Review 15

3/20/12

CS6V81.502

Seungtack Baek

**AMNESIA: Analysis and Monitoring for**

**NEutralizing SQLInjection Attacks**

In this Paper, Halfond and Orso realized growth of web application and, as a result, need for a protection against SQL injection attacks (SQLIA). To detect SQLIA, they used model-based approach. First, they analyzed code and built the model for legitimate queries. Now, at runtime, each query from user will be checked against the model and if queries from users fail to matches with their model, this query will be classified as SQLIA.

They took this idea and actually built the tool, named AMNESIA then evaluated according to the following three research questions:

1. Does it have any “leak”? (Any false negative?)
2. How heavy is it? (Any overhead?)
3. Does it interfere with the application’s functionality? (Any false positives?)

To answer the above three questions, they took 7 web applications, both commercial and non-commercial, and tested over these applications. While generating their samples, they had their “expert” colleague on software-security generate sample SQLIA’s. (Which makes me to question, “how do you know if he is “expert”? And, although generating samples are not major, but for this study, generating sample SQLIA is significant contribution. Why is he not on the authors’ list??”) Then they measured how each query was classified, either as LEGIT or ATTACK.

They claim that the result is promising. They had 100% detection rate, no false negative and no false positive. However, good grade never tells you how well you are doing. But bad grade tells you how bad you are doing. We cannot conclude that this tool will detect 100% of all SQLIA’s. According to section 5.2, they said that they can have false negatives in two situations: where SQL is too conservative, or when attack query happens to have same SQL structure as legit query. They mentioned that both cases are rare. (But, are not the rare cases the ones we should watch for?)

They conclude that this tool is “so far promising” (which I do agree to certain extent). Yet, they did not include the performance of static analysis part, such as “how long it takes to analyze the code and apply instrumentation?” In the future studies I wish to see more analysis on static portion and its application to other types of web-application.

**Question:**

1. In their sampling method, for each parameter, they used a random number from 1-60. Why do you think that they chose 60 as their cap, as opposed to 1-100 (which is typical values)? Would it be somehow related to time?