fshort-enums` option, the `-fstack-protector` option and the `-D_FORTIFY_SOURCE` option. All of these can affect passing data between functions and hence should be used uniformly throughout the binary.

The script accepts the following command line options:

```

--help
-h          Displays the usage of the script and then exits.

--version
-v          Displays the version of the script.

--verbose
-V          Enables verbose mode, causing the script to detail each action
           it takes.

--quiet
-q          Do not include the name of script in the out generated by the
           script.

--silent
-s          Produce no output. Just return an exit status.

--inconsitencies
-i          Only report files with potential ABI problems.

--ignore-unknown
           Do not report file types that are not supported or recognised.

--ignore-ABI|enum|FORTIFY|stack-prot
           Disables individual ABI checks. Multiple occurrences of this op-
           tion accumulate. Possible option values are:
           ‘ABI’      Disable checks of the general ABI information.
           ‘enum’    Disable checks of the -fshort-enum option.
           ‘FORTIFY’  Disable checks of the ‘-D_FORTIFY_SOURCE’ option.

```

`'stack-prot'`

Disable checks of the `-fstack-protect` option.

- `--tmpdir=dir`
- `-t=dir` Directory to use to store temporary files.
- `--readelf=path`
- `-r=path` Use the specified program to read the notes from the files.
- `--` Stop accumulating command line options. This allows the script to be run on files whose names starts with a dash.

6.3 The hardened script

```
hardened
  [-help]
  [-version]
  [-verbose]
  [-quiet]
  [-ignore-unknown]
  [-silent]
  [-vulnerable]
  [-not-hardened]
  [-all]
  [-file-type=auto|lib|exec|obj]
  [-skip=opt|stack|fort|now|relro|pic|operator|clash|cf|cet|realign]
  [-readelf=path]
  [-tmpdir=dir]
  [-]
  file...
```

The `hardened` script reports on the hardening status of the specified file(s). In particular it checks that the whole file was compiled with `-O2` or higher and the `-fstack-protector-strong`, `-D_FORTIFY_SOURCE=2`, `-Wl,-z,now`, `-Wl,-z,relro`, `-fPIE`, `-Wp,-D_GLIBCXX_ASSERTIONS`, `-fstack-clash-protection` `-fcf-protection=full` and `-mcet` options.

The script accepts the following command line options:

- `--help`
- `-h` Displays the usage of the script and then exits.
- `--version`
- `-v` Displays the version of the script.
- `--verbose`
- `-V` Enables verbose mode, causing the script to detail each action it takes.
- `--quiet`
- `-q` Do not include the name of script in the out generated by the script.
- `--ignore-unknown`
- `-i` Do not report file types that are not supported or recognised.

```

--tmpdir=dir
-t=dir      Directory to use to store temporary files.

--silent
-s         Produce no output. Just return an exit status.

--vulnerable
-u         Only report files that are known to be vulnerable. It records all of the necessary information about how they were built, but which were built with an incorrect set of options.
           This option is the default behaviour of the script.

--not-hardened
-n         Report any file that cannot be proven to be hardened. This is like the --vulnerable option, except that it will also report files that do not record all of the necessary information.

--all
-a         Report the hardening status of all of the files examined.

--file-type=auto|lib|exec|obj
-f=auto|lib|exec|obj
           Specifies the type of file being examined. Possible values are:
           'auto'      Automatically determine the file type from its extension. This is the default.
           'lib'       Assume all files are shared libraries. Checks that the -fPIC option was used.
           'exec'     Assume all files are executables. Checks that the -fPIE option was used.
           'obj'      Assume all files are object files. Skips checks of the bind now status.

--skip=opt|stack|fort|now|relro|pic|operator|clash|cf|cet
-k=opt|stack|fort|now|relro|pic|operator|clash|cf|cet
           Disables checks of various different hardening features. This option can be repeated multiple times, and the values accumulate. Possible values are:
           'opt'      Disables checks of the optimization level used.
           'stack'    Disables checks of the stack protection level.
           'fort'     Disables checks for -D_FORTIFY_SOURCE.
           'now'     Disables checks for 'BIND NOW' status.
           'relro'    Disables checks for 'relro' or read-only-relocs.
           'pic'     Disables checks for -fPIC/-fPIE.

```

‘operator’ Disables checks for ‘-D_GLIBCXX_ASSERTIONS’.
 ‘clash’ Disables checks for stack clash protection.
 ‘cf’ Disables checks for control flow protection. Note - these checks are only run on x86_64 binaries.
 ‘cet’ Disables checks for control flow enforcement. Note - these checks are only run on x86_64 binaries.
 ‘realign’ Disable checks for stack realignment. Note - these checks are only run on i686 binaries.

--readelf=path
-r=path Use the specified program to read the notes from the files.
-- Stop accumulating command line options. This allows the script to be run on files whose names starts with a dash.

6.4 The run-on-binaries-in script

```

run-on-binaries-in
  [-help]
  [-version]
  [-verbose]
  [-quiet]
  [-ignore]
  [-prefix='text']
  [-tmpdir=dir]
  [-files-from=file]
  [-skip-list=file]
  [-]
  program
  [program-options]
  file...
  
```

The `run-on-binaries-in` script allows other scripts, or programs, to be run on the executable files contained inside archives. This includes ‘rpm’ files, ‘tar’ and ‘ar’ files and compressed files.

The script does not recurse into directories, but this can be handled by the `find` command, like this:

```
find . -type f -exec run-on-binaries-in <script-to-run> {} \;
```

The script accepts the following command line options:

--help
-h Displays the usage of the script and then exits.
--version
-v Displays the version of the script.
--verbose
-V Enables verbose mode, causing the script to detail each action it takes.

If this option is repeated it has the special effect of cancelling out the automatic addition of the `-i` to recursive invocations of the script.

- `--quiet`
- `-q` Do not include the name of script in the out generated by the script.
- `--ignore`
- `-i` Do not report file types that are not supported or recognised.
This option is automatically enabled when the script is recursively invoked on an archive, unless the `-V -V` has been enabled. This is because it is assumed that archives are likely to contain files that do not need to be scanned.
- `--prefix='text'`
- `-p='text'` Add this text to the output from the script when it runs the program on a normal executable.
- `--tmpdir=dir`
- `-t=dir` Directory to use to store temporary files.
- `--files-from=file`
- `-f=file` Specifies a file containing a list of other files to examine, one per line.
- `--skip-list=file`
- `-s=file` Specifies a file containing a list of files not to examine, one per line. Blank lines and comments are ignored. Text after a file's name is also ignored. Filenames should start at the beginning of a line.
- `--` Stops processing of command line options. This allows the script to be run with a program whoes name starts with a dash.

Appendix A GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.

<http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or

to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.

- I. Preserve the section Entitled “History”, Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled “History” in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the “History” section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled “Acknowledgements” or “Dedications”, Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled “Endorsements”. Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled “Endorsements” or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not

add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements.”

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new

versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ‘‘GNU
Free Documentation License’’.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

