**开源软件中安全漏洞自动检测与定位**

（ICSOFT—智能计算与软件中心——倪超）

* 1. 背景

开源软件极大地推动了信息技术的发展，提升生产效率和生活质量。然而，软件漏洞(Software Vulnerability)严重影响了软件质量，破坏了软件的功能性，甚至在一些场景下会造成严重的生命和财产损失。软件漏洞检测技术可以在软件正式发布之前发现软件中存在的安全缺陷，推荐给开发人员，验证漏洞，解释漏洞，修复漏洞，从而提高软件的质量。近年来，围绕软件安全漏洞的研究工作主要分为两大方向：基于程序分析的漏洞检测和基于机器学习的漏洞检测。两类方法各有优缺点，基于程序的分析的方法具有正确性高但误报率也高特性，而基于机器学习的方法具有扩展性强但是精确性不高的特点。因此，将两种方法进行有效的结合，将会进一步提升软件漏洞检测工具的能力。

此外，今年来基于版本控制系统的开发方式得到了企业界和学术界的认可。研究人员发现，基于版本控制系统的软件漏洞检测可以在代码提交的时刻发现代码中引入的漏洞，并具有如下三个好处：

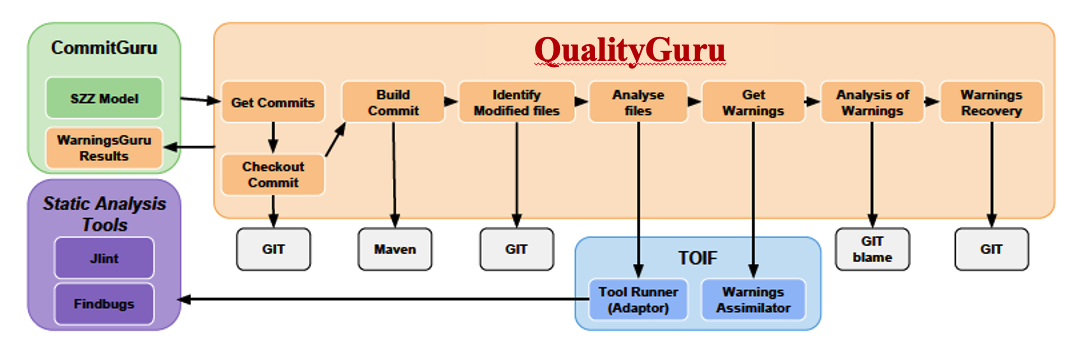
* 更细的预测粒度：被识别出来的、含有漏洞的代码修改会被连接到确定的代码修改行，这极大地缩小了需要检查的代码区域；
* 更准确的开发人员：被识别出来的、含有漏洞的代码修改，通过版本控制系统可以找到对应代码的开发者；
* 更及时的发现：在代码提交的时间点进行代码分析，判断是否可能引入漏洞。

研究人员提出了多种先进的方法，包括有监督的（e.g., CBS+ [1])，无监督的 (e.g., Churn [2])，基于机器学习的 (e.g., OneWay [3])和基于深度学习的模型(e.g., DeepJIT [4])。这些方法从不同的角度对即时软件漏洞检测进行了探索，并取得了一定的成果。

* 1. 功能与目标

本任务需要构建一个机器学习和静态程序分析结合的系统，该系统能够识别相应的代码修改是否会引入安全漏洞，并且给出漏洞的解释如漏洞的类型，在给定的数据集合上完成模型的评估,并且展示出漏洞的位置。具体要求如下：

1. 基于深度学习的分类模型完成有无漏洞的判断，需要熟悉相应模型的原理，可以参考BERT模型及其变种模型；
2. 使用静态分析工具，完成程序的分析和漏洞分析
3. 使用TOIF工具，综合静态分析工具的结果，并整合输出
4. 搭建Web展示系统
   1. 技术路线



关键技术：程序分析，细粒度分类，版本控制系统GIT, 深度学习等

技术难点：

1. 细粒度代码修改分类。如何从代码变更中判断修改是否可能引入安全漏洞，现有的方法是从特征工程的角度研究的，如何从自然语言处理或者从程序代码的角度进行判断一次代码修改是否会引入缺陷。
2. 如何将程序分析结果和深度学习检测的结果进行综合，筛选出最有可能还有漏洞的代码变更。
3. 如何打通分析工具与漏洞库之间的联系，从而对有问题的代码进行详细的漏洞类型划分
   1. 开发环境

操作系统: Linux, Windows

开发软件: Git, python, pytorch等深度学习平台

参考文献:

[1]. Huang Q, Xia X, Lo D. Revisiting supervised and unsupervised models for effort-aware just-in-time defect prediction[J]. Empirical Software Engineering, 2019, 24(5): 2823-2862.

[2]. Liu J, Zhou Y, Yang Y, et al. Code churn: A neglected metric in effort-aware just-in-time defect prediction[C]// 2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM). IEEE, 2017: 11-19.

[3]. Fu W, Menzies T. Revisiting unsupervised learning for defect prediction[C]//Proceedings of the 2017 11th joint meeting on foundations of software engineering. 2017: 72-83.

[4]. Hoang T, Dam H K, Kamei Y, et al. DeepJIT: an end-to-end deep learning framework for just-in-time defect prediction[C]//2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR). IEEE, 2019: 34-45.