УДК 004.056.53

АНАЛИЗ УЯЗВИМОСТЕЙ ПРИЛОЖЕНИЙ И ПРИЧИН ИХ ВОЗНИКНОВЕНИЯ

Михаил Коломыцев, Светлана Носок
Национальный технический университет Украины «Киевский политехнический институт»

Анотація: **Розробники додатків відіграють важливу роль у побудові безпечних комп'ютерних систем. Аналіз можливих вразливостей, причин їх виникнення дозволить розробникам точніше ідентифікувати можливі проблеми безпеки додатків, концентрувати зусилля на створенні захищених додатків.**
Summary: **Developers of applications are playing great role in constructing of secure computer systems. The analisys of all possible threats, and reasons of their appearens will helpdevelopers to make better identification of possible problems of application security, and concentrate their acts on creating secure applications.**
Ключові слова: **Вразливість, ідентифікація, аутентифікація, SQL-injection**

I Задача анализа (таксономии) уязвимостей

Реализация основного функционала приложения, сервисов безопасности, вспомогательных служб может сопровождаться возникновением уязвимостей, создающих «бреши» в системе защиты приложения.

Задача анализа уязвимостей является актуальной для разных категорий разработчиков. Для программистов анализ возможных уязвимостей, причин их возникновения позволит точнее идентифицировать возможные проблемы безопасности приложений, концентрировать усилия на использовании методов безопасного кодирования. Тестировщики используют анализ уязвимостей для построения корректной программы тестирования. Системным администраторам такая информация позволяет понять, каким образом может быть атакована их система.

Задача систематизации уязвимостей рассматривалась многими авторами (см. список литературы). В данной работе предлагается подход к анализу уязвимостей, согласно которому их можно разделить на следующие категории:

- уязвимости, возникшие на этапах проектирования и разработки (кодирования);
- уязвимости, возникшие вследствие некорректного конфигурирования или администрирования приложения в процессе его развертывания;
- уязвимости, возникшие вследствие того, что в приложении реализованы стандарты, не отвечающие требованиям безопасности (например, поддержка стандарта SSL v.2).

II Анализ причин появления уязвимостей в приложениях

Причины возникновения уязвимостей можно отнести к одной из следующих групп:

- недостаточная проработка требований к безопасности разрабатываемого приложения;
- ошибки проектирования. Недостаточный контроль за реализацией требований безопасности в процессе разработки. Например, плохо спроектированный механизм управления сессиями может позволить пользователям в Web-приложении манипулировать файлами ***cookies***, что позволит обойти процедуру аутентификации.
- ошибки кодирования. Ошибки кодирования могут быть использованы для изменения функциональности приложения, выполнения непредусмотренных действий и команд. Реализация атак становится возможной за счет переполнений буфера, ошибок форматных строк, использования интервала времени между моментом проверки права доступа к файлу и моментом его использования для изменения полномочий (race conditions). Судя по публикациям, ошибки кодирования являются самой популярной причиной возникновения уязвимостей.
- злоумышленный код. Люки, логические бомбы, код для организации атаки «салями» – характерные примеры кода, внедренного разработчиками для последующей эксплуатации.
- ошибки разворачивания (внедрения) приложения. Такие ошибки становятся следствием недостаточно продуманной и подготовленной установки приложения у заказчика. Связаны они с тем, что характеристики компьютерной инфраструктуры известны не точно. В эту категорию

попадают такие ошибки, как не удаленные отладочные учетные записи и пароли, ошибки контроля версий приложения.

- недостаточный контроль качества и тестирования приложения. Избавиться от уязвимостей только на этапе заключительного тестирования невозможно. Вопросы безопасности должны быть включены в программы по контролю качества и тестирования приложений. Регулярное тестирование приложения должно проводиться не только в предположении, что обрабатываемые данные будут иметь нормальный, ожидаемый характер, но и учитывать сценарии возможных атак.

III Основные виды уязвимостей

Поиск уязвимостей – задача непростая. Хотя на рынке присутствуют разнообразные средства обнаружения известных уязвимостей, наиболее эффективным путем выявления большинства уязвимостей остается экспертный анализ исходного кода приложения. Надо отметить, что инструментальные средства обнаружения уязвимостей, как с открытым кодом, так и коммерческие успешно используются не только разработчиками, но и хакерами.

Приведенный список уязвимостей собран авторами на основе анализа публикаций и исследований, посвященных разработке защищенных приложений (см. список литературы).

1. **Неадекватная идентификация и аутентификация.** В эту группу в частности входят случаи, когда пользователь может не проходить процедуру аутентификации, либо аутентификация не требуется вообще.
2. **Недостаточный контроль доступа.** Действия аутентифицированного пользователя не контролируются должным образом. Используя эту уязвимость, злоумышленник может получать доступ к другим учетным записям, конфиденциальной информации. Признаками уязвимости механизма контроля доступа являются, например, возможность пользователей читать и модифицировать конфигурационные файлы, создание объектов, права доступа к которым определяются пользователем.
3. **Некорректная интеграция компонент приложения.** Данная группа уязвимостей включает, например, некорректный интерфейс между приложением и используемыми криптографическими протоколами. Как следствие, возникают «люки» (backdoors), эксплуатируя которые можно получить доступ к данным, циркулирующим между компонентами, обойти процедуры контроля доступа или дополнительной аутентификации.
4. **Слабые пароли.** В эту категорию попадают такие случаи, как использование коротких паролей, паролей которые легко угадать, слишком большой период жизни пароля.
5. **Передача конфиденциальной информации в открытом виде.** Такой информацией может быть, например, пароль пользователя.
6. **Недостаточный контроль входных данных. Недостаточная проверка параметров.** Компоненты приложения (серверные компоненты, библиотеки, драйвера и т. п.) написанные на языках C, C++ и других, не относящихся к управляемым языкам программирования, могут содержать вызовы встроенных функций, не контролирующих получаемые параметры. Использование таких функций делает возможным организацию атак, имеющих общее название «переполнение буфера». Если клиентские запросы не проверяются должным образом, пользователь может сформировать параметры запроса таким образом, что становится возможным доступ к программным компонентам инфраструктуры компьютерной системы. Данная группа уязвимостей позволяет организовать различные виды атак. Одной из таких атак является «межсайтовый скриптинг» - XSS (cross site scripting). Другая разновидность атак становится возможной, если в передаваемые параметры можно включать метасимволы. Например, символы «**!**» или «|» могут быть использованы для выполнения произвольного кода. Еще одной разновидностью атак является организация «инъекций», например SQL-injection, если пользовательские запросы передаются базе данных. Признаками уязвимости для SQL-injection являются:

   - отсутствие проверки введенной пользователем информации, которая используется для построения запросов или как параметры для хранимых процедур;

   - использование конкатенации или замены строк при построении запросов.
7. **Ссылка на путевые имена (pathnames), отсутствующие на сервере.** Такие ссылки позволяют получить доступ к родительскому каталогу, либо даже к корневому каталогу файловой системы.
8. **Неадекватная или недостаточная обработка ошибок и исключительных ситуаций.** Если атакующий сможет создать исключительную ситуацию, которая не обрабатывается приложением, он может получить доступ к детальной системной информации, поскольку операционная система

выведет полную причину возникновения такой ситуации, например, код запроса, который вызвал ошибку.

9. **Наличие люков, оставленных разработчиком.** Прежде чем приложение может быть использовано для обработки конфиденциальной информации, оно должно быть тщательно отлажено, протестировано, и из него должен быть удален весь лишний код. Например, должны быть удалены учетные записи, используемые для отладки, отладочные пароли, используемые для отладки вспомогательные конструкции, флаги и т. п. Такие «люки» зачастую позволяют получить привилегированный доступ к приложению.
10. **Перехват пароля и его повторное использование.** Такая атака становится возможной, если пароль передается в открытом виде.
11. **Недостаточный контроль вводимых пользователем URL.** Данная уязвимость не возникает, если пользователю разрешается переход только по тем ссылкам, которые указаны на Web-странице.
12. **Некорректное взаимодействие с инфраструктурой безопасности компьютерной системы.** Если интерфейс взаимодействия приложения с инфраструктурой безопасности (например, использование PKI) организован некорректно, у атакующего возникает возможность скомпрометировать как приложение, так и инфраструктуру.
13. **Недостаточная регистрация действий пользователя.** Журнал регистрации действий пользователя является важным источником определения попыток реализации угроз. Приложение должно регистрировать все действия, связанные с безопасностью, а также обеспечивать целостность регистрационной информации.
14. **Присутствие неиспользуемых системных вызовов, процессов, библиотек, структур данных и т. п**. Если атакующий обнаружит в приложении наличие таких объектов, он может попытаться с их помощью заставить приложение выполняться непредусмотренным образом.
15. **Предоставление повышенных или административных полномочий.** Уязвимость возникает, если пользователю или прикладному процессу предоставляются полномочия, превышающие те, которые необходимы для выполнения функциональных задач. Возможны два источника возникновения уязвимости: а) предоставление исключительных полномочий пользователю при его авторизации, б) предоставление таких полномочий прикладному процессу, которые превышают полномочия пользователя, инициирующего данный процесс. Признаками наличия данной уязвимости являются:
    - выполнение приложения с правами Administrator, System, root и правами других привилегированных учетных записей;
    - полный набор прав доступа приложения к используемым ресурсам.
16. **Конфликты вследствие конкуренции за ресурсы**. Уязвимость возникает, если различные процессы приложения пытаются одновременно получить доступ к одному и тому же ресурсу, например, запись в одну и ту же область памяти.
17. **Усложненная структура кода.**
    Признаками усложненности кода являются:
    - присутствие многочисленных команд перехода (GoTo) и рекурсии, что затрудняет трассировку;
    - наличие больших многофункциональных модулей (вместо простых модулей с небольшим количеством функций);
    - наличие процессов с несколькими точками инициализации и завершения;
    - наличие большого количества сложных функций и функций, не отвечающих непосредственно за функционал приложения.

    Усложненность затрудняет процесс анализа кода с целью обнаружения уязвимостей и, в конечном итоге, сертификации приложения.
18. **Несоблюдение требований безопасности при инсталляции и конфигурировании приложения**. При инсталляции приложения необходимо строго ограничивать доступ к папкам и файлам, содержащим исполняемый код, прикладные данные, файлы, используемые приложением. Приложение должно быть сконфигурировано таким образом, чтобы закрыть все «дыры», оставленные разработчиками (отладочные учетные записи и функции), изменены все пароли «по умолчанию».
19. **Использование небезопасных системных вызовов, библиотек, служб и т. п.** Существует обширный перечень известных уязвимостей, присутствующих в системных вызовах, программных библиотеках и т. п. Например, использование HTTP вместо HTTPS при передаче

конфиденциальной информации, использование некоторых функций C/C++, уязвимых к атакам переполнения буфера.

20. **Недостаточная защита от злоумышленного кода.** Уязвимость возникает, если хост, на котором функционирует приложение и окружающая инфраструктура, не подвергаются тщательной проверке на наличие вирусов и другого злоумышленного кода.
21. **Недостаточная защита данных от подмены в процессе передачи.** Уязвимость возникает, если приложение не использует механизм контроля целостности передаваемых данных, например, в виде ЭЦП или хеш-функций.
22. **Использование нескольких языков программирования. Использование языков, не обеспечивающих безопасность приложения.** К таким языкам относятся в частности Pre-Hypertext Processor (PHP) и Microsoft Visual Basic Scripting Edition (VBScript).

Литература: 1. S. Weber, P. Karger, A. Paradkar A Software Flaw Taxonomy: Aiming Tools At Security. In: SESS 2005. ACM Software Engineering for Secure Systems. 2004 2. M. Bishop. Computer Security: Art and Science. Addison-Wesley, December 2002. 3. S. Christey. PLOVER—Preliminary List of Vulnerability Examples for Researchers. Draft, August 2005. **http://cve.mitre.org/docs/plover/**. *4. CVE – Common Vulnerabilities and Exposures.* **http://www.cve.mitre.org/**. *5. G. Hoglund and G. McGraw. Exploiting Software: How to Break Code. Addison- Wesley, February 2004. 6. C. E. Landwehr, A. R. Bull, J. P. McDermott, W. S. Choi. A Taxonomy of Computer Program Security Flaws, with Examples. ACM Computing Surveys, Vol. 26, No. 3, September 1994, pp. 211-254.*