**PROGRAMMING SECURE SOFTWARE SYSTEMS – ASSIGNMENT 2**

Sam Reid – 10284311

Alex Till – 10888888

20/05/2016

Michael N. Johnstone

Table of Contents

[Introduction i](#_Toc451011831)

[Pruning Algorithm #1 ii](#_Toc451011832)

[Pruning Algorithm #2 ii](#_Toc451011833)

[Program Requirements ii](#_Toc451011834)

[Class Diagram iii](#_Toc451011835)

[References iii](#_Toc451011836)

# INTRODUCTION

An integral part of planning a software system is defining the functional requirements. Before development begins documentation is required that clearly portrays what the system is meant to do and how a user interacts with it. Use cases encompass the requirements of a system and map the order of events that enable the system to achieve its goal (Bittner, 2002). The Unified Modelling Language (UML) is used to create models that visually display aspects of a system. UML is considered a standard modelling language used in systems development to identify and demonstrate the use cases of a software system (Quatrani & Evangelist, 2003). UML use case diagrams provide an effective method of modelling functional requirements of software systems while providing a simple representation of the systems functionalities for both the prospective user and developer (Shen & Liu, 2003).

While UML use cases are helpful for understanding the interactions between application users and the system, a need exists for a similar diagram that examines the threats and threat actors against the system. As the name suggests, misuse case diagrams are use case diagrams created from a malicious actor’s point of view; a misuser who create misuses. Misuse case diagrams interact with the use case diagram to demonstrate the security threats at each functional requirement (Alexander, 2003). They can be used as an effective means of modelling security requirements (Johnstone, 2011).

There exist many methodologies for creating misuses and identifying threats for a given actor and category of vulnerability. The ‘STRIDE’ methodology consists of a matrix which binds six areas of security; Spoofing identity, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege (Johnstone, 2010). Using the STRIDE matrix it is possible to enhance a UML with threat modelling capability by creating misuse cases for a given threat actor, threat type and use case (Johnstone, 2010). However, not all the misuses created will make practical sense and intelligence is needed to examine the candidate misuse cases and determine their validity.

///WHAT’S THE TASK///

By building decision tree’s and parsing candidate misuse cases through a pruning algorithm we hope to automate the validation task of creating misuse cases. This report examines popular pruning algorithms, explaining how they operate and their feasibility in relation to pruning candidate misuse cases.

~~The task is to create a program that intelligently creates valid misuse cases for a given use case. The program should accept a Use Case Diagram (UCD) in Extensible Markup Language (XML) format and create misuse cases by choosing the applicable STRIDE matrix~~

~~to generate many misuse cases for a given use case diagram we are required to investigate pruning strategies to be left with a smaller selection of viable misuse cases. There are many pruning algorithms and research is to be done to determine which one is the best to implement to achieve our projects goals.~~

# PRUNING ALGORITHM #1

# PRUNING ALGORITHM #2

# PROGRAM REQUIREMENTS

As a proof of concept, the program needs only a few key requirements:

* Read use case diagram in xml format
* Generate candidate misuse cases using STRIDE matrix
* Prune candidate misuse cases to remove invalid cases
* Print valid misuse cases to the console

# CLASS DIAGRAM

# REFERENCES

Alexander, I. (2003). Misuse cases: Use cases with hostile intent. *Software, IEEE, 20*(1), 58-66.

Bittner, K. (2002). *Use case modeling*: Addison-Wesley Longman Publishing Co., Inc.

Johnstone, M. N. (2010). Threat modelling with STRIDE and UML.

Johnstone, M. N. (2011). Modelling misuse cases as a means of capturing security requirements.

Quatrani, T., & Evangelist, U. (2003). Introduction to the Unified modeling language. *A technical discussion of UML, 6*(11), 03.

Shen, W., & Liu, S. (2003). Formalization, testing and execution of a use case diagram *Formal Methods and Software Engineering* (pp. 68-85): Springer.