Detekcia malwaru pomocou hlbokého učenia založeného na vízii*

Jakub Martinak
Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
xmartinakj@stuba.sk

11. oktober 2021

Abstrakt

Keďže v 21. storočí je čím ďalej tím viac potreba chrániť naše elektronické zariadenia pred nebezpečným malwarom tak sa chcem zamerať na túto tému v oblasti informačnej bezpečnosti a implementovanie strojového učenia do nej. V tomto článku na začiatku objasním modelovanie v softwarovom inžinierstve a ako to súvisí s mojou témou a neskôr sa budem zameriavať na presnosť určenia či sa jedná o malware alebo nie pomocou Deep Learning Vision a taktiež budem rozoberať problémy a výhody spojené s touto formou detekcie, výhody a nevýhody klasickej detekcie malwaru a jeho limity oproti strojovému učeniu.

1 Úvod

Túto tému som si vybral z dôvodu nárastu kybernetických útokov vo svete a následnej obrane voči nim pomocou moderných technológií a využitím umelej inteligencie. V prvej časti článku sa budem venovať modelovanie v softwarovom inžinierstve 2. Druhá časť sa bude zaoberať **??**, z ktorej následne prejdem na hlavnú tému tohto článku **??**. Záverečné poznámky prináša časť 6.

2 Modelovanie v Softwarovom inžinierstve

Z obr. **??** je všetko jasné.

3 Iná časť

Základným problémom je teda... Najprv sa pozrieme na nejaké vysvetlenie (časť 3.1), a potom na ešte nejaké (časť 3.1).[1]

*Semestrálny projekt v predmete Metódy inžinierskej práce, ak. rok 2021/22, vedenie: Fedor Lehocki

[1] Niekedy môžete potrebovať aj poznámku pod čiarou.

Môže sa zdať, že problém vlastne nejestvuje [1], ale bolo dokázané, že to tak nie je [2,3]. Napriek tomu, aj dnes na webe narazíme na všelijaké pochybné názory [4]. Dôležité veci možno *zdôrazniť kurzívou*.

3.1 Nejaké vysvetlenie

Niekedy treba uviesť zoznam:

- jedna vec
- druhá vec
  - x
  - y

Ten istý zoznam, len číslovaný:

1. jedna vec
2. druhá vec
   (a) x
   (b) y

3.2 Ešte nejaké vysvetlenie

Veľmi dôležitá poznámka. Niekedy je potrebné nadpisom označiť odsek. Text pokračuje hneď za nadpisom.

4 Dôležitá časť

5 Ešte dôležitejšia časť

6 Záver

Literatúra

[1] James O. Coplien. *Multi-Paradigm Design for C++*. Addison-Wesley, 1999.

[2] Krzysztof Czarnecki, Simon Helsen, and Ulrich Eisenecker. Staged configuration through specialization and multi-level configuration of feature models. *Software Process: Improvement and Practice*, 10:143–169, April/June 2005.

[3] Krzysztof Czarnecki and Chang Hwan Peter Kim. Cardinality-based feature modeling and constraints: A progress report. In *International Workshop on Software Factories, OOPSLA 2005*, San Diego, USA, October 2005.

[4] Carnegie Mellon University Software Engineering Institute. A framework for software product line practice—version 5.0. `http://www.sei.cmu.edu/productlines/frame_report/`.