# CSTE v1.0 设计文档

### CSTE简介

CSTE1.0是一个集成了QEMU、BASP等分析工具以及配有大量例子的综合性计算机运行时安全测评框架。首先CSTE1.0能够验证各种防御机制的有效性，其次CSTE1.0还能评测计算机当前运行环境的安全性。

CSTE1.0包含各种漏洞程序的实现，并给出了针对各种漏洞程序的攻击方式。其中漏洞程序使用 C 语言编写，攻击程序以 Python (Python 2.7) 脚本的形式给出。攻击脚本的执行依赖 cle 库。

CSTE1.0的开发语言为当前非常流行的Python，并采用了目前生态比较好的Python2.7。

CSTE1.0能够给出QEMU分析报告、漏洞程序执行结果、攻击结果、编译结果等。并能给出攻击成功个数、攻击失败个数、攻击成功比例等统计性结果。CSTE1.0支持粗粒度统一配置，也支持个性化配置每个样例。CSTE1.0支持统一执行所有样例，也支持单独运行某个样例。CSTE1.0支持对漏洞程序的执行过程进行分析，也支持仅运行漏洞程序，进行粗略的统计，而不使用QEMU、BASP等进行分析。同时CSTE提供了一个较人性化的命令行界面。支持几个比较重要的Linux原生命令，方便对配置文件进行修改。也提供了几条个性化的命令，来操作CSTE。CSTE1.0尽量简化了添加样例的复杂性，使得添加样例变得尽可能的简洁。同时CSTE1.0也能较好的降低使用该框架进行学习的人的学习成本。

### CSTE基本执行流程

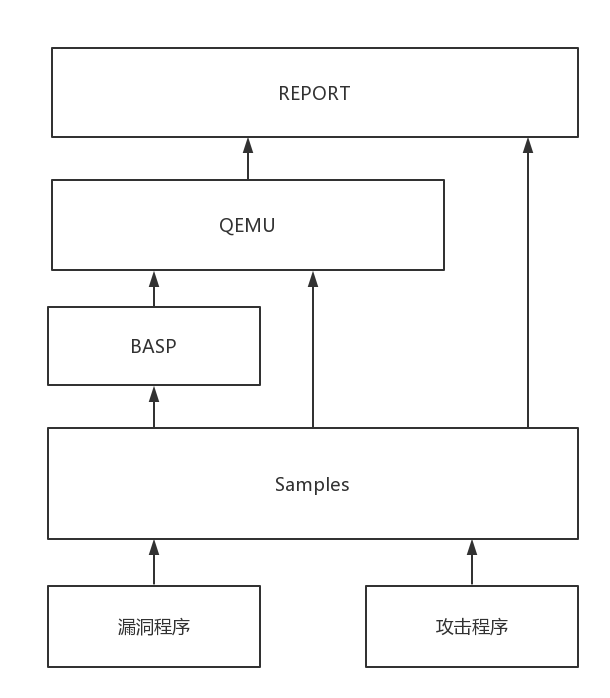
CSTE1.0基本的执行流程为自动编译漏洞程序得到可执行文件，调用BASP分析工具，得到静态分析结果，依赖BASP分析工具的分析结果，开启QEMU对漏洞程序的执行过程进行监控。调用攻击脚本对漏洞程序进行攻击。得到QEMU的分析结果，漏洞程序执行结果，攻击结果等。其中在QEMU中有各种防御机制的实现，可以很好的验证防御机制能否成功的进行防御。下面给出比较进行分析工作的执行流程图。



（图1：分析过程执行流程图）

### 设计框图

在这里给出CSTE1.0的基本框图。其中Samples为CSTE1.0框架的基础。在其之上集成了BASP分析工具、QEMU分析工具等，其中所有工具的分析过程都要依赖样例集，即Samples，分析工具会产生相应的报告，漏洞程序会输出运行结果，攻击程序会输出攻击结果，所有的这些结果，CSTE1.0都以报告的形式给出。其中图中也给出了清楚的依赖情况，首先是Sample主要由漏洞程序和攻击程序组成，所有分析的分析过程都依赖Samples，其中QEMU的部分功能依赖于BASP的输出结果，而QEMU还有一部分功能的实现不依赖于BASP的分析结果。同时Samples也可以单独执行，不进行任何分析工作。所有的结果都会生成报告，最后会汇总所有样例的执行结果。得到一些统计量，对防御机制的防御效果进行量化。



（图2：设计框图）

### 操作指南

在这里不详述CSTE1.0的使用方法，在CSTE1.0的根目录下有详细的README.md文档。在README.md文档里有对CSTE1.0各项配置参数的详细说明，还有对各条命令的详细说明，另外在README.md里给出了一个QUICK START教程，能够指导新用户快速的上手使用CSTE1.0。同时在README.md里还给出了添加样例的一些简要的要求。以及对各个已有样例的详细介绍。

### 工作展望：

CSTE1.0到此已经告一段落，日后会在CSTE1.0的基础之上实现功能更加完善、操作更加便捷、界面更加友好的CSTE2.0版本等。敬请期待！