Sammanfattning

Det här examensarbetet är baserat på idéer ur ett uppdrag från företaget Saab Aerotech men är ett eget arbete.

Målet var att undersöka om det finns behov av ett verktyg som statiskt kan detektera dynamiska minneshanteringsproblem, som till exempel minnesläckage, i applikationer skrivna i C/C++. På grund av att minneshanteringsfel i C/C++ länge har varit ett känt problem undersökte vi detta och de befintliga lösningarna till det.

Vi fann två metoder till lösningar som de flesta verktyg använde sig av; statisk och dynamisk detektering. De flesta verktyg löste problemet genom att dynamiskt detektera minnesläckor och andra brister som till exempel buffer overflows. Ett verktyg löste dock problemet genom att statisk detektera minneshanteringsfel i källkoden för applikationerna. Eftersom alla befintliga lösningar har någon form av ineffektivitet så har vi undersökt möjligheten att utveckla ett mer effektivt verktyg. Vi har kommit fram till att denna möjlighet finns men det kräver enormt mycket tid och arbete att göra ett komplett verktyg som detekterar minneshanteringsfel statiskt.

Vår prototyp detekterar dynamiska minneshanteringsproblem i källkoden statiskt. Vi har använt oss av hjälpverktygen Flex och Bison för att utveckla vår prototyp av verktyget. Prototypen kan analysera källkod skriven i programspråken C och C++ och klarar att detektera minnesläckage, felaktiga avallokeringar av minne, dangling pointers, samt läsning från och skrivning till ogiltiga minnesområden. På grund av tidsbrist har vi i nuläget inte implementerat något stöd för klasser och objekt i prototypen.

Abstract

This bachelor's project is our own project, but it is based on ideas from an assignment from the Saab Aerotech company.

The goal was to investigate if there is a need for a tool that statically can detect dynamic memory management errors, such as memory leaks, in applications written in C/C++. Since the problem of memory management errors in the C/C++ languages has been known for a long time, we decided to investigate this and the existing solutions.

We found that most tools used two methods as solutions; static and dynamic detection. Most of these tools solve the problem by dynamically detecting memory leaks and other deficiencies such as buffer overflows. However, one of these tools used static detection of these deficiencies by scanning the source code of the applications. Since all the existing solutions have some kind of inefficiency, we have investigated the possibility to develop a more efficient tool. We concluded that this is possible but it will take a lot of time and effort to implement a complete tool that statically detects memory management errors.

Our prototype statically detects dynamic memory management problems in the source code. We have used the tools Flex and Bison to develop our prototype of a static detection tool. The prototype analyzes source code written in the programming languages C and C++ and is capable of detecting memory leaks, invalid deallocations of memory, dangling pointers and reading from and writing to invalid memory areas. Currently, due to lack of time, we have not implemented any support for classes and objects in the prototype.