TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN

———— * ————

ĐỒ ÁN
TỐT NGHIỆP ĐẠI HỌC

NGÀNH CÔNG NGHỆ THÔNG TIN

PHÁT HIỆN LỖ HỔNG AN NINH TRÊN CÁC ỨNG DỤNG WEB

Sinh viên thực hiện : **Bùi Duy Hùng**
Lớp CNPM A - K49
Giáo viên hướng dẫn: TS **Nguyễn Khanh Văn**

HÀ NỘI 6-2009

PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

1. Thông tin về sinh viên
Họ và tên sinh viên: Bùi Duy Hùng
Điện thoại liên lạc: 01697266076 Email: duyhung19@gmail.com
Lớp: CNPM A – K49 Hệ đào tạo: Đại học chính quy
Đồ án tốt nghiệp được thực hiện tại: Trường ĐH Bách Khoa – Hà Nội
Thời gian làm ĐATN: Từ ngày 18 /02 /2009 đến 05 /06 /2009

2. Mục đích nội dung của ĐATN

Nghiên cứu, tìm hiểu và cài đặt thuật toán phát hiện lỗ hổng an ninh trên các ứng dụng web.

3. Các nhiệm vụ cụ thể của ĐATN

- Tìm hiểu và cài đặt thuật toán ngăn chặn tấn công SQL Injection tại thời điểm thực thi (thuật toán Runtime SQLCheck)
- Tìm hiểu thuật toán phát hiện lỗ hổng SQL Injection trong mã nguồn ứng dụng (thuật toán Static SQLCheck)
- Xây dựng hệ thống website để kiểm tra độ chính xác của công cụ cài đặt thuật toán Runtime SQLCheck

4. Lời cam đoan của sinh viên:
Tôi *Bùi Duy Hùng* cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *TS Nguyễn Khanh Văn*.
Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

Hà Nội, ngày 25 tháng 05 năm 2009
Tác giả ĐATN

Bùi Duy Hùng

5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ:

Hà Nội, ngày tháng năm 2009
Giáo viên hướng dẫn

TS Nguyễn Khanh Văn

TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP

Sự tăng trưởng nhanh chóng của Internet đã tạo ra các dịch vụ web rất hữu ích, dần dần thay thế các giao dịch thủ công truyền thống. Điển hình như, ngày nay ta đã có thể ngồi nhà mà vẫn có thể thực hiện dịch vụ như kiểm tra tài khoản ngân hàng, đặt vé máy bay, mua sắm … Nhưng hầu hết các ứng dụng này đều chứa những lỗi bảo mật tiềm ẩn mà các tin tặc có thể khai thác và thực hiện tấn công. Kết quả của việc tấn công là tính cơ mật (tính riêng tư) và toàn vẹn của thông tin bị xâm phạm. Một trong những lỗi nguy cơ bảo mật tiềm ẩn phát triển nhanh nhất trong những năm gần đây xuất phát từ những sai sót trong việc kiểm tra tính hợp lệ của chuỗi đầu vào được cung cấp bởi người dùng, được gọi là tấn công dựa trên khai thác lỗ hổng SQL Injection. Nhưng nguyên nhân sâu xa hơn là nó lại bắt nguồn từ trong tầng lõi của ứng dụng, bao gồm một hệ thống nhận, chuyển đổi và xây dựng các giá trị chuỗi (một số giá trị chuỗi lại được cung cấp từ những nguồn không tin cậy), và trình diễn những giá trị đó tới hệ thống khác như các chương trình hay các đoạn chương trình. Do đó cần phải có những kỹ thuật mới để có thể hạn chế những nguy cơ bảo mật này.

Đồ án này mô tả các kĩ thuật tấn công website dựa trên việc khác thác lỗi SQL Injection tiên tiến nhất hiện nay. Sau đó đồ án cung cấp những nguyên lý đầu tiên dựa trên các khái niệm từ các ngôn ngữ lập trình và trình biên dịch cho các lỗi bảo mật tiềm ẩn với định nghĩa hình thức cho tấn công SQL Injection. Dựa vào các mô tả này, đồ án mô tả và phân tích hai thuật toán: "bảo vệ tại thời điểm thực thi"(runtime protection) và "phân tích tĩnh"(static analysis). Hai thuật toán này đều có mục đích chung là chỉ ra những lỗi bảo mật trong mã của ứng dụng và ngăn cản các tin tặc khai thác chúng. Song mỗi thuật toán lại có một vai trò riêng trong vòng đời phát triển phần mềm. Thuật toán "bảo vệ tại thời điểm thực thi" (Runtime SQLCheck) ngăn cản hiệu quả SQL Injection trong các sản phẩm phần mềm đã được triển khai. Trong khi đó thuật toán "phân tích tĩnh" (Static SQLCheck) lại cần thiết trong suốt quá trình triển khai phần mềm và kiểm thử.

Phần đóng góp thực tế của đồ án là thực hiện cài đặt thực tế cho các thuật toán trên (Runtime SQLCheck và Static SQLCheck) cùng với những kết quả thử nghiệm trên ứng dụng web thực tế (website bán sách trực tuyến).

ABSTRACT OF THESIS

Together with the rapid growth of the Internet, many online services were born and became necessary in our daily life, these services are gradually replacing our manual transaction. For instance, we can check our bank account, book ticket, shopping even when we are at home. But most of these web applications contain many potential security vulnerabilities that hackers can easily exploit and attack. As a result, the integrity of confidential information is damaged. In recent years, one of the fastest developing potential security vulnerabilities is about checking the valid input sequence provided by users, where the so-called SQL injection attack exploits. But the primary cause is that the error is derived from the core of the applications, including receiving, conversion and construction of the sequence value (a string value supplied from an untrusted sources), and show that value to other systems such as program or modules. Therefore it is necessary to have a new technology to limit this security risk.

The thesis describes in detail website attack techniques based on the latest SQL Injection exploitation. Then, it provides the basic principles based on programming languages concepts and compiler for potential security vulnerabilities, and formal definition for SQL Injection attacks. Based on this description, the thesis describes and analyzes two algorithms: "runtime protection" and" static analysis" . These two algorithms have the same purpose which is to find out the security vulnerabilities in the application and prevent the hackers from exploiting them. However, each algorithm has a particular role within the software development cycle. The algorithms "runtime protection" (Runtime SQLCheck) effectively prevent SQL Injection in the software when it was deployed, while algorithms "static analysis" (Static SQLCheck) is needed during software implementation and testing.

Finally, the most important contribution of the thesis is practical implementation of these algorithms (Runtime SQLCheck and Static SQLCheck) together with the implementation result on actual website – Online Booksale Website.

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG

DANH MỤC CÁC THUẬT NGỮ

Thuật ngữ	Ý nghĩa
Web browser	Trình duyệt web
Web Application Server	Máy chủ ứng dụng web
Database Server	Máy chủ cơ sở dữ liệu
Meta-Character	Siêu ký tự: Là một chuỗi tổ hợp các ký tự chữ cái được sinh ra ngẫu nhiên sử dụng để đánh dấu đầu và cuối mỗi chuỗi đầu vào
Augmented Input	Đầu vào gia tăng: là chuỗi đầu vào khi đã được thêm các Meta-Character vào đầu và cuối chuỗi.
Augmented Query	Câu truy vấn gia tăng: là câu truy vấn được xây dựng từ các chuỗi hằng và đầu vào gia tăng.
Augmented Grammar	Văn phạm gia tăng: là bộ văn phạm xây dựng cho các câu truy vấn gia tăng.
Parse tree	Cây phân tích ngữ pháp
Context free grammar	Văn phạm phi ngữ cảnh
Tranducer	Bộ chuyển đổi
Finite state tranducer	Bộ chuyển đổi trạng thái hữu hạn

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Viết đầy đủ	Ý nghĩa
HTML	Hypertext Markup Language	Ngôn ngữ đánh dấu siêu văn bản
HTTP	Hypertext Tranfer Protocol	Giao thức truyền tải siêu văn bản
SQLCIA	SQL Command Injection Attack	Tấn công SQL Injection
LALR	Look Ahead Left-to-Right	Thuật toán đọc từ trên xuống theo chiều từ trái sang phải
CFG	Context Free Grammar	Văn phạm phi ngữ cảnh
SQLCIV	SQL Command Injection Vulnerability	Lỗ hổng SQL Injection

LỜI CẢM ƠN

Trước hết, em xin được chân thành gửi lời cảm ơn sâu sắc tới các thầy cô giáo trong trường Đại học Bách Khoa Hà Nội nói chung và các thầy cô trong khoa Công nghệ Thông tin, bộ môn Công nghệ phần mềm nói riêng đã tận tình giảng dạy, truyền đạt cho em những kiến thức, những kinh nghiệm quý báu trong suốt quá trình học tập và rèn luyện tại trường Đại học Bách Khoa Hà Nội.

Em xin được gửi lời cảm ơn đến thầy Nguyễn Khanh Văn – Trưởng bộ môn, Giảng viên bộ môn Công nghệ phần mềm, khoa Công nghệ Thông tin, trường Đại học Bách Khoa Hà Nội đã hết lòng giúp đỡ, hướng dẫn và chỉ dạy tận tình em trong quá trình làm đồ án tốt nghiệp.

Cuối cùng, em xin được gửi lời cảm ơn chân thành tới gia đình, bạn bè đã động viên, đóng góp ý kiến và giúp đỡ trong quá trình học tập, nghiên cứu và hoàn thành thực tập tốt nghiệp.

Hà Nội, ngày 25 tháng 05 năm 2009

BÙI DUY HÙNG

Sinh viên lớp **Công nghệ phần mềm A – K49**

Khoa **Công nghệ Thông tin - Đại học Bách Khoa Hà Nội**

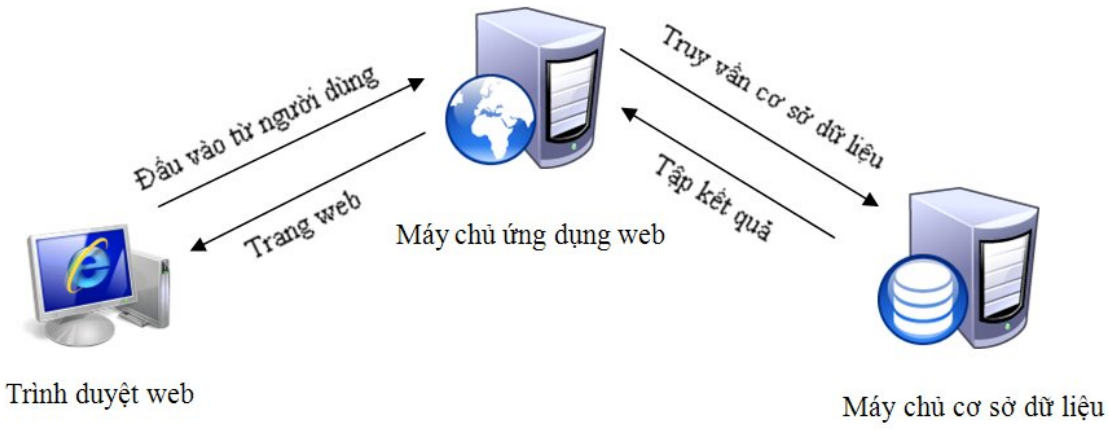
CHƯƠNG 1. GIỚI THIỆU BÀI TOÁN

An toàn và bảo mật thông tin, đặc biệt là cho dữ liệu máy tính, không chỉ là sự quan tâm của tất cả các doanh nghiệp, dịch vụ tài chính hay chính phủ mà còn là vấn đề mà bất kỳ người sử dụng máy tính nào cũng phải chú ý và thường xuyên đối mặt. Từ người dùng máy tính để bàn bình thường trong các phòng nghiên cứu đến các quản trị viên đang làm việc cho công ty lớn, tất cả đều cần hiểu biết nhất định về hệ thống thông tin và máy tính. Vấn đề này càng được khẳng định hơn khi sự phát triển của Internet đã làm cho lợi ích và tài sản của người dùng trở lên gắn liền với các ứng dụng web. Trong chương đầu tiên của đồ án, người viết xin trình bày đôi nét hoàn cảnh chung dẫn đến nhu cầu cần giải quyết bài toán "Phát hiện lỗ hổng an ninh trên các ứng dụng web"; sau đó xin giới thiệu các công việc đã hoàn thành cùng các ý tưởng thực hiện. Cuối cùng người viết xin cung cấp vắn tắt bố cục của đồ án.

1.1. CÁC ỨNG DỤNG WEB

Sự ra đời của Internet đã đánh dấu một bước nhảy vọt trong sự phát triển của ngành công nghệ thông tin. Sự phát triển nhanh của Internet đã tạo một cuộc cách mạng thục thụ. Internet nhanh chóng xâm nhập vào các hoạt động thông tin, kinh tế - xã hội cũng như các hoạt động vui chơi giải trí trực tuyến khác như game online… Dần dần Internet đã trở thành một phần không thể thiếu được trong cuộc sống của chúng ta ngày nay. Sự phát triển muôn mặt của Internet được thể hiện rõ nhất ở các ứng dụng web đa dạng, thâm nhập vào các mặt của đời sống. Điển hình các ứng dụng web được sử dụng để có thể giúp đỡ các hành khách có thể đặt và mua vé máy bay qua mạng, thực hiện mua bán và thanh toán trực tuyến hay chúng còn được sử dụng làm nơi để các cộng đồng dân cư trên mạng có thể trao đổi, giao lưu và chia sẻ các thông tin của mình (blog, các website cá nhân). Chỉ với một chiếc máy tính, một thiết bị cầm tay … có cài đặt trình duyệt web thì việc truy cập vào web và các dịch vụ có thể thực hiện ở bất cứ nơi nào.

Các dữ liệu mà ứng dụng web nắm giữ như số thẻ tín dụng, số bảo hiểm xã hội… có ý nghĩa đặc biệt quan trong đối với cả người dùng và những nhà cung cấp dịch vụ.). Sự xuất hiện mô hình tương tác Business-to-Business (B2B) và Business-to-Consumer (B2C) thì việc trao đổi thông tin chính xác và bảo mật trở nên cần thiết hơn lúc nào hết.



Hình 1.1: Kiến trúc hệ thống của ứng dụng web

Hình 1.1 đưa ra một kiến trúc hệ thống cho các ứng dụng web ngày nay. Kiến trúc này bao gồm ba tầng: Web Browser (trình duyệt web), đóng vai trò là giao diện người dùng; Web Application Server (máy chủ ứng dụng web) có chức năng quản lý logic nghiệp vụ (business logic); và Database Server quản lý các dữ liệu được lưu trữ. Web Application Server nhận đầu vào dưới dạng chuỗi từ hai tầng còn lại: đầu vào do người dùng cung cấp từ trình duyệt và tập kết quả từ cơ sở dữ liệu. Nó chuyển những đầu vào này thành đầu ra dưới dạng chuỗi cho các tầng khác: các câu truy vấn cho Database Server và các văn bản HTML cho trình duyệt máy khách. Web Application Server xây dựng mã "động", vì thế mà mã cho toàn bộ ứng dụng web không tồn tại ở bất cứ nơi nào tại bất kỳ một thời điểm nào cho bất kỳ một thực thể nào điều chỉnh. Luồng dữ liệu giữa các tầng làm nảy sinh vấn đề kiểm tra đầu vào (input validation) cho các Web Application Server: nó phải kiểm tra và/hoặc chỉnh sửa các chuỗi đầu vào trước khi tiếp tục xử lý chúng hay biến đổi chúng thành đầu ra để đưa đến tầng khác thực thi. Lỗi kiểm tra hay các đầu vào chưa được xử lý có thể gây ra những vấn đề bảo mật của ứng dụng web.

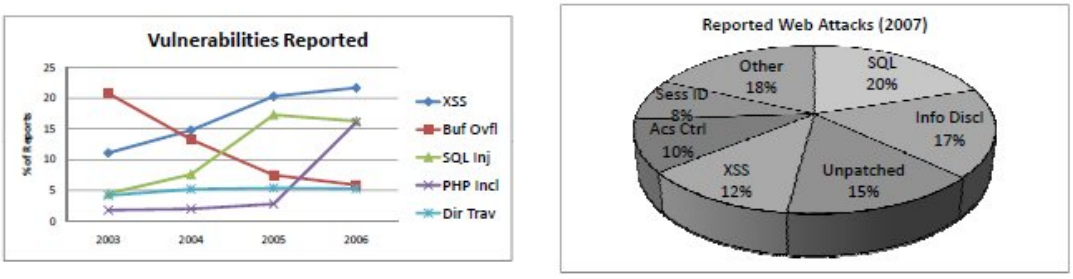
1.2. NHỮNG NGUY CƠ TIỀM ẨN TỪ VIỆC KIỂM TRA DỮ LIỆU ĐẦU VÀO

Song song với sự phát triển thì các ứng dụng web đang phải đối đầu với những nguy hiểm phá hoại tiềm ẩn. Sự phá hoại ở đây không phải là sự xóa bỏ ứng dụng web khỏi cuộc sống số mà sự phá hoại ở đây là việc ăn trộm, thay đổi, giả mạo và xóa bỏ thông tin trong các ứng dụng web. Chính từ những phá hoại này mà lợi ích của người dùng, các doanh nghiệp, chính phủ… đang đứng trước những nguy cơ mất mát và vi phạm khôn lường. Hai mối họa đáng quan tâm nhất là Cross-site scripting (XSS) và SQL Injection. Đây là hai loại lỗi phổ biến dựa trên những thiếu

sót từ khâu kiểm tra đầu vào trong các ứng dụng web. Các tin tặc thường sử dụng những lỗi này (XSS và SQL Injection) để làm cho Web Application Server sinh ra các văn bản HTML và các câu truy vấn cơ sở dữ liệu mà người lập trình viên không có ý định nhắm tới hay nói cách khác là tin tặc sử dụng những lỗi này để tấn công ứng dụng web.

Những việc tấn công này được gọi là tấn công ở mức ứng dụng. Chúng không thể bị ngăn cản bởi các tường lửa kiểm duyệt các gói tin (phân tích chữ ký các gói IP riêng lẻ và chỉ định các cổng cụ thể). Tấn công ở tầng ứng dụng khác với tấn công ở tầng mạng.

Tấn công ở tầng ứng dụng khai thác các lỗi bảo mật trong mã ứng dụng web và các hạn chế của giao thức HTTP. Tấn công ở tầng ứng dụng không thể bị chặn bởi các tường lửa và các phần mềm diệt virus. Các tường lửa mạng để mở cổng 80 cho web server. Các ứng dụng web giao tiếp với người sử dụng thông qua cổng này. Nếu một kẻ tấn công có thể truy cập ứng dụng thì hắn có thể tấn công ứng dụng mà không bị tường lửa ngăn chặn. Cho ví dụ, một người có tài khoản hợp lệ tại một hệ thống ngân hàng. Người này kết nối tới tài khoản cá nhân của mình bằng cách xác thực và thiết lập một phiên hợp lệ. Nếu người này chèn thêm một đoạn mã để có thể truy cập trái phép thông tin của các người dùng khác thì tường lửa mạng hay Instrusion Detection Systems(IDSs) sẽ không thể chặn được.



(a) (b)

Hình 1.2: Số lần tấn công và các nguy cơ tiềm ẩn hiện nay

Hình 1.2a cho ta thấy tỷ lệ phần trăm những nguy cơ bảo mật trong những năm từ 2003 – 2006: XSS, SQL Injection, PHP file inclusions, buffer overflows (tràn bộ đệm), và directory traversals (duyệt thư mục). Đây là top năm nguy cơ bảo mật được báo cáo trong năm 2006. Lưu ý rằng trong năm 2006 thì tỉ lệ XSS và SQL Injection là khá gần nhau: 21.7% và 16%. Còn trong Hình 1.2b cho chúng ta xem báo cáo thống kê các vụ tấn công website trong năm 2007. Mặc dù có nhiều vụ tấn công chưa được báo cáo hay chưa được phát hiện thì biểu đồ này cũng chỉ ra các con số 12% và 20% của các vụ tấn công web được thực hiện trong năm 2007 là

XSS và SQL Injection tương ứng. Còn theo thống kê mới nhất thì trong năm 2008 các website nạn nhân bị tấn công bằng SQL Injection là gần 500.000 – một con số đáng để chúng ta lưu tâm. Qua những thống kê trên thì chúng ta có thể thấy là SQL Injection ngày càng trở thành phương thức tấn công khá ưu thích đối với các tin tặc.

Tấn công SQL Injection là một trong các cách tấn công mức ứng dụng rất phổ biến trong các ứng dụng web. Đây là một kiểu tấn công mà tin tặc lợi dụng lỗ hổng trong khâu kiểm tra dữ liệu nhập vào trong các ứng dụng web và các thông báo lỗi trong hệ quản trị cơ sở dữ liệu để "tiêm vào" (inject) và thi hành các câu truy vấn SQL không hợp lệ (các câu truy vấn mà người lập trình không mong muốn). Từ đó tin tặc có thể sử dụng các công cụ để thu thập thông tin từ một số bảng và cột nhất định trong cơ sở dữ liệu.

Bước kế tiếp trong việc tấn công cơ sở dữ liệu là tin tặc sẽ chèn them các mã lệnh điều khiển máy chủ cơ sở dữ liệu tải về các phần mềm khác trên Internet giúp chúng có được quyền kiểm soát cao hơn đối với mục tiêu.

SQL Injection là kiểu tấn công có mục tiêu rất cụ thể và thường là mục tiêu đơn lẻ cho mỗi một vụ tấn công. Chính vì thế mà những vụ tấn công như thế này thường không gây được sự chú ý rộng rãi như virus hay sâu máy tính.

Âm thầm như thế nhưng thiệt hại của những vụ tấn công này lại rất lớn. Nếu như một máy chủ cơ sở dữ liệu bị tin tặc chiếm quyền kiểm soát thì sẽ có một khối lượng rất lớn thông tin cá nhân tài chính của người dùng sẽ rơi vào tay chúng. Và nếu thành công thì có thể nói nguồn thông tin mà tin tặc thu được còn nhiều hơn rất nhiều so với tấn công phishing. Tin tặc không phải mất công giả mạo để lừa người sử dụng cung cấp thông tin cá nhân tài chính. Tỉ lệ thành công của các vụ tấn công SQL Injection thường rất cao.

Do đó vấn đề đặt ra ở đây là chúng ta cần xem xét lại một cách nghiêm túc tính bảo mật của hệ thống của mình trước khi trở thành nạn nhân của SQL Injection. Xong nhiều khi chúng ta không thể kiểm soát được hết tất cả các nguy cơ về lỗi SQL Injection tiềm ẩn trong mã ứng dụng. Đã có nhiều phương pháp khắc phục SQL Injection khá phổ biến được đưa ra như:

- ✓ Kiểm tra dữ liệu đầu vào, dùng các biểu thức chính quy
- ✓ Sử dụng các thư viện Prepared trong Java hay SqlParameter trong .Net
- ✓ Sử dụng Stored Procedure

Nhưng những phương pháp vẫn còn có những nhược điểm. Để phòng chống SQL Injection người dùng thường lựa chọn phương pháp kiểm tra dữ liệu đầu vào bằng cách dùng những biểu thức chính quỹ. Nhưng nhiều khi họ lại bỏ qua việc

kiểm tra này hoặc nếu có kiểm tra thì các biểu thức chính quy thường vẫn còn sai sót, dẫn tới ứng dụng. Nhưng có một số khác thì lại lựa chọn các thư viện Prepared (Java) hay SqlParameter (.NET) để phòng chống SQL Injection. Các thư viện này ngăn cản SQL Injection khá hiệu quả khi nó ràng buộc kiểu của các tham số được truyền vào trong câu truy vấn. Song đối với trường hợp khi ứng dụng thực hiện tìm kiếm theo nhiều trường phức tạp, phong phú thì những thư viện này lại không thể dùng được. Và hầu hết nhiều người cho rằng Stored Procedure là an toàn tuyệt đối, nhưng sự thực điều này chỉ đúng một nửa. Vì, ví dụ, trong SQL Server cung cấp cho người quản trị cơ sở dữ liệu hàm EXEC có thể thực thi câu truy vấn được xây dựng động ở dạng chuỗi. Việc sử dụng rộng rãi hàm này khiến cho Stored Procedure không được ở mức an toàn tuyệt đối nữa.

Trước bối cảnh như vậy, nhu cầu cấp thiết đặt ra trước mắt chúng ta là cần phải có một công cụ có thể phát hiện và ngăn chặn được SQL Injection hiệu quả.

1.3. MỘT SỐ PHƯƠNG PHÁP PHÒNG CHỐNG SQL INJECTION PHỔ BIẾN

Thường thường thì người lập trình viên thực hiện việc kiểm tra đầu bằng cách xử lý từng đầu vào một cách độc lập. Phương pháp này lại để lại cho ta hai khả năng gây ra lỗi: việc kiểm tra có thể bị bỏ qua, và việc kiểm tra có thể không đúng. Hầu hết các ngôn ngữ lập trình web đều cho truyền các đầu vào không tin cậy tới máy khách hoặc cơ sở dữ liệu. Không có gì trong các ngôn ngữ lập trình web, trình biên dịch hay các hệ thống thực thi cảnh báo cho người lập trình viên biết rằng việc kiểm tra bị bỏ qua. Do đó, việc phân tích luồng thông tin động và tĩnh là cần thiết để đảm bảo rằng tất cả các đầu vào không tin cậy được kiểm tra.

Các thủ tục kiểm tra có thể có lỗi. Đã có rất nhiều kỹ thuật kiểm tra đầu vào được đưa ra như: giới hạn độ dài của chuỗi đầu vào hay phổ biến hơn là sử dụng các biểu thức chính quy để lọc đầu vào. Một phương pháp thay thế cho việc ngăn cản SQL Injection là thay thế các đầu vào, có thể bằng cách thêm các dấu sổ (\ ) vào đằng trước các dấu nháy đơn. Tất cả các kỹ thuật này chỉ có thể là một sử cải thiện của các đầu vào chưa được điều chỉnh, nhưng tất cả chúng đều có những điểm yếu. Không có gì trong chúng có thể đảm bảo cấu trúc tĩnh của các câu truy vấn hay các trang web được sinh ra từ chối hoàn toàn các đầu vào "tồi" (bad input), cho ví dụ, các bộ lọc biểu thức chính quy vẫn còn có những hạn chế. Do đó một trình phân tích chuỗi cần thiết phải có để đảm bảo các tấn công SQL Injection không thể xảy ra.

Việc phát hiện ra những nơi có thể bị tấn công SQL Injection trong mã ứng dụng (qua việc phân tích chuỗi hay nói khác đi là phân tích mã nguồn) sẽ trở nên vô nghĩa nếu như chúng ta không có một thủ tục đúng nghĩa để ngăn chặn nó. Vì biết trước được tại đó có thể có tấn công SQL Injection mà không có được biện pháp nào ngăn chặn thì công việc lập trình của người lập trình viên sẽ đi vào bế tắc. Vậy nên đòi hỏi cần phải có một phương pháp ngăn chặn SQL Injection tại thời điểm thực thi (ứng dụng đang chạy).

1.4. NHIỆM VỤ, KẾT QUẢ VÀ NGHIỆM THU

Trong đề tài đồ án tốt nghiệp lần này, người viết đồ án xin trình bày hai phương pháp Static SQLCheck và Runtime SQLCheck có thể phát hiện và ngăn chặn SQL Injection hiệu quả trong các ứng dụng web. Cách thức xây dựng của hai phương pháp này chủ yếu đều dựa trên văn phạm phi ngữ cảnh.

Static SQLCheck và Runtime SQLCheck hoàn toàn có thể hoạt động độc lập với nhau bởi vì phạm vi và tác dụng của chúng là khác nhau. Runtime SQLCheck có thể ngăn chặn được SQL Injection hiệu quả tại thời điểm thực thi (thời điểm ứng dụng đang chạy), nó từ chối mọi đầu vào do người dùng cung cấp gây ra tấn công SQL Injection. Trong khi đó thì Static SQLCheck lại có khả năng phát hiện được những đoạn mã trong ứng dụng có thể bị khai thác SQL Injection. Song hai sự kết hợp của hai phương pháp này sẽ tạo ra cho ta một hệ thống có khả năng phát hiện (Static SQLCheck) và ngăn chặn (Runtime SQLCheck) SQL Injection hiệu quả và mạnh mẽ.

Sau đây là một số kết quả mà người viết đã đạt được trong quá trình làm đồ án:

- ✓ Tìm hiểu và nắm bắt được cách thức xây dựng văn phạm phi ngữ cảnh
- ✓ Xây dựng thành công bộ văn phạm gia tăng cho thư viện RuntimeSQLCheck.
- ✓ Hiểu cách thức làm việc và sử dụng thành thạo bộ sinh trình phân tích ngữ pháp (parser generation) GOLD Parser.
- ✓ Nghiên cứu và nắm vững được các tư tưởng thuật toán của hai phương pháp Runtime SQLCheck và Static SQLCheck.
- ✓ Xây dựng thành công thư viện RuntimeSQLCheck (được viết trên hai ngôn ngữ .NET và Java) để ngăn chặn tấn công SQL Injection trên các ứng dụng web.
- ✓ Đã xây dựng được hệ thống website bán sách trực tuyến để thử nghiệm độ chính xác của thư viện RuntimeSQLCheck.

1.5. BỐ CỤC ĐỒ ÁN

Với các phương pháp đã trình bày và những công việc đã làm được phần còn lại của đồ án sẽ được tổ chức như sau:

Chương 2 sẽ trình bày về những kỹ thuật tấn công SQL Injection tiên tiến mà các tin tặc vẫn sử dụng để tấn công các ứng dụng web. Đây là tiền đề trước khi đưa các phương pháp phòng chống và ngăn chặn SQL Injection.

Chương 3 sẽ là những phân tích và đánh giá của người viết về phương pháp Runtime SQLCheck để ngăn chặn SQL Injection từ đầu vào do người dùng cung cấp.

Chương 4 sẽ trình bày về phương pháp Static SQLCheck. Cơ chế hoạt động của phương pháp này là phân tích mã nguồn ứng dụng web để tìm ra các đoạn mã có khả năng bị khai thác và tấn công SQL Injection.

Trong *Chương 5* người viết trình bày những thử nghiệm cài đặt thực tế của thuật toán Runtime SQLCheck, đó là thư viện RuntimeSQLCheck, kèm theo đó là cách thức sử dụng trong các ứng dụng web thực tế. Cuối cùng, từ những cài đặt thử nghiệm này, những đánh giá và kết luận được chính người viết rút ra từ kết quả kiểm thử của thư viện RuntimeSQLCheck trên hệ thống website bán sách trực tuyến do chính người viết phát triển, cũng được trình bày rõ trong phần cuối của chương.

Chương 6 Đánh giá và kết luận trình bày những so sánh và đánh giá tính hiệu quả và hữu dụng của hai phương pháp Runtime SQLCheck và Static SQLCheck. Hướng phát triển của đề tài trong tương lai cũng được đề xuất trong chương này.

Cuối cùng, phần *Phụ Lục* sẽ trình bày về kiến trúc và cách thức hoạt động của chương trình sinh bộ phân tích ngữ pháp: Gold Parser