doi:10.11959/j.issn.1000-436x.2017238

基于二阶分片重组盲注的渗透测试方法

乐德广[1,2,3]，龚声蓉[1]，吴少刚[3]，徐锋[3]，刘文生[4]

(1. 常熟理工学院计算机科学与工程学院，江苏 常熟 215500；2. 苏州大学计算机科学与技术学院，江苏 苏州 215006；
3. 中科梦兰电子科技有限公司，江苏 常熟 215500；4. 泉州市公安局公共信息网络安全监察支队，福建 泉州 362000)

摘 要：针对如何克服当前 SQL 注入渗透测试存在的盲目性，以生成优化的 SQL 注入攻击模式、增强渗透测试攻击生成阶段的有效性，提高对 SQL 注入渗透测试的准确度问题，提出一种基于二阶分片重组的 SQL 盲注漏洞渗透测试方法。该方法通过对 SQL 注入攻击行为进行建模，并以模型驱动渗透测试多形态和多种类的攻击生成，从而降低 SQL 注入渗透测试盲目性，提高其准确度。通过实际的 Web 应用 SQL 注入漏洞测试实验与比较分析，不仅验证了所提方法的有效性，而且通过减少在安全防御环境下对 SQL 注入漏洞检测的漏报，提高其测试的准确度。

关键词：SQL 注入；渗透测试；攻击模型；二阶分片重组

中图分类号：TP393 **文献标识码**：A

Penetration test method using blind SQL injection based on second-order fragment and reassembly

LE De-guang[1,2,3], GONG Sheng-rong[1], WU Shao-gang[3], XU Feng[3], LIU Wen-sheng[4]

(1. School of Computer Science & Engineering, Changshu Institute of Technology, Changshu 215500, China;
2. School of Computer Science and Technology, Soochow University, Suzhou 215006, China;
3. Lemote Electronic Technology Co., Ltd., Changshu 215500, China;
4. Public Information Network Safety Supervision Division, Quanzhou Municipal Public Security Bureau, Quanzhou 362000, China)

Abstract: How to get rid of the blindness of current SQL injection penetration test, produce the optimized attack pattern of SQL injection, enhance the effectiveness in the phase of attack generation, and improve the accuracy of vulnerability detection of SQL injection using penetration test, is a big challenge. In order to resolve these problems, a new penetration test method using blind SQL injection was proposed based on second-order fragment and reassembly. In this method, the SQL injection attack model was built firstly and then the multiform and multi-type attack patterns of SQL injection penetration test driven by the SQL injection attack model was produced, which can reduce the blindness of SQL injection penetration test and improve the accuracy of SQL injection vulnerability detection. The experiments of SQL injection vulnerability detection was conducted through the actual Web applications by using proposed method in comparison with current methods. The analysis results of test show the proposed method is better compared with other methods, which not only proves the effectiveness of proposed method, but also improve the accuracy of SQL injection vulnerability detection by reducing false negative in the defensive environment.

Key words: SQL injection, penetration test, attack model, second-order fragment and reassembly

收稿日期：2017-09-22

基金项目：国家自然科学基金资助项目（No.61402057）；江苏省产学研前瞻性联合研究基金资助项目（No.BY2016050-01）；江苏省科技计划基金资助项目（No.BK20160411）

Foundation Items: The National Natural Science Foundation of China (No.61402057), The Production and Research Prospective Joint Research Project of Jiangsu Province (No.BY2016050-01), The Jiangsu Provincial Natural Science Foundation (No.BK20160411)

1 引言

Web 在互联网各种业务领域中的广泛应用同时，Web 漏洞却给当前迅速发展的 Web 应用带了极大的安全威胁[1]。通过安全漏洞测试来确定 Web 应用是否存在漏洞，是保障 Web 应用安全性的必要措施[2]。渗透测试是一种根据信息收集模拟攻击和反应分析的方式检测软件安全漏洞的有效方法，特别是在确定 Web 应用安全漏洞（如 SQL 注入漏洞、XSS 漏洞等）存在性方面具有许多优点，因此对其研究成为近年来的热点[3]。但在对 Web 应用的 SQL 注入渗透测试的相关研究中[4,5]，传统的一阶 SQL 注入渗透测试方法由于受到 WAF 和代码防御等防御技术的影响，已经无法有效检测出由新型二阶 SQL 注入[6]等引起的 Web 应用安全漏洞。而针对二阶 SQL 注入渗透测试，目前尚未有适当理论方法来指导生成多种类、多形态的 SQL 注入渗透测试的模拟攻击输入以准确测试 Web 应用防御机制充分性，发现更全面的 SQL 注入漏洞。这使二阶 SQL 注入渗透测试盲目性较强、易造成漏报而降低渗透测试准确度。

为此，本文研究如何从优化 SQL 注入攻击方式的角度，提高 SQL 注入渗透测试准确度，并提出一种基于二阶分片重组的 SQL 盲注渗透测试方法。该方法通过对 SQL 注入的攻击行为建立新的攻击模型，描述 SQL 注入攻击行为的全貌和逻辑规律，提高渗透测试中考虑攻击行为的全面性，指导建立二阶分片重组 SQL 盲注的新型注入攻击方式，并提出对应的渗透测试方法。通过该方法最终生成全面反映二阶分片重组盲注攻击手段和攻击输入的优化 SQL 注入渗透测试用例，以触发更充分的 SQL 注入漏洞，减少渗透测试漏报率，提高对 Web 应用 SQL 注入渗透测试准确度。

2 相关工作

在 Web 应用渗透测试研究中，文献[7]指出信息收集、攻击生成和反应分析是 Web 应用渗透测试的 3 个基本步骤，与 SQL 注入渗透测试的 SQL 注入点查找、SQL 注入攻击和 SQL 注入漏洞识别 3 个阶段相对应，是影响和决定渗透测试准确度的关键因素。

为提高 SQL 注入渗透测试的准确度，目前 SQL 注入渗透测试领域主要通过提高对 Web 应用的输入点发现能力或改进漏洞分析反应等研究降低漏报或误报。例如，在增强渗透测试中爬行方式收集信息能力的研究方面，文献[8]提出一种基于模型分析改进爬虫的 SQL 注入漏洞测试方法，增加渗透测试信息收集的完整性。文献[9]提出一种自动生成有效输入并填入 Web 应用表单的改进爬虫技术，以此发现更多的 Web 应用输入点、提高渗透测试覆盖面。在非爬行方式的渗透测试信息收集改进方法方面，Alenezi[10]提出一种新的基于源码静态分析的渗透测试信息收集方法，解决在测试复杂 Web 应用时，一般爬行方式无法查找到 Web 应用某些类型输入点的问题。

对于渗透测试反应分析阶段的研究，Kim[11]通过数据库日志的数据挖掘，比较 Web 应用中正常与受攻击后 SQL 查询树结构，提出一种改进的 Web 应用反应分析方法检测 SQL 注入漏洞。Jang[12]提出一种基于有效识别 SQL 查询结果大小的反应分析方法，该方法通过识别送往后台数据库执行的 SQL 命令查询结果大小的不同反应来判定 SQL 注入漏洞，以克服 Web 应用受攻击后返回信息大小不明确造成漏报的问题。此外，文献[13]基于标记图（token graphs）对 SQL 命令行为进行建模，并通过 SVM 机器学习比较 Web 应用正常的行为与其受到攻击后的反应来判定 SQL 注入安全漏洞。

以上研究工作主要关注渗透测试的信息收集彻底性和漏洞反应分析正确性 2 个方面的问题，而对于 SQL 注入攻击生成的有效性则缺乏研究。例如，文献[10]等相关研究将攻击生成作为外部因素，仅关注如何使其所引用攻击输入有效到达可攻击输入点，而不对所引用的攻击输入自身的多样性或充分性进行分析。

在 SQL 注入渗透测试的攻击生成研究中，文献[14]提出随机枚举方式生成攻击输入。这种随机枚举的渗透测试方式难以做到全面测试 Web 应用，确定其防御措施是否充分，难以触发隐藏于不充分防御措施后的 SQL 注入漏洞，从而易造成对 SQL 注入漏洞的漏报或误报而降低渗透测试准确度。文献[15]提出基于组合规避的覆盖准则实例化 SQL 注入测试数据，但未对包括渗透测试用例多样性或充分性在内的攻击生成有效性问题进行研究。文献[16]提出一种基于静态分析的二阶 SQL 注入漏洞测试工具，但是无法准确定位存储阶段污染数据的中间存储位置和无法判断触发阶

段污染数据到达危险函数前是否经过有效过滤，因此，存在误报率和漏报率高的问题。文献[17]提出基于动静态结合分析的二阶SQL注入漏洞检测方法，但是该方法只考虑在存储阶段拼接了污点信息，且把完整 SQL 代码写在同一语句中的情况，因此，同样存在过高的漏报率。说明当前对 SQL 注入渗透测试的相关研究中，未充分考虑如何采用多形态、多种类的攻击输入全面测试 SQL 注入漏洞存在性的问题。此外，也没有进一步研究防御措施的防护充分性问题，如果将未知是否充分的防御措施都视为已安全而不存在漏洞，会引起漏洞渗透测试的漏报现象。

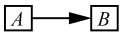
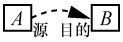
近几年，一些研究中提出了基于模型的渗透测试方法[18]，为渗透测试相关要素建模，以模型规范或指导渗透测试过程。例如，文献[19]提出一种将 Web 应用开发过程与渗透测试活动结合，以实现安全知识共享的模型驱动渗透测试框架。此外，一些专家也对基于攻击树的 SQL 注入攻击建模方法进行了研究[20,21]。但是，这些研究中所提出的模型方法仍然未对上述分析的如何以模型指导 SQL 注入渗透测试攻击方式、增强攻击生成有效性来判定 SQL 注入漏洞及其防御充分性进行研究，未对如何通过改进攻击方式以提高 SQL 注入渗透测试准确度进行明确阐述。因此，本文针对目前 SQL 注入攻击生成和攻击建模方面存在的不足及亟待解决的问题，进行基于攻击模型的 SQL 注入渗透测试研究，以形式化建模指导 SQL 注入渗透测试攻击方式的生成及优化，达到提高渗透测试准确度的目的。

3　SQL 注入攻击建模

SQL 注入攻击建模将实际的 SQL 注入攻击行为转化为模型化描述，为渗透测试提供攻击位置、攻击输入、SQL 注入漏洞反应规律方面的信息。根据 SQL 注入特点，在建模描述其攻击行为时，目前采用的方法有基于攻击树（AT, attack tree）和安全目标模型（SGM, security goal model）等方法[21]。

AT 模型存在对 SQL 注入攻击输入描述不全面，描述规律不能适应所有攻击场景或未对攻击输入规律进行描述等问题[22]。此外，在 AT 模型中攻击行为和结果都用节点表示，容易造成混乱，树分支和节点易产生冗余和重复等缺点。SGM 是一种用来描述漏洞、安全特性、攻击或安全软件开发的新型建模方法。因此，本节采用基于 SGM 的 SQL 注入攻击建模，所建立的模型从攻击目标角度描述 SQL 注入攻击规律，使所提出的 SQL 注入攻击模型能够更好地描述 SQL 注入攻击输入规律。实现通过模型表达 SQL 注入攻击位置、攻击输入和漏洞反映规律信息，同时使所建立的 SQL 注入攻击模型减少表达上的冗余，并可以表述 SQL 注入攻击输入分类等方面的信息，SGM 的建模规则如表 1 所示。

表 1　　安全目标模型建模图例

图形化符号	描述
	根
	与 SGM 无关联的子目标
	与 SGM 关联的子目标
	子目标之间的依赖边
	与（AND）操作符
	或（OR）操作符
	子目标之间的信息边

在表 1 中，根节点表示可以通过实现各子目标而达到的总目标，每个 SGM 只能有一个根节点，其图形化符号用圆角矩形表示。子目标表示一个有助于实现或阻止实现模型总目标的局部目标，每个 SGM 可以含有多个子目标，其中，白色的矩形符号表示和 SGM 无关联且有助于实现总目标的子目标。白色六边形符号表示和 SGM 相关联且有助于实现总目标的子目标，黑色六边形符号表示和 SGM 相关联且阻止实现总目标的子目标。操作符表示子目标之间可实现的关系，包含与（AND）和或（OR）2 种。依赖边表示子目标之间 AND 或 OR 的依赖关系，用一根带箭头的实线符号表示。例如，一根从 *A* 指向 *B* 的依赖边表示为了实现本模型所描述的总目标，子目标 *A* 与 *B* 应当依次实现。每个子目标节点还可以有一个或多个信息口，用于产生（源）或输入（目的）信息。子目标之间传递信息的情况用信息边表示，其图形化符号为带箭头的虚线。

由于 SGM 是描述安全相关的行为目标，因此，基于 SGM 对 SQL 注入攻击建模，需定义 SQL 注入攻击的目标。首先，从攻击总目标角度对 SQL 注入攻击规律进行宏观描述，将 SGM 中的根节点表示为实现 SQL 注入攻击的总目标。然后，将 SQL 注入攻击内在规律描述向 SGM 的子目标描述方式转化，即自底向上描述、按照攻击总目标，将实现这个总目标的攻击分为窃取系统信息、绕过认证和

注入运行恶意命令 3 类子目标。接着，模型进一步向上分别描述实现这 3 种子目标各自所需的攻击子目标，如注入运行 SQL 命令需要查找 Web 应用注入点、注入条件式或注入可执行命令的子目标等，同样其他各攻击子目标的描述也是自底向上表达要实施的攻击子目标，直到模型最上端的一系列攻击注入子目标和查找 Web 应用注入点攻击子目标。新的基于 SGM 所建立 SQL 注入攻击模型如图 1 所示。

在图 1 中，对模型节点进行分层描述，并加入了对 SQL 注入攻击位置信息和漏洞反应规律、反作用的描述。从图 1 中可以看出，模型中自上而下每条不含反作用标节点的路径代表一种攻击子目标的过程。其中，最上端描述攻击输入，中部子目标节点描述 Web 对攻击输入的漏洞反应、即 Web 有何种反应时为具有 SQL 注入安全漏洞。模型中黑色六边形的反作用节点表示 SQL 注入攻击被 Web 应用防御拦截造成不成功的攻击路径。将图 1 中自上而下实现每种攻击子目标的成功攻击路径定义为攻击模式。

定义 1　攻击模式为三元组<GOL，IN，OUT>。其中，GOL 为 SQL 注入的攻击目标，如错误信息利用、盲注、绕过认证、存储过程利用、注入运行 SQL 命令等；IN 为 SQL 注入攻击输入，如条件式、不等式、异常字符、异常命令、时间推断等的集合及它们的组合运算，包括与运算、或运算和复合运算等；OUT 为 Web 应用存在 SQL 注入漏洞的反应。例如，可通过注入异常字符或无法执行的命令来诱发 Web 应用产生错误信息，从中所获得有价值的信息。或者根据包含条件式的时间延迟使 SQL 命令予以执行等。

4　基于二阶分片重组的 SQL 盲注渗透测试

根据第 3 节的 SQL 注入攻击模型及定义 1，本节对攻击模式中的盲注攻击目标进行优化，攻击输入为二阶分片重组的攻击载荷，提出一种基于二阶分片重组盲注的新型攻击模式。这种攻击模式结合二阶 SQL 注入与盲注，并通过分片的方式把 SQL 盲注攻击载荷拆分成多个片段存到数据库中。然后，在触发攻击阶段重组这些片段，并实施 SQL 注入攻击。下面将详细介绍基于二阶分片重组的 SQL 盲注模式及其渗透测试方法。

4.1　二阶分片重组 SQL 盲注

基于二阶分片重组的SQL盲注攻击采用分片存储注入与重组触发注入方式处理攻击载荷。分片存储注入是指在存储攻击载荷时对其做分片处理，即把攻击载荷拆分成几个片段并在每个片段首尾两端添加连接字符，最后把修改后的攻击载荷片段分别通过不同的注入点存入数据库中。重组触发注入是指为了触发攻击载荷，把存储于不同字段中的攻击载荷片段构造到同一个SQL查询语句中。图 2 显示了基于二阶分片重组的 SQL 盲注攻击流程。

从图 2 可以看出，分片存储攻击阶段包括确定分片存储注入点，分片攻击载荷和存储分片的攻击载荷等操作。重组触发攻击阶段包括查询分片的攻击载荷，重组分片的攻击载荷和触发重组的攻击载荷。下面将详细介绍这些步骤。

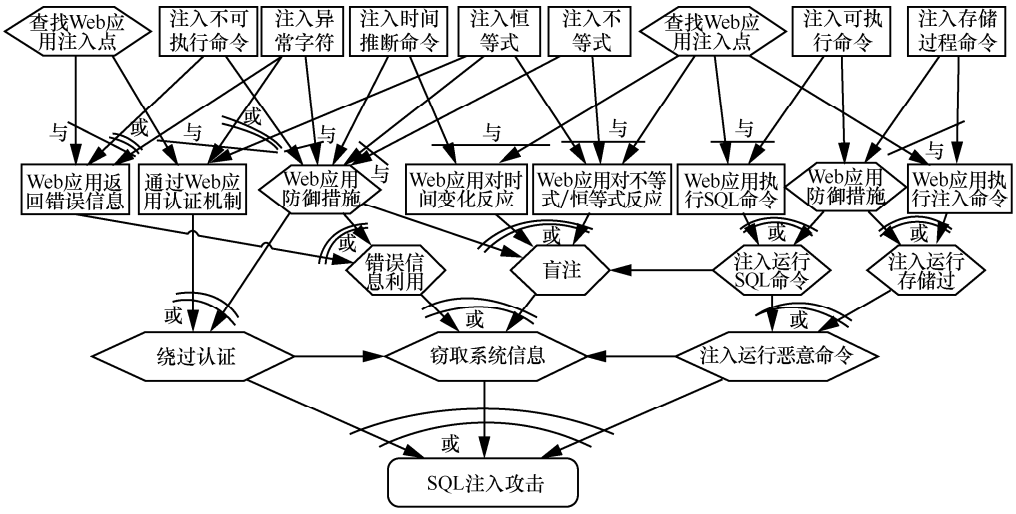


图 1　SQL 注入攻击 SGM 模型

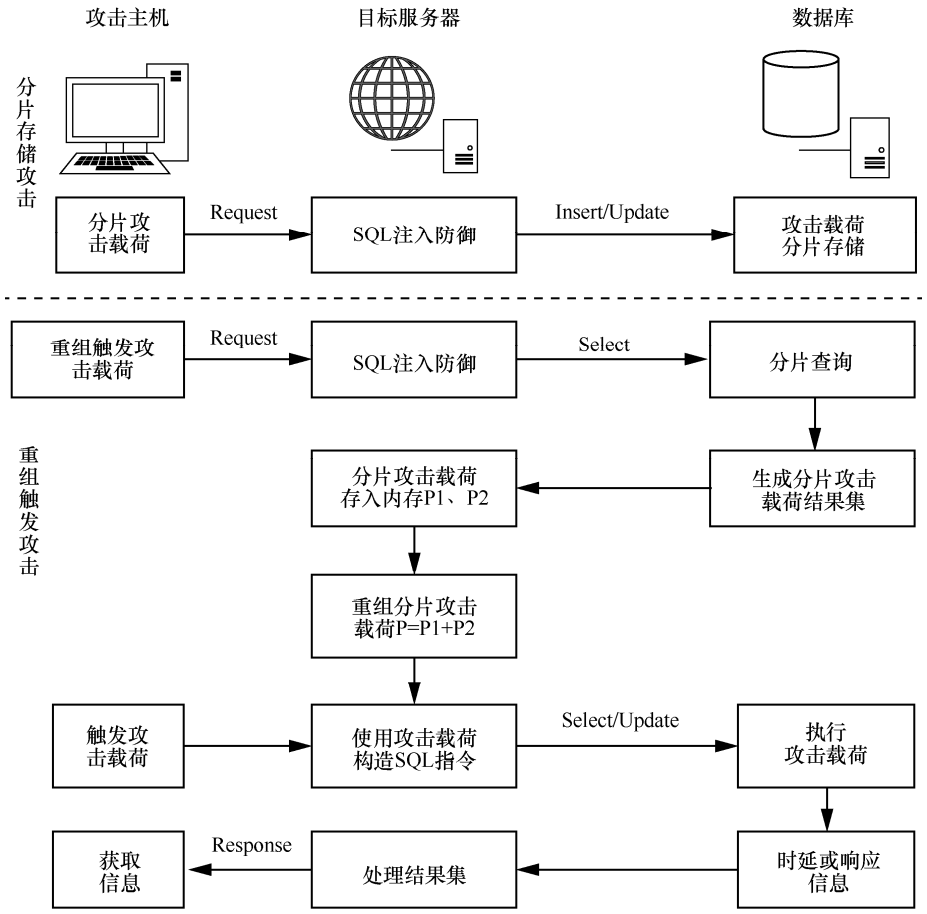


图 2　二阶分片重组 SQL 盲注

4.1.1　分片存储注入攻击

1) 确定分片存储注入点

根据图 1 模型可知，通常分片存储注入点是由重组触发阶段使用不安全的方式所构造的 SQL 查询所决定。例如，Web 应用程序通过使用以下代码实现向数据库中插入一条订单记录。

Insert into mall_product_order (‘user_id’,’ order_id’,’buyer_id’,’seller_id’,’consignee_address’,’consignee_tel’,’consignee_mobile’,’product_id’,’transport’,’time’) values (‘?’,’?’,’?’,’?’,’?’,’?’,’?’,’?’,’?’,’?’,’?’)

在以上 Insert 查询字符串中，consignee_address、consignee_tel、consignee_mobile、product_id、transport 等内容来自于数据库中的其他字段。而这些字段的数据可以通过 Update 或 Insert 等方式写入。因此，Web 应用向数据库写入上述 5 个字段的代码都可能成为分片存储注入点。

2) 分片攻击载荷

在基于二阶分片重组的 SQL 盲注中，分片不是简单把攻击载荷拆分成几个片段，而是还要对每个拆分后的片段做出适当处理。通常为了让这些攻击载荷片段能够最终重组为完整的攻击载荷，需要在它们的头部和尾部添加具有连接功能的字符。常用的有“/* */”“\”“‘”“#”“--”等 5 种，如表 2 所示。

表 2　连接字符

连接符	描述
/* */	多行注释符
\	表示特殊字符
‘	字符分割符
#	单行注释符
--	单行注释符

在表 2 中，“/* */”是 SQL 中的多行注释符。其功能是注释掉“/*”与“*/”中间的 SQL 语句。在重组触发注入点中，可以使用它把同一数据库操作语句中不相邻的攻击载荷片段连接起来。例如，向具有 5 列数据的表 article 添加纪录，其拼接攻击载荷前的 Insert 查询语句如下所示。

Insert into article values(‘param_1’,’ param_2’,’ param_3’,’ param_4’,’ param_5’)

如果分片存储注入只可以控制表中的第 1 列和第 3 列的数据，则可以将“select password from admin LIMIT 1”攻击载荷分片成“sel/*”和“/*ect password from admin LIMIT 1’,’1’,’2’,’3’,’4’); -- ”拼接到上述 Insert 查询字符串中。拼接后的 Insert 查询字符串如下所示。

Insert into article values(‘sel/*’,’ param_2’,’ /*ect password from admin LIMIT 1’,’1’,’2’,’3’,’4’); --’,’ param_4’,’ param_5’)

在上述 Insert 查询字符串中，使用“/* */”和“--”的方式分别注释掉了原查询中第 1、3 列之间和第 3 列之后的内容，并在所构造的第 3 列的数据中补齐了被注释列的内容。上述 Insert 查询字符串简化后的代码如下所示。

Insert into article values(‘select password from admin LIMIT 1’,’1’,’2’,’3’,’4’)

数据库在执行上述语句后向 article 表中添加一条第 1 列的值是 admin 表中第 1 条 password 的值，其余 4 列值分别是 1、2、3、4 的记录。

3) 存储分片的攻击载荷

完成攻击载荷的分片后，可以通过确定的分片存储注入点，将分片后的攻击载荷存入数据库。存储方法包括抓包和在 Web 表单中提交。例如，可以在用户资料注册或修改功能中直接修改 consignee_address、consignee_tel、consignee_mobile 3 个字段的内容。由于无法直接在浏览器中修改提交订单时所发送的 product_id 和 transport 2 个字段的内容，这时就可以采用代理类软件（如 Burpsuite）抓取数据分组进行修改[23]。

4.1.2 重组触发注入攻击

重组触发注入是指在 Web 应用程序中以非安全的方式所构造的 SQL 查询，且该查询中的数据来自分片存储注入点存储的分片数据。数据库在编译新构造的 SQL 查询时，由于不同攻击载荷片段中的连接字符而生成包含完整且连续攻击载荷的执行计划。依据攻击载荷被分片存储的位置不同，重组触发操作可以分为重组同一表中的分片攻击载荷、重组不同表中分片攻击载荷和通过额外数据库操作重组触发 3 种方法。

1) 同一表中攻击载荷片段重组

重组同一表中攻击载荷片段方法是指在做重组操作时，重组的攻击载荷片段来自于同一表中。用户只需要把各个攻击载荷片段从数据库中提取出来并重新构造 SQL 查询就能实现把不同攻击载荷片段重组为完整的攻击载荷，最终被数据库执行。例如，Web 应用的用户注册功能会向 Users 表添加一条用户记录，这条用户记录中存在多个注入点，如用户名、密码、邮箱、昵称、用户签名、联系方式等。用户在分片存储阶段中把攻击载荷进行拆分存储到这些注入点中。而在重组触发阶段，用户只需要对 Web 服务器提交修改用户资料请求。Web 应用就会把这些攻击载荷片段重组到 Update 查询字符串中，并发送给数据库执行。数据库在执行拼接了众多攻击载荷片段的 Update 就会引起 SQL 注入攻击。

由于所重组的攻击载荷来自于同一张表，因此重组同一表中攻击载荷片段方法是最接近传统二阶 SQL 注入技术，也是最容易实施的重组触发方法。但同时也正是由于同一张表中的注入点往往有着相似的权限，或被相似的防御措施处理，它所要求的 Web 应用环境也最为宽松，绕过 SQL 注入防御措施的能力最差。

2) 不同表中攻击载荷片段重组

重组不同表中的攻击载荷片段方法是指在做重组攻击载荷操作时，所重组的攻击载荷片段来自于不同表中。用户需要把各个攻击载荷片段从数据库不同表中提取出来并重新构造 SQL 查询，才能把不同攻击载荷片段重组为完整的攻击载荷，并被数据库执行。例如，在电商网站提交订单功能中，Web 应用会把提交订单用户的资料和订单信息一起插入到存储订单信息的表中。为实现该功能 Web 应用会生成一个 Insert 查询字符串，并在其中拼接用户资料表中的数据和用户通过表单新提交的订单数据。如果此时是以非安全方式构造 Insert 查询字符串，数据库执行后将触发攻击载荷。

重组不同表中攻击载荷片段的方法可以整合不同注入点条件实施注入攻击，在实施过程也较为容易，绕过 SQL 注入防御措施的能力强。例如，Web 应用的用户评论功能为了能让用户正常输入评论而必须允许如“‘”“\”“#”等特殊字符的输入。而在用户注册等功能中则严格限制了这些特殊字符。通过使用重组方法，用户可以把攻击载荷中的关键字符存入这些注入点，从而绕过 SQL 注入防御措施重组完整攻击载荷并触发 SQL 注入攻击。

3) 额外数据库操作重组

额外数据库操作重组方法是指在做重组攻击载荷操作时，通过额外 Select 子查询把位于不同表中的攻击载荷片段存储到同一个表中。并且在最终触发操作之前，这种重组操作可以多次进行。例如，在 Web 应用程序修改订单功能中，用户可以通过在这些注入点中添加子查询以查找来自其他表或者其他数据库中的攻击载荷，并通过 Update 方式更新到订单表中。最后通过提交订单操作触发攻击载荷。

额外数据库操作方法是对重组不同表中的攻击载荷片段方法的进一步改进。它与后者的区别在于通过 Select 子查询的方式从任意表中提取攻击载荷片段。该方法突破了重组不同表中的攻击载荷片段方法只能重组位于触发 SQL 语句中所包含的注入点中数据的限制。对于绕过 SQL 注入防御措施有更好的效果。

4.2 渗透测试方法

传统 SQL 注入漏洞扫描工具无法对基于二阶分片重组的 SQL 盲注漏洞提供有效的检测[4,5]。首先，二阶 SQL 注入本身不会在 Web 应用的当前数据提交部分触发，因而想要检测到该漏洞需要寻找 Web 应用中能够操作已存入数据库中用户输入的模块。其次，SQL 盲注特点决定了其触发检测条件不能采用 2 种经典的易于检测的响应作为断依据，而无论是使用基于布尔值还是使用基于时间的盲注技术，都增加了判断难度。由于工具本身无法理解 Web 应用的逻辑，只能通过遍历的方式寻找关联页面，而也正是由于无法理解 Web 应用逻辑，在寻找关联页面的方式上没有有效的判断依据。因此，本文提出基于二阶分片重组的 SQL 盲注的渗透测试方法，其渗透测试流程如图 3 所示。

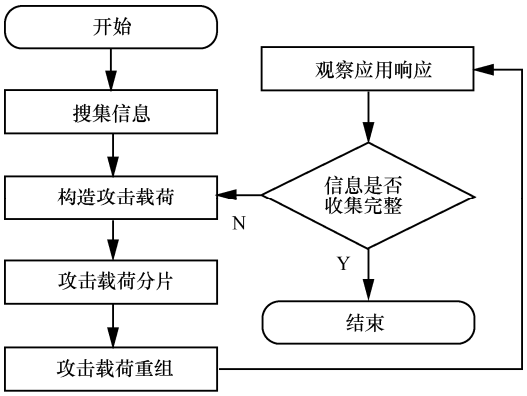


图 3　基于二阶分片重组盲注的渗透测试流程

从图 3 可以看出，搜集信息是利用基于二阶分片重组的 SQL 盲注渗透测试的第 1 步也是最重要的一步。在这一步中，通过浏览整个网站理解该 Web 应用的逻辑关系，并依据测试用户对于网站逻辑的理解，寻找一组会重复用到相同数据的关联页面。在这组关联页面中，测试用户通过其中一个页面中存入的数据会被其他页面重复使用。在找到关联页面后，尝试注入测试语句引发 Web 应用的异常响应判断渗透攻击载荷的触发点。之后，通过表单提交、抓包修改等方式寻找和确定注入点。最后，判断触发点所使用的 SQL 语句类型。

构造渗透攻击载荷是利用基于二阶分片重组的 SQL 盲注进行渗透测试的第 2 步。根据第 3 节 SQL 注入攻击模型中，攻击想要获取的信息，利用基于时间或基于布尔值的推断技术构造 SQL 盲注渗透攻击载荷，并随着第 5 步中 Web 应用返回的信息不断调整渗透攻击载荷。

渗透攻击载荷分片是利用基于二阶分片重组的 SQL 盲注攻击进行渗透测试的第 3 步。测试用户依据在第 1 步中搜集到的信息决定渗透攻击载荷分片的数量、渗透攻击载荷片段间的连接方式、渗透攻击载荷中需要补齐的原 SQL 查询元素等操作。对渗透攻击载荷进行分片处理，并分别存储到不同注入点中。

第 4 步，渗透攻击载荷重组。测试用户通过请求在第 1 步中确定的会产生异常的页面使这些被分片存储渗透攻击载荷片段重组到一个 SQL 查询语句中并被触发。

第 5 步，观察应用响应。Web 应用执行渗透攻击载荷后产生异常（如时延或逻辑等），测试用户由此获得想要提取的信息并决定下一步如何构造渗透攻击载荷。渗透测试流程跳到第 2 步继续执行，直到测试用户获得足够的漏洞判断信息。

5 测试与分析

为验证本方法的正确性和有效性，下面构建实验环境，对实际测试用例进行 SQL 注入漏洞检测与分析。

5.1 不同 SQL 盲注方法比较测试与分析

首先，通过对真实 Web 应用的 SQL 注入测试比较不同盲注漏洞检测方法的测试效果。本文选择 3 个实际中广泛使用的开源 Web 应用 phpmps v2.4、cmseasy v4 和 mallbuilder v7 作为测试用例，并通过

基于时间的盲注[24]、基于布尔值的盲注[25]以及本文基于二阶分片重组的 SQL 盲注进行测试。其中，基于时间的盲注是向测试参数中注入含有“sleep (4)”异常条件的攻击载荷，判断 Web 应用是否会产生延时进行测试。基于布尔值的盲注是向测试参数中分别注入“And 1=2”和“And 2=2”逻辑条件，通过判断 Web 应用能否返回正常页面进行测试。基于二阶分片重组的盲注是通对含有“sleep (4)”异常条件的攻击载荷进行分片存储、重组触发后，观察 Web 应用是否会产生延时进行测试，测试结果如表 3 所示。

表 3 不同 SQL 盲注方法测试比较

Web 应用	基于时间盲注	基于布尔值盲注	本文方法
phpmps v2.4	×	×	√
cmseasy v4	×	×	√
Mallbuilder v7	×	×	√

从表 3 测试结果可以看出，不论是基于时间的盲注还是基于布尔值的盲注都没有测出这 3 个 Web 应用存在的 SQL 注入漏洞，而本文方法能够测出这 3 个 Web 应用都存在 SQL 注入漏洞。通过对这 3 个 Web 应用进行源代码审计发现，它们都存在二阶 SQL 注入漏洞，并且每个注入漏洞都存在多个注入点。由于这 3 个 Web 应用都对来自客户端的用户输入进行了严格的 SQL 注入过滤，因此，对这 3 个 Web 应用都无法直接进行一阶 SQL 盲注漏洞检测。其中，Web 应用 phpmps v2.4 和 Mallbuilder v7 严格限制了注入点的输入字符的数量，导致单纯任何一个注入点无法容纳足够的攻击载荷，因而只能采用基于二阶分片重组的 SQL 盲注漏洞检测方法进行测试。在本次实验测试中，测试 Web 应用 phpmps v2.4 属于被重组的攻击载荷片段来自不同表的情况，即在用户注册过程中向 phpmps_users 表的 username 字段注入部分载荷，而在用户评论功能的 phpmps_comment 表中 content 字段注入剩下的载荷。由于在提交用户评论时，应用程序会把 username 字段和 content 字段中的值一起拼装到新构造的 Insert 查询中，而最终存储进数据库。当查看该用户评论时就会触发完整的攻击载荷。测试 Web 应用 Mallbuilder v7 属于被重组的攻击载荷片段在同一表中的情况。该 Web 应用在添加订单时直接使用了数据库中的用户资料，且没有做二次过滤，因此存在二阶 SQL 注入漏洞。测试用户在该 Web 应用的用户注册功能中向 mallbuilder_users 表中 consignee_name、consignee_address、consignee_tel、consignee_mobile 等字段中注入攻击载荷片段，并最终在添加用户订单操作时把这些攻击载荷片段重组到新的 Insert 查询字符串中。新构造的 Insert 查询字符串被数据库执行后引起 SQL 注入。测试 Web 应用 cmseasy v4 对用户注册时 e-mail 只限制了输入字符数量不超过 50 而对内容没有限制，可以向 e-mail 中注入子查询拼接用户评论中的攻击载荷片段。因此，在一阶 SQL 盲注检测无效的情况下，基于二阶分片重组的 SQL 盲注漏洞检测不仅能够突破注入点中字符数的限制，还能够突破 Web 应用中数据库执行权限的限制，相比于一阶 SQL 盲注漏洞检测具有更多的应用场景。

5.2 WAF 防御测试与分析

Web 应用防火墙（WAF, Web application firewall）通过审计设备、访问控制、架构网络设计和加固等功能实现对 Web 应用的保护。下面，通过在真实 Web 应用中部署 WAF 后的 SQL 注入测试本方法在 WAF 防御下的测试效果。

首先，采用 Linux+Apache+MySQL 实验环境建立 5 种不同类型的开源 Web 应用，并且分别部署 PHPIDS v0.6.4 和 GreenSQL v1.3.0 这 2 种 WAF，分别用 WAF1 和 WAF2 表示。然后，分别采用重言式、非法/逻辑错误的查询、联合查询、附带查询、利用存储过程等一阶 SQL 注入渗透测试方法[26]和本文方法对这些 Web 应用进行测试。在绕过 WAF 时，使用大小写混合、编码、注释、替换关键字、HTTP 参数控制、整合绕过、等价函数与命令、缓冲区溢出等方法对攻击载荷进行变换。其中，本文方法通过 ASCII (substring(user(), 1,1)) = ‘a’代码进行测试，并利用分片的方法结合上述 8 种普通方法绕过 WAF，测试结果如表 4 所示。

表 4 WAF 环境下的 SQL 注入漏洞测试结果比较

Web 应用	一阶 SQL 注入			本文方法		
	无	WAF1	WAF2	无	WAF1	WAF2
XDcms v5	√	×	×	√	√	√
天生创想 OA v2014	√	×	×	√	√	√
phpyun v3.2	√	×	×	√	×	×
齐博 CMS v5.0	√	×	×	√	√	√
U-Mail v9.8.57	√	×	×	√	√	√

从表 4 可以看出，本次测试的 5 个 Web 应用都存在一阶和二阶的 SQL 注入漏洞。当没有部署 WAF

时，可以通过实施一阶和二阶 SQL 注入检测，成功发现这些 SQL 注入漏洞。但在部署 WAF 后，发现重言式、非法/逻辑错误的查询、联合查询、附带查询、利用存储过程等一阶 SQL 注入测试被有效拦截，而无法检测出 Web 应用存在 SQL 注入漏洞。在本文方法的测试中，不论系统是否部署有 WAF，仍然能够测试出 Web 应用存在 SQL 注入漏洞。这是因为本文方法在测试中，二阶 SQL 注入漏洞的注入点是在 Web 应用具有存储功能的部分，而这一部分由于需要用户输入数据，无法采用严格的输入过滤（如用户评论功能不能禁止“‘”等字符）。此外，由于本文方法对攻击载荷进行分片处理，使分片后的攻击载荷特征分散，更不易被 WAF 检测出来，导致测试用户可以通过对攻击载荷变形绕过 WAF 的防御。测试网站 phpyun v3.2 在部署了 PHPIDS v0.6.4 和 GreenSQL v1.3.0 这 2 种 WAF 时均无法检测出注入漏洞。这是因为 phpyun v3.2 存在的分片注入点只有 2 个，导致对攻击载荷的分片数量较少，SQL 注入特征依然明显而被 WAF 拦截。以上测试比较结果说明，本文方法能够减少在 WAF 环境下的 SQL 注入漏洞漏报，提高漏洞检测的准确度和在 WAF 安全防御环境下的充分性。

5.3　代码防御测试与分析

下面，测试本方法在部署不同代码防御技术环境下的 SQL 注入漏洞检测效果，并与现有的 SQL 注入漏洞检测方法进行比较分析。首先，通过 PHP 实现 2 个 Web 应用：社区应用 Web1 和通讯录应用 Web2，其中，Web1 采用 MySQL 数据库，Web2 采用 SQL Server 数据库，并分别在其中部署参数化查询、addslashes 函数、Trim 函数、str_replace 和开启 Magic-quote-gpc 等代码防御方法[27]。然后，分别采用重言式、非法/逻辑错误的查询、联合查询、附带查询、利用存储过程等 SQL 注入漏洞检测方法和本文方法对 Web1 和 Web2 应用进行测试，其中，本文方法通过 if (ASCII(substring(user(),1,1))= ‘a’, SLEEP(5), 1)和 IF ASCII(substring(user(),1,1))= ‘a’ waitfor delay ‘00:00:05’攻击载荷进行测试，测试结果如表 5 所示。

从表 5 可以看出，由于在 Web1 和 Web2 应用中采用了参数化查询、addslashes 函数、Trim 函数、Magic-quote-gpc、str_replace 等代码防御技术，因此采用传统的一阶 SQL 注入渗透测试方法进行 SQL 注入漏洞测试时，由于上述防御措施会对攻击载荷进行“失活”处理而无法检测出 Web1 和 Web2 应用存在 SQL 注入漏洞。但是在采用本文方法进行测试时，虽然有些代码防御技术能够使分片攻击载荷中的注入语句暂时“失活”，但是仍然会使得攻击载荷被存入数据库中。当通过重组的方法从数据库中提取出这些攻击载荷并重新用来构造 SQL 查询时，就会“激活”这些攻击载荷，并造成 SQL 注入，因此不但可以正确检测出 SQL 注入漏洞，而且可以减少因代码防御而产生的漏报，提高检测准确度。

表 5　不同代码防护下的 SQL 注入漏洞测试结果比较

代码防御技术	一阶 SQL 注入		本文方法	
	Web1	Web2	Web1	Web2
参数化查询	×	×	√	√
addslashes 函数	×	×	√	√
Trim 函数	×	×	√	√
Magic-quote-gpc	×	×	√	√
str_replace	×	×	√	√

6　结束语

SQL 注入漏洞在出现后的十几年时间里，从开始的不被重视到现在 Web 应用普遍采用防注入措施，一直是影响 Web 应用安全的首要漏洞。二阶 SQL 注入是 SQL 注入发展出的一种新技术，如同 SQL 注入漏洞刚出现时一样并没有引起应有的重视，以至于在很多安全从业人员中也不太了解，得以广泛存在于现实之中。本文提出基于二阶分片重组盲注的渗透测试方法在无法使用基于错误的数据提取和带内数据连接的情况下，把基于布尔值或基于时间的攻击载荷作为数据分片存入数据库中，然后再通过 Web 应用能够读取或修改之前存入数据库中数据的相关功能重组、触发攻击载荷。测试用户通过分析应用程序的响应获取数据。实验测试证明本方法不仅比现有的一阶 SQL 注入渗透测试方法更加有效，而且能在部署 WAF 和代码防御的环境下仍能正确检测出 Web 应用的 SQL 注入漏洞，大大降低了渗透测试的漏报率，提高了准确度，在 Web 的安全检测中具有重要的应用价值。本文的测试还只是基于本文方法的手工测试，在下一步的工作中，需要继续通过软件开发技术研究能够进行自动分片注入点定位和分片存储注入，并进行自动分片重组触发注入的自动化二阶分片重组盲注渗透

测试工具，进一步提高本方法的测试效率。

参考文献：

[1] OWASP. The ten most critical Web application security risks[S]. OWASP Top 10, 2017.

[2] ANTUNES N, VIEIRA M. Designing vulnerability testing tools for Web services: approach, components, and tools[J]. International Journal of Information Security, 2017,16(4): 435-457.

[3] ANTUNES N, VIEIRA M. Penetration testing for Web services[J]. IEEE Computer, 2014, 47(2): 30-36.

[4] DEEPA G, THILAGAM P S. Securing Web applications from injection and logic vulnerabilities: approaches and challenges[J]. Information and Software Technology, 2016, 74(6):160-180.

[5] DALAI A K, JENA S K. Neutralizing SQL injection attack using server side code modification in Web applications[J]. Security & Communication Networks, 2017, 2017(2) :1-12.

[6] 乐德广, 李鑫, 龚声蓉, 等. 新型二阶 SQL 注入技术研究[J]. 通信学报, 2015, 36(Z1): 85-93.
LE D G, LI X, GONG S R, et al. Research on second-order SQL injection techniques[J]. Journal on Communications, 2015, 36(Z1):85-93.

[7] HALFOND W G J，CHOUDHARY S R, ORSO A. Improving penetration testing through static and dynamic analysis[J]. Software Testing Verification & Reliability, 2011, 21(3):195-214.

[8] SALAS M I P, MARTINS E. A black-box approach to detect vulnerabilities in Web services using penetration testing[J]. IEEE Latin America Transactions, 2015, 13(3):707-712.

[9] CHEN J M, WU C L. An automated vulnerability scanner for injection attack based on injection point[C]//IEEE International Computer Symposium (ICS). 2010:113-118.

[10] ALENEZI M, JAVED Y. Open source Web application security: a static analysis approach[C]//IEEE International Conference on Engineering & MIS (ICEMIS). 2016:1-5.

[11] KIM M Y, LEE D H. Data-mining based SQL injection attack detection using internal query trees[J]. Expert Systems with Applications, 2014, 41(11):5416-5430.

[12] JANG Y S, CHOI J Y. Detecting SQL injection attacks using query result size[J]. Computers & Security, 2014, 44(2):104-118.

[13] KAR D, PANIGRAHI S, SUNDARARAJAN S. SQLiGoT: detecting SQL injection attacks using graph of tokens and SVM[J]. Computers & Security, 2016, 60(3):206-225.

[14] KIEZUN A, GUO P J, JAYARAMAN K, et al. Automatic creation of SQL Injection and cross-site scripting attacks[C]//31st IEEE International Conference on Software Engineering. 2009: 199-209.

[15] HUANG H C, ZHANG Z K, CHENG H W, et al. Web application security: threats, counter measures, and pitfalls[J]. IEEE Computer, 2017,50(6):81-85.

[16] DAHSE J, HOLZ T. Static detection of second-order vulnerabilities in Web applications[C]//23rd USENIX conference on Security Symposium (USENIX). 2014:989-1003.

[17] YAN L, LI X H, FENG R T, et al. Detection method of the second-order SQL injection in Web applications[J]. Lecture Notes in Computer Science, 2014, 8332(1):154-165.

[18] MARBACK A, DO H, HE K, et al. A threat model-based approach to security testing[J]. Software-Practice & Experience, 2013, 43(2): 241-258.

[19] XIONG P L. A model-driven penetration test framework for Web applications[D]. University of Ottawa, 2012.

[20] KAUR N, KAUR P. Modeling a SQL injection attack[C]// 3rd IEEE International Conference on Computing for Sustainable Global Development (INDIACom). 2016:77-82.

[21] BYERS D, SHAHMEHRI N. Unified modeling of attacks, vulnerabilities[C]//ICSE Workshop on Software Engineering for Secure Systems (SESS). 2010: 36-42.

[22] 田伟, 许静, 杨巨峰, 等. 模型驱动的 Web 应用 SQL 注入渗透测试[J]. 高技术通讯, 2012,22(11):1161-1168.
TIAN W, XU J, YANG J F, et al. Model-driven penetration test of the SQL injection in Web applications[J]. Chinese High Technology Letters, 2012, 22(11):1161-1168.

[23] VIBHANDIK R, BOSE A K. Vulnerability assessment of Web applications - a testing approach[C]//4th IEEE International Conference on e-Technologies and Networks for Development (ICeND). 2015:1-6.

[24] LIBAN A, HILLES S M. Enhancing MySQL injector vulnerability checker tool (mysql injector) using inference binary search algorithm for blind timing-based attack[C]//IEEE 5th Control and System Graduate Research Colloquium. 2014:47-52.

[25] DABAS A, SHARMA A K. Understanding advanced blind SQLI attack[J]. International Journal of Engineering Research and General Science, 2015,3(3): 1548-1552.

[26] HALFOND W, VIEGAS J, ORSO A. A Classification of SQL-injection attacks and countermeasures[C]//International Symposium on Secure Software Engineering (ISSSE). 2006: 12-23.

[27] ANTUNES N, VIEIRA M. Defending against Web application vulnerabilities[J]. IEEE Computer, 2012,45(2):66-72.

作者简介：

乐德广（1975-），男，福建三明人，博士，常熟理工学院副教授，主要研究方向为信息安全与下一代互联网技术等。

龚声蓉（1966-），男，湖北天门人，博士，常熟理工学院教授、博士生导师，主要研究方向为图像处理与信息安全等。

吴少刚（1973-），男，安徽宿松人，博士，中科梦兰电子科技有限公司研究员，主要研究方向为计算机系统结构、并行与分布式计算等。

徐锋（1981-），男，江苏常熟人，中科梦兰电子科技有限公司高级工程师，主要研究方向为计算机体系结构及自主安全。

刘文生（1969-），男，福建泉州人，泉州市公安局高级工程师，主要研究方向为网络安全。