# eyoung许可声明

eyoung遵循GNU GENERAL PUBLIC LICENSE, V2开源许可证。

# 如何获取eyoung源代码

eyoung被托管在github.com上，源代码路径是：

*https://github.com/eyoung-father/eyoung.git*

可以使用svn或git进行代码的checkout：

*svn checkout https://github.com/eyoung-father/eyoung.git eyoung*

*git clone https://github.com/eyoung-father/eyoung.git*

目前最新的开发分支是EYOUNG\_R1，路径是：

*https://github.com/eyoung-father/eyoung.git/branches/EYOUNG\_R1*

# 如何编译eyoung源代码

编译eyoung源代码分一下几步：

1. 安装预依赖的相关工具，相关工具都放在源代码的tool目录下
   1. 安装m4-1.4.16，依次执行：

*tar xzf m4-1.4.16.tar.gz*

*cd m4-1.4.16*

*./configure*

*make*

*make install*

* 1. 安装bison-2.7，依次执行：

*tar xf bison-2.7.tar.gz*

*cd bison-2.7*

***patch data/yacc.c ../yacc.c.diff***

*./configure*

*make*

*make install*

* 1. 安装flex-2.5.37，依次执行：

*tar xf flex-2.5.37.tar.bz2*

*cd flex-2.5.37*

*./configure*

*make*

*make install*

* 1. 安装libelf-0.8.9，依次执行：

*cd libelf-0.8.9.tar*

*./configure*

*make*

*make install*

1. 编译准备，执行命令：

*make prepare*

1. 编译代码，执行命令：

由于eyoung内置了JIT编译器，所以其对处理器有一定依赖，目前eyoung支持Intel i386、Intel x86\_64和ARM v7三种体系架构。

对于Intel i386体系，编译代码需要执行

*make ARCH=i386*

对于ARM v7体系，编译代码需要执行

*make ARCH=arm*

对于Intel x86\_64体系，编译代码需要执行

*make ARCH=x86-64* 或省略ARCH参数，直接执行

*make*

1. 编译相关demo，执行命令：

*cd libdecoder/http/testsuite/*

*make*

1. 拷贝第三步中生成的目标库到环境变量LD\_LIBRARY\_PATH的路径：

*cp build/lib/\*.so [ld-path]*

# 如何执行eyoung示例

在libdecoder/http/testsuite目录下执行make，并执行以下shell命令：

*./http\_xss http\_xss.ey case/req-15.msg*

在这里：

* http\_xss是编译好的可执行程序，其对应的源文件是demo\_http\_xss.c。可以查看该文件，初步了解eyoung api。eyoung api的详细说明，在《Programming Guide》中进行详细说明。
* http\_xss.ey是针对HTTP协议的规则框架，其中包含了一条XSS检测的规则。eyoung规则的写法，在《Signatures Specification》中进行详细说明。
* req-15.msg模拟了一个http post请求，其中包含了一个XSS漏洞的利用。该文件由demo\_http\_xss.c中的测试驱动程序进行解析，解析结果被送到libdecoder/http/decode下的http协议分析模块进行解析。协议解析的结果被http协议分析模块提交给libengine下的分析引擎进行事件分析和攻击检测。

此外libdecoder/html/testsuite、libdecoder/pop3/testsuite下也包含很多用于DEMO的测试用例。