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

a) 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
```

b) 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
```

c) 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
```

d) 安装 libelf-0.8.9，依次执行：

```
cd libelf-0.8.9.tar
./configure
make
make install
```

2， 编译准备，执行命令：

```
make prepare
```

3， 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
```

4， 编译相关 demo，执行命令：

```
cd libdecoder/http/testsuite/
make
```

5， 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。