УДК 004.056:004.4.052(045)

Сінько Ю. І., к.п.н.

КОНЦЕПЦІЯ НАДІЙНИХ І БЕЗПЕЧНИХ ОБЧИСЛЕНЬ

Національний авіаційний університет
yusin@ukr.net

Розглянуто основні принципи розробки надійних і безпечних програм. Запропонована концепція надійних і безпечних обчислень (trustworthy computing) може бути використана розробниками програмного забезпечення для покращення надійності та безпеки своїх комп'ютерних систем і програм.

Ключові слова: надійні і безпечні обчислення, безпека коду, надійність коду, конфіденційність інформації, оперативність і коректність бізнесу

Постановка проблеми

Одним з найбільш важливих і перспективних напрямків інформаційних технологій (ІТ) є надійне і безпечне програмування. У більш широкому значенні, в сучасній англомовній термінології, даний термін звучить як *trustworthy computing*, що в буквальному перекладі означає обчислення, що заслуговують довіри. Що це означає на практиці?

Основна частина

Як відомо, протягом усього розвитку програмування та програмного забезпечення, за будь-яких підходах до розробки програм, найбільш важливими їх якостями визнавалися працездатність і корисність (*usability*) – програма повинна виконувати те, чого від неї очікують, у заданих умовах, а також бути дружньою до користувача (бути зручною), мати комфортний інтерфейс користувача, «вести себе розумно», з точки зору життєвої та професійної логіки. Тільки така програма, на наш погляд, заслуговує довіри користувачів, тому і ставимо ці якості на перше місце при визначенні поняття *trustworthy computing*.

Слід визнати, що найважливішою якістю програм є їх безпека *(security[1])*: програма повинна вміти захищатися від зовнішніх загроз і атак та зберігати працездатність після їх відбиття. Сьогодні на це звертають особливу увагу, оскільки, через широке розповсюдження мереж і через ускладнення архітектури програм, з'явилося набагато більше можливостей для хакерських атак, наслідки яких можуть бути дуже серйозні, так як використання програмного забезпечення стало частиною нашого повсякденного життя.

Інша найважливіша якість програми – надійність *(reliability[2])*: програма повинна «вести себе розумно» і передбачувано у разі некоректних вихідних даних і забезпечувати безвідмовну роботу протягом якомога довшого періоду часу. Поняття надійності раніше було прийнято кількісно характеризувати в термінах *Mean Time Between Failures (MTBF)* - середнього часу напрацювання на відмову, тобто середнього часу, протягом якого система працює безвідмовно. Тому сьогодні потрібні більш детальні і точні кількісні оцінки і передбачення надійності, над якими і ведеться робота.

Третє. Програма повинна забезпечувати захист і коректність використання конфіденційної інформації *(privacy[3])* – особистих і банківських даних, інформації, що використовується в повсякденній роботі підприємства та ін. На такий вид інформації необхідно отримати дозвіл у

[1] Стійкість коду програмного продукту до зовнішніх загроз і атак, бути здатним до їх запобігання та відбиттю (ослаблення їх ефекту).

[2] Здатність коду програмного продукту забезпечувати специфіковану для нього поведінку, яка передбачувана в заданих умовах.

[3] Збереження кодом програмного продукту конфіденційного статусу для любої, що використовується ним приватної чи корпоративної інформації про користувача, його додатках та інфраструктури.

користувача і використовуватися тільки в разі крайньої необхідності, для конкретних цілей, які програма повинна зрозуміло пояснювати користувачеві, своєчасно попереджувати його про це і заручатися його схваленням і довірою.

Четверте. Фірма-розробник програми повинна забезпечувати цілісність і коректність бізнесу *(business integrity*[4]*)*, пов'язаного з розробкою і використанням програми. Дана якість має дві сторони. По-перше, це чітка організація супроводження програми – швидке виправлення помилок і швидкі відповіді на запитання користувачів, які, в свою чергу, повинні оцінити хорошу організацію бізнесу, пов'язаного з розробкою і супроводженням програми, і, як наслідок, продовжувати користуватися програмою і тим самим приносити її розробникам дохід. По-друге, це контроль прозорості, законності, коректності бізнесу компанії-користувача програми. Мається на увазі, насамперед, що програма повинна підказувати користувачу, як її використовувати коректно. Якщо використання програми пов'язано з веденням якого-небудь бізнесу – продажів, реклами, перерахування коштів тощо, – то програма повинна контролювати коректність і законність всіх операцій, які користувач виконує з її допомогою, а в разі виявлення будь-яких відхилень від цього правила – попереджати і консультувати користувача, щоб запобігти його можливі несанкціоновані та незаконні дії.

Чотири перерахованих якості програм – *security, reliability, privacy, business integrity*, яким сьогодні надається особливе значення, стали основою ініціативи *trustworthy computing (TWC)*[1], проголошеної в 2002 р. корпорацією Microsoft. У меморандумі *TWC* фірми Microsoft вони названі «Чотирма колонами *TWC» (four pillars)*. Якість *usability* ми додаємо до розглянутої парадигми, виходячи з міркувань здорового глузду.

Головна мета ініціативи *TWC* – звернути увагу фірм-розробників програмного забезпечення на особливу важливість зазначених якостей програм. А розробники повинні прагнути до того, щоб враховувати вимоги *TWC*, починаючи з самих ранніх етапів розробки.

Класичною роботою по *TWC* стала книга [2], в якій детально, на змістовних прикладах, пояснюються принципи розробки безпечного коду. Книга має додаток на *Web*-сайті, де доступні для скачування через *Web* корисні приклади безпечного коду.

З метою докорінного покращення надійності та безпеки власних програм фірма Microsoft розробила і застосовує особливу схему життєвого циклу розробки безпечних програм *(Security Development Lifecycle, SDLC)* [3]. Вона є результатом аналізу серйозних уроків, що вдалося витягти *Microsoft* з досвіду експлуатації попередніх версій *Windows*, до *Windows X*P. Для покращення безпеки вже випущеної і поширеною по всьому світу ОС *Windows* 2000, яка була розроблена ще до введення *SDLC*, *Microsoft* довелося терміново перервати подальшу розробку системи, «посадити за парти» всіх її розробників, навчити їх у найкоротші можливі терміни принципам розробки безпечних програм, а потім за короткий термін виправити вже існуючий, гігантський код операційної системи і поширити її нову, більш надійну і безпечну версію. У світову історію програмування цей безпрецедентний крок увійшов під назвою «*Security push*» [2].

Щоб уникнути подібних дорогих екстрених заходів при розробці нового програмного забезпечення, *Microsoft* рекомендує застосовувати схему *SDLC* – проектувати, розробляти і тестувати реалізацію підсистеми безпеки на кожному етапі життєвого циклу програми, тобто займатися безпекою програми постійно, починаючи з самих ранніх етапів її розробки. У цьому – основна суть схеми *SDLC*.

Оскільки для практичного виконання даного принципу потрібні висококлас-

[4] Швидкість, якість і повнота супроводження програмного продукту, а також дотримання фірмою-розробником програмного продукту всіх законодавчих актів і юридичних установлень.

ні експерти з комп'ютерної безпеки, якими більшість інженерів-програмістів, на жаль, аж ніяк не являються, – *Microsoft* рекомендує в кожному проекті мати невелику групу експертів з безпеки, або хоча б одного експерта, можливо, запрошеного з іншої компанії, – «*security buddy*». Мета експертної групи з безпеки – постійний контроль дотримання правил розробки безпечного коду на всіх етапах розробки, консультації розробників з безпеки та ін.

Згідно з принципами *TWC і SDLC*, безпека системи починається ще до її розробки. У документі, що описує вимоги (*requirements)* до системи, повинні бути окремими розділами сформульовані вимоги до безпеки, типові можливі загрози *(threats)* системи та методи їх пом'якшення *(mitigation).* Для здійснення цих принципів необхідно ще на початкових етапах розробки займатися моделюванням загроз *(threat modeling)* [4].

Перерахуємо найважливіші принципи розробки безпечного коду, згідно TWC:

- Принцип мінімізації поверхні програми яку атакують *(minimizing the attack surface).* Необхідно проектувати і реалізовувати програму так, щоб хакери мали якомога менше можливостей «зламати» дані працюючої програми і вивести її з ладу.

- Принцип мінімальних привілеїв *(least privilege).* У більшості випадків розробка та налагодження програм ведеться розробниками «від імені адміністраторів», тобто програма тестується від імені користувача, що має права адміністратора, наприклад, право зміни реєстру або інших системних файлів. При спробі використовувати таку програму від імені звичайного користувача можуть виникнути проблеми, пов'язані з відсутністю повноважень. Щоб цього не сталося, програму необхідно обов'язково тестувати від імені звичайного непривілейованого користувача. Функціональність програми не повинна спиратися на те, що користувач має права адміністратора.

- Принцип забезпечення безпеки згідно з проектом, за замовчуванням, при розгортанні *(secure by design, by default, by deployment).* Як зазначалося раніше, ще на етапі проектування програми необхідно передбачити для неї заходи безпеки. Програма має бути реалізована так, щоб бути безпечною за замовчуванням, тобто всі заходи і перевірки безпеки за замовчуванням повинні бути включені (навіть якщо це призведе до деякого уповільнення програми або деяких незручностей для користувача). Нам добре відомо, що останнє означає на практиці: ОС і браузер запрошують додаткові підтвердження від користувача в разі потенційно небезпечних для системи дій – перегляду незнайомого сайту, викачування і інсталяції чужих програм та ін. Це характерно для всіх версій *Windows*, починаючи з 2002 р.

Світова спільнота програмістів спочатку дещо скептично поставилася до ініціативи *TWC*, однак за 13 років ситуація докорінно виправилася. Проблеми з безпекою і з зовнішніми атаками відчувають програмні продукти всіх фірм-розробників і мільйони їх користувачів. *Microsoft* своєю ініціативою *TWC* подає приклад іншим компаніям, індивідуальним розробникам і користувачам, і тепер справа – за нами: ми повинні цю ініціативу підхопити і розвивати, для нашої з вами користі і безпеки програм, які ми розробляємо і використовуємо.

Стосовно нас, шановні колеги по IT-освіті, це означає, насамперед, що ми повинні вчити наших студентів принципам, технологіям і інструментам *TWC* і їх практичного застосування.

На жаль, поки не тільки в Україні, але й у всьому світі навчання *TWC* недостатньо поширене. Цим займаються, головним чином, університети і коледжі в США і Канаді, орієнтовані на військові розробки, – історично саме військові і близькі до них організації, з цілком зрозумілих причин, відчувають особливий інтерес до комп'ютерної безпеки.

Така ситуація має бути терміново виправлена – неможливо більше мирити-

ся з (як доводиться визнати) висококваліфікованими атаками хакерів, які зламують захист банків, найважливіші сайти, що проникають до нас з недобрими намірами через IP-мережі, електронну пошту тощо, – і при цьому завдають багатьом з нас щодня відчутний матеріальний і моральний збиток.

Як уже зазначалося, однією з найсерйозніших проблем розвитку *TWC*, як наукового напрямку, є розробка кількісних оцінок *(quantitative assessment)* безпеки, ризику при розробці та використанні програм, надійності програм. У цьому відношенні слід відзначити роботи [5, 6], завдяки яким, вперше в більш ніж 50-річній практиці програмування, стає можливим використання реалістичних і практично орієнтованих кількісних оцінок якостей і властивостей програм, які відповідають основним положенням *TWC*. Методи таких оцінок засновані на методах математичної статистики.

Підкреслимо, що без кількісної оцінки характеристик *TWC* застосування принципів TWC в значній мірі втрачає як наукове так і практичне значення.

Висновки

1. Надійні та безпечні обчислення (*trustworthy computing)* – актуальний напрямок досліджень і розробок в області IT, так як від нього залежить надійність і безпека часом життєво важливих програмних продуктів повсякденного використання.

2. Складові частини ініціативи *Trustworthy Computing* фірми *Microsoft* – це безпека, надійність, дотримання конфіденційності інформації, оперативність і коректність бізнесу з розробки і супроводження програм.

Список літератури

1. Web-сторінки корпорації Microsoft, присвячені Trustworthy Computing. Режим доступу: http://www.microsoft.com/mscorp/twc/default.mspx.

2. Howard M. Writing Secure Code. / M. Howard, D. LeBlanc // Microsoft Press, 2002.

3. Howard M. Security Development Lifecycle. / M. Howard, S. Lipner // Microsoft Press, 2006.

4. Snyder W. Threat Modeling / W. Snyder, F. Swiderski // Microsoft Press, 2004.

5. Bernstein L. Trustworthy Systems Through Quantitative Software Engineering. / L. Bernstein, C.M. Yuhas // Wiley-IEEE Computer Society Press, 2005, 437 pp.

6. Sahinoglu M. Trustworthy Computing. Analytical and Quantitative Engineering Evaluation / M. Sahinoglu // Wiley Interscience. John Wiley & Sons, 2007, 320 pp.

7. Safonov V.O. Using aspect-oriented programming for trustworthy software development. / V.O. Safonov // Wiley Interscience. John Wiley & Sons, June 2008. ISBN 978-0-470-13817-5.

Статтю подано до редакції 19.03.2015