МОДЕЛЬ ОБНАРУЖЕНИЯ УЯЗВИМОСТЕЙ В WEB-ПРИЛОЖЕНИЯХ, ИСПОЛЬЗУЕМЫХ В ДИСТАНЦИОННОМ ОБУЧЕНИИ

Д.Е. Оношко[1], В.В. Бахтизин[2]

[1] Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь, onoshko@bsuir.by
[2] Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь, bww@bsuir.by

Abstract. A web-application SQL-injection vulnerability detection model based on the static analysis of source code for applications used in distance learning is given. Evaluation system based on abstract interpretation of the application's source code is described. Ways of extending the vulnerability detection model for specific use cases and its generalization for other types of vulnerabilities are shown.

Одной из важнейших задач в организации дистанционного обучения является организация эффективного взаимодействия между участниками этого процесса — преподавателями и студентами. Помимо административных вопросов существенную роль играет техническая база, в том числе программное обеспечение, используемое в организации документооборота, сопутствующего учебному процессу.

Программные средства такого рода нередко реализуются в виде web-приложений, что обусловлено необходимостью обеспечить доступ к этим программным средствам для пользователей, располагающихся на значительном удалении друг от друга и от учебного заведения, обеспечивающего дистанционное обучение. Однако возможность получить доступ к web-приложению из любой точки земного шара влечёт за собой необходимость контроля его качества, и особенно — наличия в нём уязвимостей в различного рода атакам.

По данным Открытого проекта обеспечения безопасности web-приложений (OWASP) по состоянию на 2013 год наиболее распространённым видом угрозы для различных типов приложений, в т.ч. и web-приложений, являются SQL-инъекции [1]. Как показывает анализ свойств этого вида угроз, главной причиной уязвимости к ним web-приложений является некорректная обработка данных, поступающих в приложение извне: отсутствие или недостаточность фильтрации таких данных позволяет злоумышленнику сформировать запрос к web-приложению таким образом, чтобы web-приложение выполнило действия, не предусмотренные разработчиком.

Обнаружение и исправление ошибок такого рода является трудоёмкой задачей, требующей от специалиста не только глубокого понимания механизмов эксплуатации подобных уязвимостей, но и концентрации внимания, а также хорошего знания архитектуры анализируемого web-приложения. При этом любое изменение, вносимое в код web-приложения в ходе разработки или сопровождения, требует повторного его анализа на наличие подобных уязвимостей, поскольку такие изменения могут сделать доступными для использования злоумышленником ранее не проявлявшиеся ошибки. Таким образом, целесообразно автоматизировать такой анализ, сократив тем самым издержки на контроль качества web-приложения.

Для автоматизации анализа web-приложений на наличие уязвимостей предлагается использовать модель обнаружения уязвимостей, основанную на статическом анализе исходных кодов путём абстрактной интерпретации.

В рамках модели обнаружения уязвимостей предлагается рассматривать исходные коды web-приложения как совокупность процедур $Q = \{P_1, P_2, \ldots P_N\}$, где N — общее количество процедур, составляющих web-приложений. Программный интерфейс процедур предлагается описывать как множество параметров двух видов:

Библиотека БГУИР

– *in-параметры* — для описания данных, передаваемых в процедуру;
– *out-параметры* — для описания данных, возвращаемых процедурой.

Помимо параметров процедур в модели также рассматриваются данные, передаваемые в качестве параметров (переменные, константы). Параметрам и данным назначаются оценки, которые в простейшем случае могут иметь бинарный характер.

Для переменных используется следующая система оценок:

– оценку S получают переменные, подстановка значений которых в SQL-запрос *не нарушит* логики, предусмотренной разработчиками web-приложения.

– оценку U получают переменные, подстановка значений которых в SQL-запрос *может нарушить* логику, предусмотренную разработчиками web-приложения.

В такой системе оценок наилучшей считается оценка S, наихудшей — U.

Оценка для in-параметра определяется как наихудшая оценка, которую могут иметь данные, фактически передаваемые в качестве данного параметра:

– оценку S получают in-параметры, в качестве которых должны передаваться данные с оценкой не ниже S.

– оценку U получают in-параметры, в качестве которых должны передаваться данные с оценкой не ниже U (т.е. могут передаваться любые данные).

Оценка для out-параметра определяется как наихудшая из оценок данных, которые могут возвращаться через данный out-параметр.

Пусть на i-м шаге имеются результаты анализа процедур из подмножества $Q_i = \{P_1, P_2, \ldots P_{C(i)}\}$, где $C(i) \le N$ — количество таких процедур (зависит от количества стандартных процедур с заранее известными оценками параметров). Тогда, поскольку все процедуры принадлежат одному и тому же web-приложения, найдётся процедура $P_{C(i)+1}$, зависящая только от процедур из множества Q_i. Анализируя её операторы, можно получить оценки для всех её параметров. Последней будет проанализирована процедура P_N — главная процедура web-приложения. Наличие среди in-параметров P_N хотя бы одного, имеющего оценку выше U, говорит о наличии в web-приложении уязвимости. Отслеживая путь данных, поступающих через этот параметр, можно определить причину уязвимости и исправить допущенную разработчиком ошибку.

Предлагаемая модель обнаружения уязвимостей позволяет автоматизировать трудоёмкий процесс анализа исходных кодов web-приложений на наличие уязвимостей к SQL-инъекциям, а результаты, получаемые в ходе такого анализа, могут быть использованы для оценки качества web-приложения и принятия административных решений, направленных на обеспечение надлежащего уровня качества.

Литература

1. OWASP Top 10-2013. The Ten Most Critical Web Application Security Risks. [Электронный ресурс]. — Режим доступа: http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202013.pdf. — Дата доступа: 31.10.2013

Библиотека БГУИР