ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ ЖОҒАРЫ БІЛІМ МИНИСТРЛІГІ

«Л.Н. ГУМИЛЕВ АТЫНДАҒЫ ЕУРАЗИЯ ҰЛТТЫҚ УНИВЕРСИТЕТІ» КЕАҚ

Студенттер мен жас ғалымдардың
«ǴYLYM JÁNE BILIM - 2023»
XVIII Халықаралық ғылыми конференциясының
БАЯНДАМАЛАР ЖИНАҒЫ

СБОРНИК МАТЕРИАЛОВ
XVIII Международной научной конференции
студентов и молодых ученых
«ǴYLYM JÁNE BILIM - 2023»

PROCEEDINGS
of the XVIII International Scientific Conference
for students and young scholars
«ǴYLYM JÁNE BILIM - 2023»

2023
Астана

УДК 001+37
ББК 72+74
G99

«ǴYLYM JÁNE BILIM – 2023» студенттер мен жас ғалымдардың XVIII Халықаралық ғылыми конференциясы = XVIII Международная научная конференция студентов и молодых ученых «ǴYLYM JÁNE BILIM – 2023» = The XVIII International Scientific Conference for students and young scholars «ǴYLYM JÁNE BILIM – 2023». – Астана: – 6865 б. - қазақша, орысша, ағылшынша.

ISBN 978-601-337-871-8

Жинаққа студенттердің, магистранттардың, докоранттардың және жас ғалымдардың жаратылыстану-техникалық және гуманитарлық ғылымдардың өзекті мәселелері бойынша баяндамалары енгізілген.

The proceedings are the papers of students, undergraduates, doctoral students and young researchers on topical issues of natural and technical sciences and humanities.

В сборник вошли доклады студентов, магистрантов, докторантов и молодых ученых по актуальным вопросам естественно-технических и гуманитарных наук.

УДК 001+37
ББК 72+74

ISBN 978-601-337-871-8

6. Wolman A., Voelker G., Sharma N., Cardwell N., Karlin A., Levy H. On the scale and performance of cooperative Web proxy caching // Operating Systems Review 1999 -34(5) - p.16-31
7. Douglis E., Feldmann A., Krishnamurthy B., and Mogul J. Rate of change and other metric: a live study of the World Wide Web // In Proc. of the 1st USENIX Symp. on Internet Technologies and System – 1997. P. 147-158
8. Долгих Д.Г., Сухов А.М. Системы резервирования трафика. Эффект изменения документов в глобальной сети // Телекоммуникации, 2007. № 5 - с. 29-31

УДК 004.056

ФОРМАЛЬНЫЕ МЕТОДЫ ВЕРИФИКАЦИИ ПРОГРАММНЫХ СИСТЕМ НА ПРИМЕРЕ ЯЗЫКА FRAMA-C

Мұқанова Айкүн Сабитқызы
aikunmukanova@gmail.com
Магистрант 2 курса факультета «Информационные технологии» ЕНУ им.Л.Н.Гумилева, Астана, Казахстан
Научный руководитель – Сауханова Ж.С.

Введение. Программное обеспечение играет важную роль в современном мире, корректность и безопасность программного кода являются ключевыми аспектами, которые нужно учитывать при разработке программных систем. В связи с этим все большую популярность получают формальные методы верификации, которые позволяют доказывать корректность программного кода математически. Одним из инструментов для формальной верификации является язык программирования Frama-C, который предназначен для анализа и верификации программных систем на языке C. Frama-C позволяет проводить статический анализ программного кода и доказывать его корректность с помощью формальных методов. Целью данной статьи является рассмотрение принципов работы языка Frama-C и его применения для анализа и верификации программного кода на языке C. Будут рассмотрены основные концепции Frama-C, а также пример его применения для анализа корректности и безопасности программного кода.

Ключевые слова: формальные методы, верификация, программные системы, язык Frama-C, корректность ПО, безопасность.

Формальные методы верификации программных систем являются одним из важных направлений исследований в области информационных технологий. Они позволяют доказать корректность программного кода на основе формальных методов, что является необходимым условием для создания надежных и безопасных программных систем. Один из инструментов для формальной верификации программного кода - язык программирования Frama-C. Этот язык основан на языке C и используется для формальной верификации программных систем, написанных на языке C [1].

Frama-C содержит несколько модулей, каждый из которых предназначен для проверки определенного аспекта кода. Некоторые из этих модулей включают в себя:

- Value - модуль, предназначенный для анализа числовых значений переменных в программном коде. Он используется для поиска ошибок в коде, связанных с неправильным использованием переменных.
- WP (Weakest Precondition) - модуль, который позволяет автоматически генерировать условия для проверки корректности работы программного кода. Он используется для проверки корректности работы функций и алгоритмов.
- Jessie - модуль, который предназначен для проверки правильности использования указателей в программном коде. Он позволяет автоматически генерировать условия, которые проверяют правильность работы с указателями [2].

Для использования языка Frama-C необходимо создать специальный проект и загрузить в него исходный код программной системы, которую необходимо проверить. После этого можно использовать различные модули Frama-C для проверки корректности кода [3]. Процесс

верификации программы в Frama-C начинается с анализа ее исходного кода. Затем Frama-C преобразует код в формальный язык, который может быть использован для доказательства корректности кода. В процессе верификации Frama-C автоматически генерирует формулы, которые доказывают корректность кода [4]. Если верификация успешна, то Frama-C дает уведомление о том, что код корректен. Одним из примеров использования Frama-C является проверка корректности программного кода, который используется для управления двигателями воздушных судов. В этом случае Frama-C используется для проверки корректности кода, который контролирует работу двигателей в различных режимах полета. В результате верификации было установлено, что код корректен и соответствует требованиям безопасности и надежности. Еще один пример использования Frama-C - это проверка корректности кода, который используется в автоматических тестах для приложений. В этом случае Frama-C используется для проверки корректности работы тестового кода и обнаружения возможных ошибок в нем [5]. Стоит отметить, что верификация программного кода с помощью Frama-C может быть сложной и требовательной задачей. Это связано с тем, что верификация требует специальных знаний и навыков в области формальных методов верификации и языка программирования C. Тем не менее, использование формальных методов верификации программных систем является необходимым для создания надежных и безопасных программных систем. В этом смысле язык Frama-C является мощным инструментом для формальной верификации программного кода на языке C [6]. Ниже на рисунке 1 предоставлен пример программы, написанный на языке C (справа) и представленный на платформе Frama-C (посередине), которая вычисляет сумму элементов в матрице:

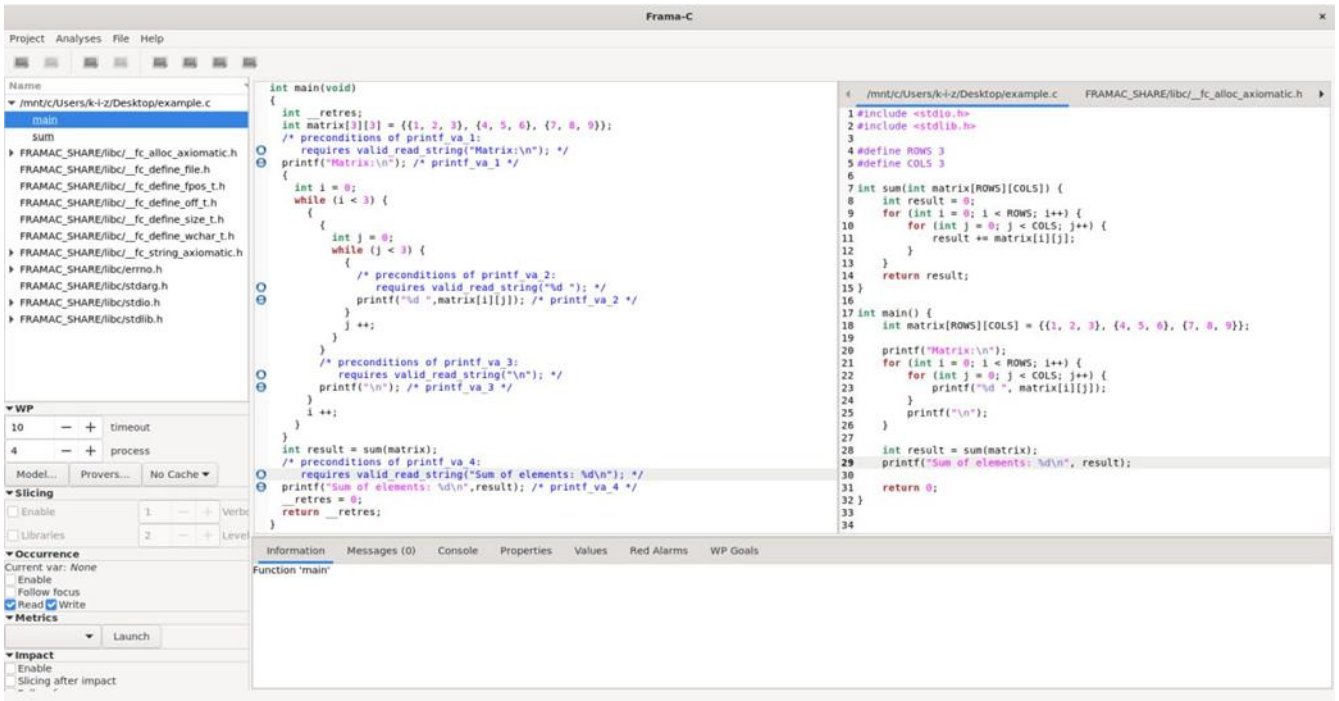

Рисунок 1. Пример программы на платформе Frama-C

В этой программе используются многомерные массивы и функция для вычисления суммы элементов в матрице. С помощью Frama-C можно провести анализ этой программы на корректность и безопасность, например, можно проверить, что функция “sum” корректно работает с матрицами и не приводит к выходу за границы памяти, а также проверить, что программа не содержит ошибок при работе с указателями или памятью.

Далее, проведем анализ программы на корректность и безопасность с помощью Frama-C.

Запустим анализ с помощью следующей команды:

- frama-c -val -no-val-show-annotations example.c

Опции -val и -no-val-show-annotations указывают, что мы хотим провести статический анализ программы с использованием модуля Value, а также не выводить аннотации в результате анализа.

В результате анализа Frama-C выдает следующие предупреждения:

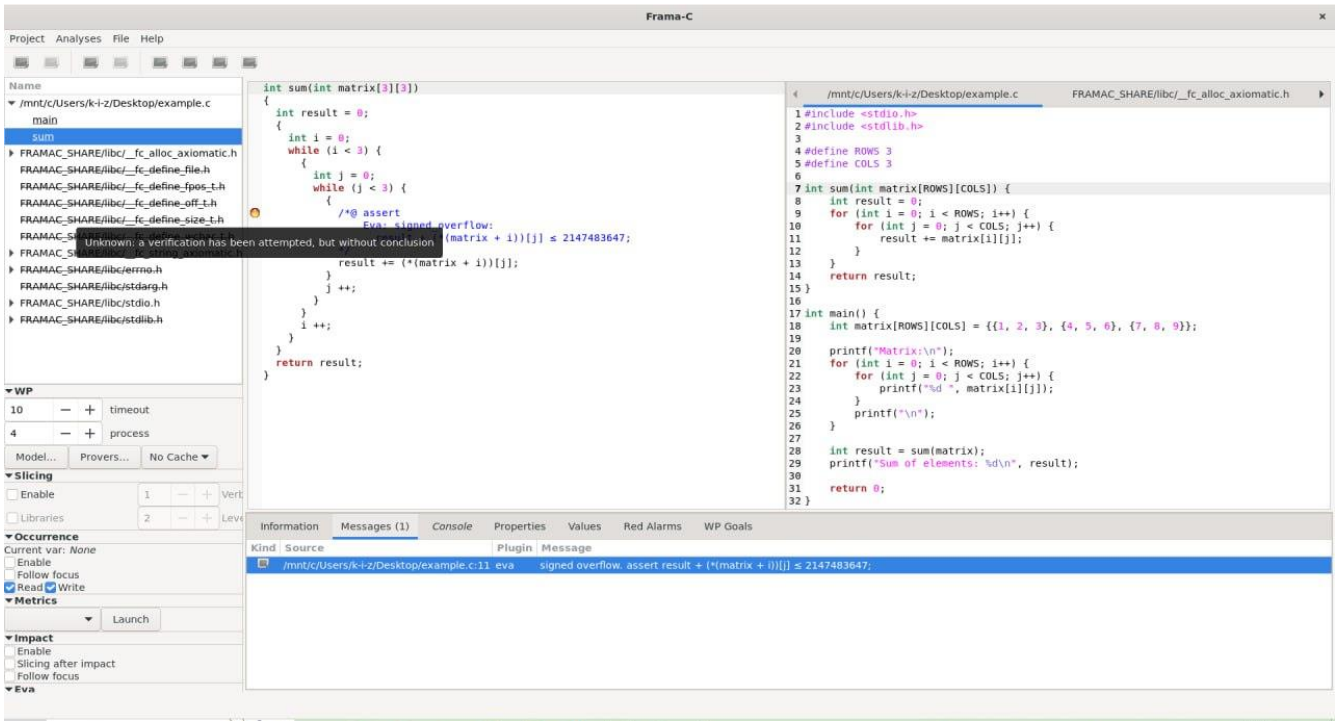

Рисунок 2. Пример анализа программы на платформе Frama-C

Эти предупреждения свидетельствуют о том, что в функции sum возможно выход за границы массива matrix. Это связано с тем, что размеры массива matrix указаны явно и они не соответствуют размеру, который задается при его инициализации в функции main. Для исправления этой проблемы можно использовать макросы или константы, чтобы задавать размеры массива единообразно. Таким образом, анализ с помощью Frama-C позволил выявить проблемы в программе и указать на возможный выход за границы массива. Это позволит программисту устранить эти ошибки до того, как они приведут к серьезным последствиям в реальной системе.

Заключение

В заключении можно отметить, что язык Frama-C является отличным инструментом для анализа и верификации программного кода на языке C. Он позволяет проводить статический анализ кода, обнаруживать ошибки и доказывать корректность и безопасность программы с помощью формальных методов [7]. В статье были рассмотрены основные концепции языка Frama-C, такие как модульность, поддержка спецификаций, анализ потока управления и анализ данных. Также был приведен пример применения Frama-C для анализа корректности и безопасности программного кода. Большой плюс языка Frama-C заключается в том, что он интегрируется с другими инструментами формальной верификации, такими как Why3 и Coq [8]. Это позволяет использовать множество различных методов для доказательства корректности и безопасности программного кода. Несмотря на все преимущества Frama-C, стоит отметить, что он не является универсальным решением для всех задач. В некоторых случаях могут потребоваться другие инструменты и методы верификации программного кода. В целом, язык Frama-C является важным инструментом для разработчиков программного обеспечения, которые заботятся о корректности и безопасности своих программных систем [9]. Он позволяет проводить формальную верификацию кода на языке C, что может помочь избежать многих ошибок и проблем в работе программы.

Список использованных источников

1. Cuellar, J. "Frama-C: A Software Analysis Perspective." Formal Methods for Components and Objects. Springer, Berlin, Heidelberg, 2012. P.233-259.
2. Eigenmann, R. "Towards a Comprehensive Evaluation of Frama-C." Formal Techniques for Distributed Objects, Components, and Systems. Springer, Cham, 2015. P.92-106.
3. "Frama-C User Manual", https://frama-c.com/download/frama-c-user-manual.pdf.
4. Beringer, L. "Verified Low-Level Programming Embedded in Frama-C." Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages. ACM, 2017.
5. Boulmé, S. "Static Analysis of Concurrent C Programs with Frama-C. Formal Techniques for Distributed Objects, Components, and Systems. Springer, Cham, 2016. P.3-19.
6. Marché, C. "Why Frama-C?" Communications of the ACM 61.3. 2018. P.67-75.
7. Griesmayer J. and Grosu R. "Formal Methods for the Verification of Computer Systems", Springer, 2015.
8. "Formal Verification of C Programs with Frama-C", https://hal.archives-ouvertes.fr/hal-01224831/document.
9. Blanchard A. "Introduction to C program proof with Frama-C and its WP plugin", 2020. P. 212.

УДК: 004.855.5

A SURVEY ON STATIC MALWARE ANALYSIS AND DETECTION METHODS

Mukhamadiyev Madiyar
mukhamadiyev_mg_1@enu.kz
MSc Student, L.N. Gumilyov Eurasian National University
Aldosh Balziya
b.nurgaliyeva@astanait.edu.kz
Teacher of Department of Intelligent Systems and Cybersecurity, Astana IT University, Astana, Kazakhstan
Konyrkhanova Assem
konyrkhanova_aa@enu.kz
Associate Professor of the Department of Information Security of the Information technologies faculty, L.N. Gumilyov National Eurasian University

Nowadays the world encounters numerous amounts of malwares, spreading and infecting IT fields, causing business process decline and data loss. To mitigate against malign programs, first must be defined the types of malicious software, which are divided into groups: ransomware, spyware, fileless malware, trojans, adware, worms, rootkits, and more. In this paper, we propose the first steps in automating the static analysis of a malware sample. Analyzing malware involves examining its behavior and characteristics to understand its functionality, purpose, and potential impact on a computer system.

Static malware analysis is a technique used to analyze malware without running it on a computer system. It involves examining the malware's code, structure, and behavior to understand its functionality, purpose, and potential impact on a computer system. Automated instruments such as VirusTotal [1] (Figure 1), which is an internet service and a scanning engine that analyzes suspicious files and accelerates the identification of viruses, worms, trojans and other types of malware detected by antivirus software, can assist in static analysis. The results of scanning files by the service do not depend on any particular antivirus software. In the computer world, a trojan horse is any software that misleads consumers about its true purpose. The story of the trojan horse, which was a trick that led to the destruction of the city of Troy in Ancient Greece, is where the phrase "Trojan Horse" comes from. A computer worm is a separate piece of malicious software that can be found on computers and that is designed to duplicate itself in order to infect other systems. It often spreads over computer networks, betting on security flaws on infected machines to gain access to their systems.