УДК 004.056.5
О. Зимницький
Тернопільський національний технічний університет імені Івана Пулюя

ВРАЗЛИВОСТІ РЕАЛІЗАЦІЇ КРИПТОГРАФІЧНИХ МЕТОДІВ ЗАХИСТУ ПРОТОКОЛУ SSL/TLS

UDC 004.056.5
O. Zymnytskyi
(Ternopil Ivan Puluj's National Technical University, Ukraine)

VULNERABILITIES OF THE IMPLEMENTATION OF CRYPTOGRAPHIC METHODS IN SSL/TLS PROTOCOL

Сучасні методи криптографічного захисту інформації забезпечують достатній рівень захисту оскільки вони ґрунтуються на складних математичних принципах, а за рахунок відкритості алгоритму їх ретельний криптоаналіз проведений і продовжує проводитися значною кількістю спеціалістів у сфері захисту інформації по всьому світу.

У зв'язку з цим вразливості у методах шифрування досить рідкісне явище. Частіше вони стають застарілими через довжину ключа чи інші фактори і з розвитком апаратного забезпечення та технологіями розподілених обчислень стають вразливими до атак перебору чи різного роду колізійних атак.

Проте у мережі ці методи криптографічного захисту використовуються у рамках певного протоколу, найпопулярнішим з яких на сьогодні є TLS. І вразливості можуть бути закладеними у саму архітектуру протоколу, як це трапилося з SSLv3 і вразливістю CVE-2014-3566 (так звана POODLE атака). Невідомо скільки зловмисників користувалися цією атакою до моменту її виявлення у 2014 році. Чи CVE-2011-3389 (BEAST) до якої були вразливі всі блокові шифри, що використовуються у SSLv3 і TLSv1.0

Однак найчастіше виникають вразливості через помилки робробників бібліотек, що реалізують TLS протокол. Лише за поточний рік у CVE зареєстровано 9 вразливостей OpenSSL, дві у GnuTLS. І чим більшою є спільнота розробників тим важче попереджати чи виявляти навмисне чи ненавмисне внесення вразливостей у вихідний код бібліотеки чи програми, що реалізує криптографічний захист інформації і виникає потреба в автоматизації цього процесу.

Для попередження виникнення таких вразливостей необхідно проводити ретельний аналіз змін, що роблять розробники та всього коду в цілому. Це можна робити за допомогою статичного та динамічного аналізу коду, зокрема модульного та інтеграційного тестування коду.

Сучасні статичні аналізатори здатні виявляти вразливі ділянки коду, це неправильне використання пам'яті, блокування потоку чи виклик вразливих чи застарілих функцій з зовнішніх бібліотек.

Динамічний аналіз передбачає запуск програми або її частин. Деякі аналізатори дозволяють виявити приховані вади, переважно пов'язані з неправильним доступом до оперативної пам'яті, що можуть становити загрозу.

Модульне та інтеграційне тестування робить код криптографічної підсистеми більш передбачуваним. Наприклад, це дозволяє впевнитися у правильності реалізації (тобто відповідності стандартам) окремих алгоритмів шифрування, хешування та ЕЦП, а також у їх сумісності під час передачі інформації по мережі. Можна перевіряти різні сценарії атак та заносити існуючі у попередніх версіях вразливості для попередження їх появи знов. Проте, все залежить від якості подібних тестів, тобто наскільки добре вони покривають код. Це можна визначати по відсотку покритих тестами стрічок (англ. code coverage), що викликаються під час тестування, а також за допомогою мутаційного тестування. Останнє суттєво покращує якість автоматизованих тестів.