УДК 004. 41:004.056 **doi: 10.26906/SUNZ.2018.4.115**

О. В. Коваленко

Центральноукраїнський національний технічний університет, Кропивницький, Україна

РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ПЕРЕДТЕСТОВОЇ КОМПІЛЯЦІЇ ТА РОЗПОДІЛУ ДОСТУПУ

В даній роботі розроблено інформаційну технологію передтестової компіляції та розподілу доступу в якості практичного застосування в області комп'ютерної інженерії та розробки програмних додатків. В рамках розроблюваного методу розподілу доступу при оптимізації з урахуванням передкомпіляційного профілю програми проводиться необхідний збір даних, що формуються в множині профілів користувачів. Для підвищення точності урахування профілів користувача, специфіки його діяльності та характеристик комп'ютерної системи пропонується розбиття процесу компіляції на дві фази: фаза синтезу програмного забезпечення з урахуванням можливостей сучасних компіляторів; фаза адаптації та розподілу доступу до програмного забезпечення з урахуванням профілів програми і користувача. Такий поділ передтестової компіляції на дві фази дозволить вирішити наступні задачі: розподіл доступу користувачів з урахуванням можливостей персоналізації відповідних профілів; врахування внутрішніх характеристик комп'ютерної системи користувачів (архітектури, планувальника команд, та ін.); врахування можливостей розподілу доступу при збірці та підтримці програмного забезпечення. Для вирішення задач динамічної машинно-незалежної оптимізації доцільно скористатися відомою технологією компіляції LLVM. У запропонованій інформаційній технології передтестової компіляції та розподілу доступу в першій фазі виконується процедура машинно-незалежної компіляції з використанням LLVM. Результат першої фази зберігається у файл LLVM і додатково генеруються дані про архітектуру програмного засобу та алгоритм можливої інсталяції. Виконання другої фази можливо з використанням програмних засобів віртуального моделювання (віртуальних машин), а також безпосередньо на комп'ютерних системах користувачів з урахуванням особливості їх профілів і характеристик обчислювальних засобів. Таким чином, розроблено метод передтестової компіляції та розподілу доступу, що відрізняється від відомих врахуванням профілів користувача при синтезі додатку, а також використанням ресурсів «хмарних сховищ» в процесі отримання інсталяційних версій. Це дозволить підвищити рівень безпеки розроблюваних додатків.

Ключові слова: передтестова компіляція, розподіл доступу, LLVM.

Вступ

Проведені дослідження показали, що в процесі управління розробкою ПЗ, проектування, реалізації, верифікації, а також тестування безпеки ПЗ існує ряд загальних і спеціальних рекомендацій, розроблених і випробуваних на практиці експертами [1, 2]. Серед них можна виділити наступні: управління якістю, інжиніринг вимог безпеки, моделювання загроз, аналіз атак, аналіз вразливостей в наявному коді, перевірка вхідних даних, забезпечення безпеки компілятором, статичний аналіз, проникаюче тестування, аудит коду, розробка посібника і контрольних списків розробників, незалежний огляд безпеки та ін.

Аналіз представлених рекомендацій показав, що в більшості своїй вони зачіпають відомі розробникам ситуації безпеки. У той же час вони не враховують можливостей безпосередньої тестової роботи кодерів-розробників ПЗ в рамках методів «смоук»-тестування і передтестової роботи в рамках компіляції програмного коду.

У той же час, як показують дослідження [1-4], саме в рамках передтестової роботи існують додаткові можливості підвищення безпеки ПЗ за рахунок врахування профілю програм і користувачів, а також оптимізації системи розподілу доступу до досліджуваних даних.

Аналіз досліджень та постановка завдання. В рамках розроблюваного методу розподілу доступу при оптимізації з урахуванням передкомпіляційного профілю програми проводиться необхідний збір даних, що формуються в множині профілів користувачів.

Для підвищення точності урахування профілів користувача, специфіки його діяльності та характеристик комп'ютерної системи пропонується розбиття процесу компіляції на дві фази:

– фаза синтезу ПЗ з урахуванням можливостей сучасних компіляторів;

– фаза адаптації та розподілу доступу до ПЗ з урахуванням профілів програми та користувача.

Такий поділ передтестової компіляції на дві фази дозволить вирішити наступні задачі:

1. Розподіл доступу користувачів з урахуванням можливостей персоналізації відповідних профілів. При цьому враховуються можливості першої фази компіляції – можливості динамічної машинно-незалежної оптимізації на даних конкретного ПЗ і користувача.

2. Врахування внутрішніх характеристик комп'ютерної системи користувачів (архітектури, планувальника команд, та ін.).

3. Врахування можливостей розподілу доступу при збірці та підтримці ПЗ.

Для вирішення задач динамічної машинно-незалежної оптимізації доцільно скористатися відомою технологією компіляції LLVM. З літератури [5] відомо, що в рамках цієї технології представлені статичний компілятор, компонувальник, віртуальна машина, JIT-компілятор. Функціонування системи забезпечується єдиною внутрішньою структурою, яка може бути проілюстрована в текстовому форматі, в формі структур даних в оперативній пам'яті, а також в двійковому вигляді як біт-код. Цей біт-код може бути збережений в проміжних об'єктних файлах для

подальшої оптимізації, в тому числі динамічної. При цьому можливо використовувати всі надані LLVM можливості по обробці внутрішнього представлення (включаючи різні аналізи, трансформації і т.п.). Тому інфраструктура LLVM надає зручну базу для досліджень по динамічній оптимізації програм [5].

Метою роботи є розробка інформаційноъ технології передтестової компіляції та розподілу доступу в якості практичного застосування в області комп'ютерної інженерії та розробки програмних додатків.

Результати досліджень

У розроблюваному методі передтестової компіляції та розподілу доступу в першій фазі виконується процедура машинної-незалежної компіляції з використанням LLVM. Результат першої фази зберігається в файл LLVM і додатково генеруються дані про архітектуру програмного засобу та алгоритм можливої інсталяції.

Виконання другої фази можливо з використанням програмних засобів віртуального моделювання (віртуальних машин), а також безпосередньо на комп'ютерних системах користувачів з урахуванням особливості їх профілів і характеристик обчислювальних засобів.

Слід зауважити, що в другій фазі методу передтестової компіляції та розподілу доступу можливо декілька варіантів реалізації:

1) Автоматична компіляція з урахуванням налаштованої (представленої) архітектури КС.

2) Оптимізація та інсталяція ПЗ з урахуванням профілю поведінки користувача.

3) Оптимізація та інсталяція з урахуванням профілю поведінки користувача на етапі збірки, підтримки (вимушеного простою).

Структурна схема розроблюваного методу представлена на рис. 1.

Слід зауважити, що представлені на рис. 1. етапи і додаткові програмні інструменти передбачені з урахуванням використовуваної технології LLVM.

В ході першої фази розроблюваного методу передтестової компіляції та розподілу доступу пропонується використання технології proxy-компіляції з викликом спеціалізованої утиліти make.sh, а також передачі управління і всіх використовуваних, при формуванні профілю поведінки користувача і розподілу доступу, параметрів. При цьому додатково прописуються команди компілятора LLVM-GCC.

Для підтримки незалежності процесу збірки ПЗ від скриптових додатків, передбачені процедури збереження вихідних результатів. Кінцевим результатом першої фази є інсталяційний пакет зі спеціальними політиками скриптів компіляції та збірки.

Як було зазначено вище, у другій фазі можливі декілька варіантів реалізації на основі статичної та динамічної компіляції (JIT-компілятор) та інсталяції. В обох випадках ці процедури виконуються з урахуванням профілю поведінки користувача або особливостей комп'ютерної системи. У роботі пропонується наступна схема синтезу профілю поведінки користувача (рис. 2):

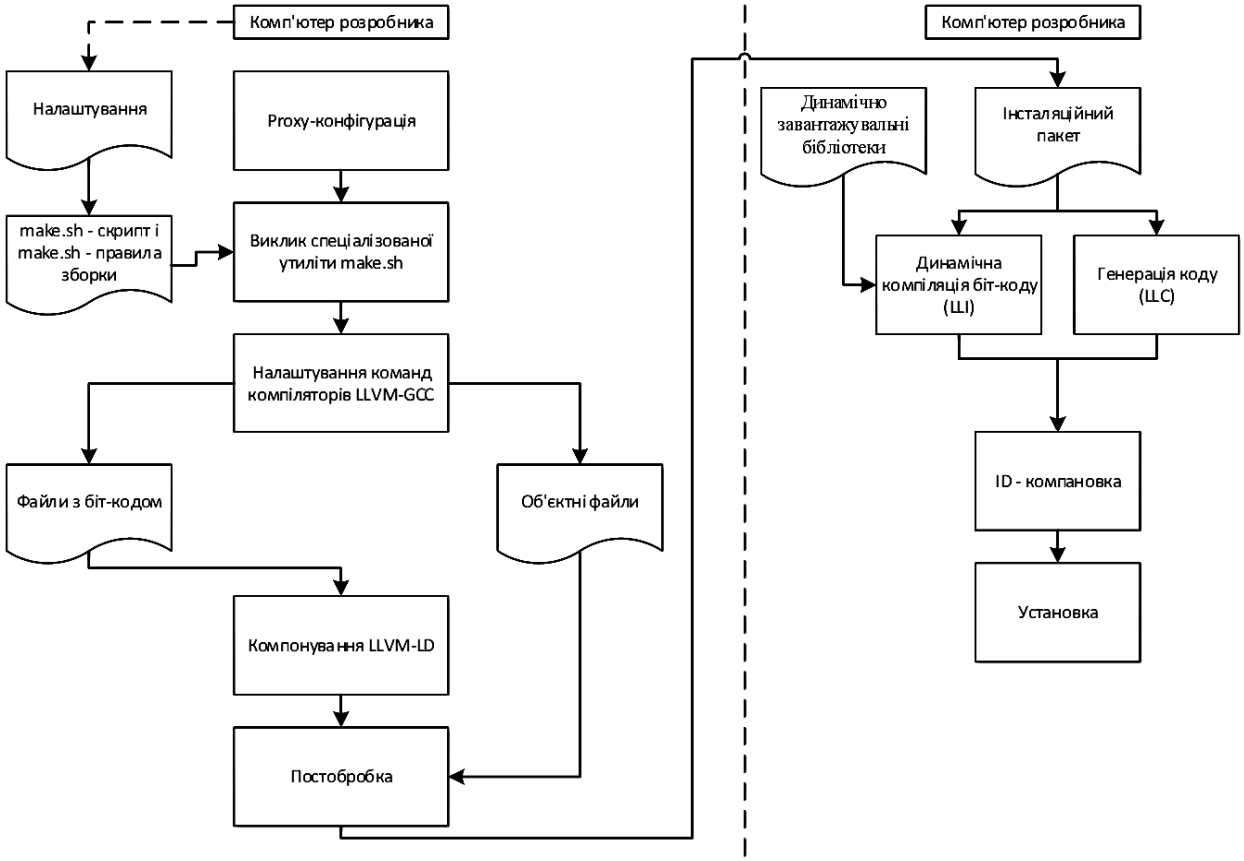

Рис. 1. Структурна схема методу передтестової компіляції та розподілу доступу

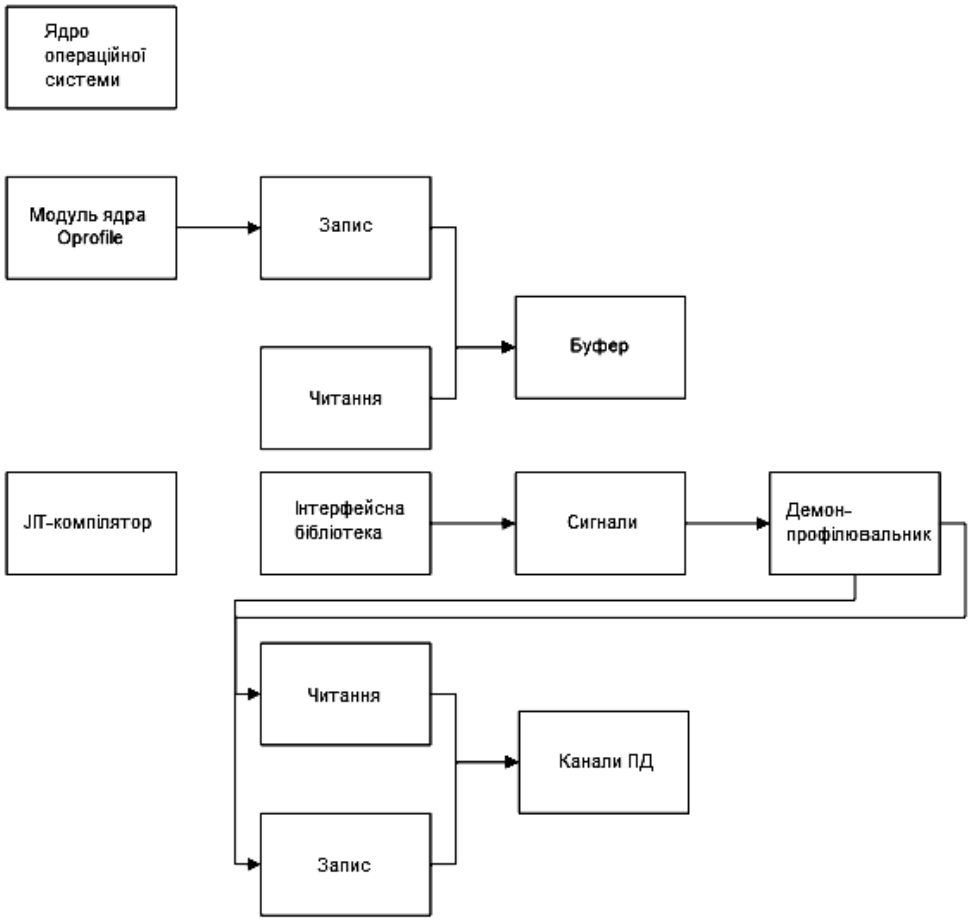

Рис. 2. Схема синтезу профілю поведінки користувача

1) JIT-компілятор отримує із інсталяційного файлу біт-код, компілює його і запускає, а також ініціює процес обміну статистичними даними. Модуль ядра операційної системи виконує збір статистики про особливості комп'ютерної системи і скидає дані статистики в буфер пам'яті.

2) «Демон» профілю користувача має можливість доступу до даних і буфера, розпізнає і записує їх в спеціалізовані канали передачі.

Оскільки код програми, згенерованої в першій фазі, непомітний системі і виконується машинно-незалежним компілятором, він може бути поміщений в незалежні простори комп'ютерної системи. «Демон» вибирає файл з прапором «незалежні дані», записує цю інформацію в канал ПД і продовжує роботу до отримання відповідної команди про завершення роботи.

Для коректного використання зібраної інформації про конфігурацію та поведінку користувача під час компіляції необхідне підлаштування отриманого динамічного профілю до точного профілю по базовим блокам і ребрам графа потоку управління програми.

Оскільки дані профілю отримані шляхом синтезу окремих даних, то в них неминуче є похибки, що виникають через період їх отримання, а також затримок на передачу даних компілятору. Отже, такі дані не можуть бути безпосередньо використані оптимізаційними фазами LLVM таким же чином, як і дані статичного профілю – наприклад, побудований профіль по ребрам графа потоку управління не буде задовольняти рівнянням потоку.

Для вирішення цієї задачі можна використовувати алгоритм, описаний в роботах [6] фахівців Google для компілятора GCC, в яких автори пропонують підходи модифікації профілю користувача з урахуванням обмежень задачі про максимальний потік мінімальної вартості.

Основні блоки алгоритму представлені на рис. 3.

Алгоритм приймає на вхід структурований по базовим блокам профіль користувача, а також інформацію статичного аналізу циклів: вона буде необхідна для попередньої оцінки ваг ребер графа потоку управління функції. Ос-

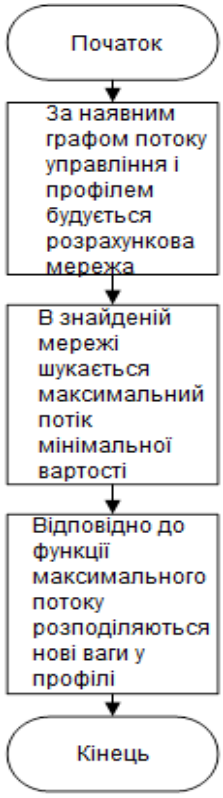

Рис. 3. Основні блоки алгоритму переоцінки профілю поведінки користувача

кільки профіль користувача синтезований для інструкцій, сумарна кількість блоків даних для профілю на базовому блоці оцінимо як середнє по частотах інструкцій.

Одним з варіантів рішення поставленої задачі є використання алгоритму Голдберга-Рао [7], що є модифікацією алгоритму Форда-Фалкерсона [7].

Цей, досить сучасний (1997 р.) алгоритм, вперше поліпшив оцінку *O(nm)*, що вважалася непереборною. Його результат: *O(min{n2/3,n1/2}m Log(n2/m)LogU)* [7].

Алгоритм є слабо-поліноміальним, але, з огляду на теорему подібності Габова [8], в якій для порівняння слабо і сильно-поліноміальних алгоритмів використовується *LogU = O(Log n)*, серед фахівців він знайшов позитивну оцінку.

У кожній своїй ітерації, алгоритм Голдберга-Рао шукає тупиковий потік, використовуючи неодиничну функцію для визначення довжини допустимих дуг. Після побудови максимального потоку будується залишкова мережа, в якій за алгоритмом, представленим в роботах [8] видаляються цикли негативної вартості.

По мережі максимального потоку мінімальної вартості відновлюється вихідний граф потоку управління: по ребрам, побудованим в результаті перетворення вершин, ми відновлюємо профіль вершин. Ребра, відповідні вихідному графу не видаляються методом, тому що їм була призначена нескінченна максимальна пропускна здатність, і значить, теж можуть бути відновлені.

Слід зауважити, що побудовані лічильники можна використовувати в будь-яких оптимізаціях, що підтримують профіль, як результати збірки звичайного профілю способом інструментування. Проте, безпосереднє застосування профілю до наявного біт-коду призведе до помилок невідповідності профілю – це обумовлено тим, що перед запуском кодогенерації LLVM додатково виконує кілька проходів, що змінюють машинно-незалежне представлення, і зібраний профіль вже не відповідає збереженому біт-коду.

Таким чином, для практичного застосування профілю поведінки користувача, зібраного шляхом синтезу блоків даних, необхідно виконувати правила і алгоритми, описані вище, а також процедури оптимізації, що виконуються над машинно-незалежним представленням.

Метод передтестової компіляції та розподілу доступу дозволяє виконувати процедури оптимізації на машині користувача як динамічно, через створення відповідних JIT-компіляторів, так і під час простою програми, з урахуванням зібраного профілю, що відображає поведінку користувача, і характеристик його машини.

Однак, в сучасних умовах використання мобільних засобів, найчастіше оптимізація програм на пристрої є скрутною (не вистачає необхідної кількості ресурсів). Вирішити це протиріччя можна компілюючи на сервері додатків, при цьому на пристрій покласти лише завдання збору та передачі профілю на сервер.

Для організації такого сервера потрібні вдосконалені методики (методи) безпечної передачі додатків. В роботі завдання передачі і обробки додатків пропонується вирішити за допомогою методу безпечної маршрутизації метаданих в хмарні антивірусні системи, описаного в роботі [9].

У цій роботі передачу даних пропонується організувати безпечним шляхом в «хмарне сховище» додатків, що забезпечує як переносимість програм в рамках одного сімейства процесорних архітектур ARM, так і високу ступінь безпеки додатків, що зберігаються.

Після передачі і розподілу додатка в «хмарне сховище» для забезпечення безпеки пакет може бути перевірений інструментами середовища Svace на наявність критичних помилок і вразливостей (використовуючи біт-код LLVM, а також інформацію про зв’язки програми), а також проведено тест безпеки методами, використаними в середовищі Tfex.

Після перевірки додатку на безпеку, за запитом завантажити додаток конкретним користувацьким пристроєм в сховище, здійснюється другий етап схеми двоетапної компіляції і генерується інсталяційний пакет з об’єктним кодом для конкретного пристрою.

Таким чином, розроблено метод передтестової компіляції та розподілу доступу, що відрізняється від відомих врахуванням профілів користувача при синтезі додатку, а також використанням ресурсів «хмарних сховищ» в процесі отримання інсталяційних версій.

Це дозволить підвищити рівень безпеки розроблюваних додатків.

Висновки

В якості практичного застосування в області комп’ютерної інженерії та розробки програмних додатків в роботі розроблено метод передтестової компіляції та розподілу доступу. Відмінною особливістю методу є врахування профілів користувача при синтезі додатку, а також використанням ресурсів «хмарних сховищ» в процесі отримання інсталяційних версій. Це дозволить підвищити рівень безпеки розроблюваних додатків.

Подальші дослідження направлені на визначення ряду практичних рекомендацій по використанню методів та засобів управління безпекою додатків.

СПИСОК ЛІТЕРАТУРИ

1. Kniberg Henrik Scrum and XP from the Trenches – 2nd Edition. InfoQ 2015. 94 c.
2. Канер Сем Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений . К.:ДиаСофт, 2001. 544 с.
3. Семенов С. Г., Халифе К., Захарченко М. М.. Усовершенствованный способ масштабирования гибкой методологии разработки программного обеспечения. Сучасні інформаційні системи = Advanced Information Systems. 2017. № 1(1). С. 79-84.

4. Gary Stoneburner, Alice Goguen, and Alexis Feringa Risk Management Guide for Information Technology Systems Recommendations of the National Institute of Standards and Technology // Computer Security Division Information Technology Laboratory National Institute of Standards and Technology Gaithersburg. 2002. 55 с.
5. С.С. Гайсарян, Ш.Ф. Курмангалеев, К.Ю. Долгорукова, В.В. Савченко, С.С. Саргсян Применение метода двухфазной компиляции на основе LLVM для распространения приложений с использованием облачного хранилища URL:https://cyberleninka.ru/article/n/primenenie-metoda-dvuhfaznoy-kompilyatsii-na-osnove-llvm-dlya-rasprostraneniya-prilozheniy-s-ispolzovaniem-oblachnogo-hranilischa
6. LLVM: компилятор своими руками. Введение. URL: https://habrahabr.ru/post/277717/
7. Алгоритм Голдберга-Рао. URL: http://algolist.manual.ru/maths/graphs/maxflows/Goldberg_Rao.php
8. Гмурман В.Е. Теория вероятностей и математическая статистика. М.: Высшая школа, 2003. 479 с.
9. Смірнова С.А. Метод антивирусной защиты данных с использованием облачных вычислительных технологий: дис. на здобуття наук. ступеня канд. техн. наук: 05.13.21. Київ, 2017. 174 с.

Рецензент: д-р техн. наук, проф. О.А. Смірнов,
Центральноукраїнський національний технічний університет, Кропивницький
Received (Надійшла) 25.06.2018
Accepted for publication (Прийнята до друку) 29.08.2018

**Разработка информационной технолог
ии предтестовой компиляции и распределения доступа**

А. В. Коваленко

В работе разработана информационная технология предтестовой компиляции и распределения доступа в качестве практического применения в области компьютерной инженерии и разработки программных приложений. В рамках разрабатываемого метода распределения доступа при оптимизации с учетом передкомпиляционного профиля программы проводится необходим сбор данных, формируемые во множестве профилей пользователей. Для повышения точности учета профилей пользователя, специфики его деятельности и характеристик компьютера предлагается разбивка процесса компиляции на две фазы: фаза синтеза программного обеспечения с учетом возможностей современных компиляторов; фаза адаптации и распределения доступа к программному обеспечению с учетом профилей программы и пользователя. Такое разделение предтестовой компиляции на две фазы позволит решить следующие задачи: распределение доступа пользователей с учетом возможностей персонализации соответствующих профилей; учета внутренних характеристик компьютера пользователей (архитектуры, планировщика команд и др.); учета возможностей распределения доступа при сборке и поддержке программного обеспечения. Для решения задач динамической машинно-независимой оптимизации целесообразно воспользоваться известной технологией компиляции LLVM. В предлагаемой информационной технологии предтестовой компиляции и распределения доступа в первой фазе выполняется процедура машинно-независимой компиляции с использованием LLVM. Результат первой фазы сохраняется в файл LLVM и дополнительно генерируются данные об архитектуре программного средства и алгоритм возможной установки. Выполнение второй фазы возможно с использованием программных средств виртуального моделирования (виртуальных машин), а также непосредственно на компьютерных системах пользователей с учетом особенности их профилей и характеристик вычислительных средств. Таким образом, разработан метод предтестовой компиляции и распределения доступа отличается от известных учетом профилей пользователя при синтезе приложении, а также использованием ресурсов «облачных хранилищ» в процессе получения установочных версий. Это позволит повысить уровень безопасности разрабатываемых приложений.

Ключевые слова: предтестовая компиляция. распределение доступа, LLVM

**Development of information technology
for pretest compilation and distribution of access**

O. Kovalenko

In this work the information technology of pretest compilation and distribution of access as a practical application in the field of computer engineering and development of software applications is developed. In the framework of the developed method of distribution of access, during the optimization taking into account the precompiling profile of the program, the necessary data collection formed in a set of user profiles is performed. To improve the accuracy of consideration of the user profiles, specific character of their activity and characteristics of the computer system, it is proposed to split the compilation process into two phases: the phase of software synthesis that takes into account the capabilities of modern compilers; the phase of adaptation and distribution of the software access that takes into account the profiles of the program and the user. Such a division of pretest compilation into two phases will allow to solve the following tasks: to distribute user access taking into account the possibilities of personalization of the corresponding profiles; to take into account the internal characteristics of the user's computer system (its architecture, task planner, etc.); to take into account the possibilities of distribution of access when assembling and maintaining the software. To solve the problems of dynamic machine-independent optimization it is advisable to use the known technology of compilation of LLVM. In the first phase of the proposed information technology of pretest compilation and distribution of access the process of machine-independent compilation using the LLVM is performed. The result of the first phase is stored in the LLVM file and additional data about the software architecture and the possible installation algorithm is generated. The execution of the second phase is possible with the use of virtual simulation software (virtual machines), as well as directly on users' computer systems taking into account the features of their profiles and characteristics of computing means. Thus, a pretest compilation and distribution method is developed that differs from the known by taking into account user profiles in the synthesis of the application, as well as by the use of cloud storage resources while obtaining installation versions. This will increase the security of the applications being developed.

Keywords: pretest compilation, distribution of access, LLVM