Перелік посилань

1. Janak, K., 2005. Differences in volume of round Timber caused by different determination methods Drvna industrija 56 (4) pp.165-170.

2. Knyaz V.A., 2002. Method for on-line calibration for automobile obstacle detection system, International Archives of `Photogrammetry and Remote Sensing, Proceedings of ISPRS Commission V Symposium "CLOSE-RANGE IMAGING, LONG-RANGE VISION", Vol. XXXIV, part 5, Commission V, September 2-6, Corfu, Greece. Pp. 48-53

3. Knyaz V.A Vizilter Yu.V., Zheltov S.Yu., 2004. Photogrammetric techniques for measurements in woodworking industry. International Archives of Photogrammetry and Remote Sensing, Vol. XXXIII, part B5/2, XXth ISPRS Congress, 12-23 July 2004, Istanbul, Turkey pp. 42-47

Метод комплексного тестування операційних систем реального часу

Ліщук Б.
Науковий керівник – к.т.н., доц. Кльоц Ю.П.
Хмельницький національний університет

Важливим напрямом дослідження для сучасних систем реального часу є вирішення проблеми тестування операційних систем реального времені. Для пошуку помилок на різних етапах життєвого циклу такого програмного забезпечення (ПЗ) даного класу пропонується застосувати методи фазінг-тестування. Для здійснення даного завдання досліджено сучасні підходи застосування фазінга і розглянуті особливості життєвого циклу бортових систем цивільної авіації. В роботі описуються як деталі реалізації окремих низькорівневих кроків, таких як процеси генерації тестових даних, отримання зворотного зв'язку і фіксації виникають аномалій, так і підходи високого рівня, такі як застосування методів тестування захищеності, побудова моделі загроз, пошук і пріоритетизація точок взаємодії з навколишнім середовищем, визначення пріоритетів ризику для цільової системи. Для практичної реалізації концепції використаний прототипів ARINC 653, для якого наводяться потенційні точки взаємодії і способи для їх фазінг-тестування. Здійснено приклад реалізації системи фаззінг-тестування для стадії інтеграції компонентів, в якій розглядається питання довіри до стороннього обладнання і поставленого виробником драйверу.

Сучасна операційна система реального часу (ОСВЧ) передставляє собою багатофункціональну операційну систему, призначену для безпечного розподілу ресурсів обчислювальних комплексів між функціональним програмним забезпеченням (ФПЗ), виконує готове завдання обчислювальної і логічної обробки даних в реальному часі. Висхідною вимогою до сучасних

ОСРЧ є забезпечення сумісності з стандартами ARINC 653 [1]. ARINC 653 (Avionics Application Standard Software Interface), за аналогією з іншими галузевими стандартами, такими як розширення POSIX для підтримки реального часу [2], OSEK-VDX [3] і µITRON [4], застосовується в медицині, станках з програмним управлінням, автомобільної промисловості та інших вбудованих системах, описує вимоги для ОС, що визначають організацію і управління безліччю завдань на заданому апаратному забезпеченні.

Стандарт ARINC 653 містить шість частин, деякі з них в свою чергу розбиваються на окремі документи. Наприклад, специфікація тестів сервісів на відповідність представлена в частинах A і B для обов'язкових додаткових сервісів відповідно. Повний список частин стандарту ARINC 653: огляд ARINC 653; обов'язкові сервіси; додаткові сервіси; специфікація тестів сервісів на відповідність; обмежені сервіси; рекомендовані можливості базового ПЗ.

Працююча операційна система з підтримкою ARINC 653 як кінцевий продукт представляє собою інтегрований модуль, який розбивається на базовый модуль и функціональне програмне забезпечення, яке в свою чергу розділене на підмодулі і компонети.

Склад ФПЗ, яке використовується, визначається завчасно і не підлягає зміні під час роботи системи. Пам'ять ФПЗ фізично розділена на спеціальні ізольовані секції, які називаються розділами: ARINC-сумісні і сторонні (системні). Кожен розділ представляє собою окрему програмну підсистему. З метою мінімізації взаємного впливу ФПЗ працюючого на одному обчислювальному модулі, розподіл ресурсів між ФПЗ також виконується завчасно та не змінюється. Розділи мають циклічно задані статично розписані завдання розкладу, гарантії часових ізоляцій якого забезпечуються операційною системою. Кожен розділ містить не менше одного процесу, що має такі атрибути як тривалість, період, пріоритет, поточний стан.

Запропоновані на початку дослідження гіпотези, такі як можливість виявлення максимальної кількості помилок на стадії прототипування і можливості розробки нових алгоритмів, здатних краще враховувати тимчасові характеристики фаззінг-тестування для операційних систем реального часу, знайшли часткове підтвердження в процесі вирішення задач. В ході роботи були запропоновані додаткові ідеї, деякі з яких було використано в розробленому прототипі: можливість проведення фазінг-тестування з урахуванням порушення обмежень за часом на неточних емуляторах за допомогою послідуючих запусків на апаратному забезпеченні; необхідність моделювання взаємовпливів шин і виділення потенційно залежних і незалежних пристроїв для збільшення продуктивності; використання напівавтоматичної генерації вбудованих перевірок на основі напівформальних вимог з супровідної документації з подальшим їх вбудовуванням в окремі компоненти ПЗ через засоби динамічного

інструментарію; здійснення аналізу ітерацій циклів в реалізації фазера для пошуку аномалій по порушенню строків завершення; кластеризації місць виникнення переривань від таймерів або пристроїв з урахуванням пріоритетів, здатних в більшій мірі впливати на поведінку системи.

Перелік посилань

1. ARINC Industry Activities. ARINC 653P1. Airlines Electronic Engineering Committee (AEEC). 2015-08-21. 285c.

2. IEEE Std 1003.1-2017. Portable Applications Standards Committee System Services Working Group. IEEE Standard for Information Technology Portable Operating System Interface (POSIX). Base Specifications, издание7. Введён 2017. 3951c.

3. ISO 17356-3:2005. ISO/TC 22/SC 31 Data communication. Road vehicles Open interface for embedded automotive applications Part 3: OSEK/VDX Operating System (OS). Введён 2005-11. 61c.

4. Ken Sakamura, TRON ASSOCIATION. μITRON4.0 Specification. Версия 4.03.00. Введён 2007. URL: https://www.tron.org/wp-content/themes/ dp-magjam/pdf/specifications/en_US/WG024-S001-04.03.00_en.pdf (дата звернення: 31.05.19).

Алгоритми побудови та функціонування нейромережевої штучної імунної системи для виявлення шкідливих програм

Мазурок М.В.
Науковий керівник: ктн. доц. Джулій В.М.
Хмельницький національний університет

Розглянемо процеси генерації, навчання, відбору та функціонування імунних детекторів на основі нейронних мереж. Генерується початкова популяція імунних детекторів, кожен з яких являє собою штучну нейронну мережу. Представимо нейромережевий імунний детектор у вигляді чорного ящика, який має n входів і два виходи (рисунок 1).

Вихідні значення детектора формуються після подачі всіх образів на нього відповідно до наступного виразу:

$$Z_1 = \begin{cases} 1, \textit{якщо чистий файл} \\ 0, \textit{інакше} \end{cases}$$

$$Z_2 = \begin{cases} 1, \textit{якщо вірус} \\ 0, \textit{інакше} \end{cases}$$