浅谈企业SQL注入漏洞的危害与防御——安全小课堂第五期

2016-04-08 京东安全应急响应中心

安全小课堂第5期

SQL注入攻击是一个非常传统的攻击方式，且SQL注入方式与手段变化多端。据不完全统计，在日常漏洞中，SQL注入占比约10%，虽不算高占比，但其危害极大，业内企业因此蒙受损失的新闻层出不穷。我们本期和各位聊一聊企业SQL注入漏洞的危害与防御。

今天的嘉宾分别是百度安全架构师d4rkwind、乌云核心白帽子（网络尖刀核心成员）紫霞仙子，还没落座，大家的掌声已是如雷贯耳。

1

豌豆妹

首先聊聊业内SQL注入的现状呗~

哆啦A梦

过了这么多年，注入的问题依旧还是如此多!最根源的问题是，没有安全编码规范。

2

豌豆妹

SQL注入带来的风险有哪些？

小新

SQL注入上得了机器权限，下得了数据。攻击者利用SQL注入漏洞，带来的风险有很多，例如数据库被拖库，管理员和重要人员信息泄露，甚至还能通过SQL注入漏洞直接获取webshell或者服务器系统权限等等。

3

豌豆妹

SQL注入漏洞有哪些分类呢？

葫芦娃

比如按照利用方式分类。常见的注入类型有：盲注，报错注入，time盲住，union注入， 内联查询注入，拼接（堆）查询注入。
按照攻击入口分，可以有GET型的SQL注入、POST型的注入、Cookie型的注入、Header型SQL注入等。
按照注入点类型分，可以分为整型注入、字符型注入等。

豌豆妹

目前所掌握的注入漏洞种类，出现频率较高有哪些？

小丸子

以平时关注的来说，频率高的有： 盲注，time盲注，报错注入，union注入。

豌豆妹

危害最大的呢？

哆啦A梦 

在不影响正常服务的情况下，拼接查询算最高危害的，接下来就是union。

豌豆妹

比较易被检测出来的有哪些？

小新 

现在比较好检测的注入有：盲注，time盲注，报错注入等。

4

豌豆妹

能分享下SQL注入漏洞的挖掘思路么~~例如SQL注入漏洞经常出现的位置，如何去判断这类漏洞易出现的位置等。

葫芦娃 

我喜欢白盒（结合黑盒）的方式，能发现很多有意思的产品实现和有意思的点。 挖掘思路上，大的方面是： 爬虫+规则。

小新 

关于注入的位置：常发生于用户和服务交互处（增删改查操作），ajax，接口等等，用这个检测报错注入，比较方便。

哆啦A梦 

举个栗子。👇
PHP+MYSQL扩展默认字符编码为GBK，且代码并未在real_escape_string前强制调用set_charset（real_escape_string是官方推荐的标准安全转义方法），然后研发在代码中使用query('set names utf8')设置连接字符编码 ，我们就可以通过info?name=d4rkwind%df\' or sleep(3)-- -&other=xxx方式来注入成功（打破了只要不是“set names gbk”就不会存在半个双字节编码SQL漏洞的误区）。原理很有意思的，这个tips估计外界都没公布过。PHP mysql_real_escape_string 对SQL语句中用户可控的name变量进行了GBK编码下的转义，即d4rkwind%bf\' or sleep(3)-- - 转义成了d4rkwind%bf\\' or sleep(3)-- -,总的SQL语句成为
select xxx from yyy where name='d4rkwind%bf\\' or sleep(3)-- -'但这个“select xxx from yyy where name='d4rkwind%bf\\' or sleep(3)— -'” 实际上是utf-8的了。所以，成功闭合了。

5

豌豆妹

如何监测和防御这类漏洞的存在？各公司是否有更有效的手段来防御和发现这类问题

呢？

小丸子

最好的防御，就是内部先发现 。监测方面目前大多都是：日志监控+waf。从厂商角度，日志是最好的资源，日志监控很重要。日志推荐走数据库日志，越是离资源操作近的地方，越是容易做到真正的安全。数据库日志容易解析，语法出错的、语法读Info表的，都明确是黑客嘛，还能帮我们发现SQL注入点。
统一的filter也是有效的（如waf），统一filter的目标不是避免漏洞被发现，还是避免漏洞被利用，一个不能被利用的SQL注入，可以认为不是漏洞，脱离了潜在的风险，任何漏洞都不是漏洞。更有效的filter是能读懂访问者的输入（攻击都是来自于输入），如此才能提高准确率，满足大流量互联网公司的诉求，所以，这几年逐渐已经有WAF能初筛似的解析输入，理解语法语义，对于语法错误的可以直接放行。当然，一定要记得是初筛，世界上不同类型数据库不同版本数据库的语法语义都有一定的差异，不可能一个filter能实现所有的解析，太精确的，就只能帮助防御特定版本的数据库了。

哆啦A梦

研发还是得多培训，让他们有安全意识。大部分漏洞，尤其是TOP10漏洞，都是研发的问题，而非设计的问题。

葫芦娃

重要业务开发完成后，必须代码扫描，测试过程中，也会有安全测试和扫描。但是公司体量大，代码层面人工审计太累，不符合投入产出比，业内也没有牛逼的白盒扫描器，这个得慢慢推动，过程比较长。但是我们尝试过一种方式，很有效，已经在重点业务上线了，就是强制框架，强制安全库，所有SQL操作必须使用这个库，不然上不了线，自动化检查是否有非这个库外的SQL操作，这个准确率超级高。另外，我们线上，不仅有实时的安全扫描，24小时不间断，还有日志监控，相应速度到秒级。

豌豆妹

如何能够第一时间发现正在被SQL注入攻击？

哆啦A梦

总结下就是：日志监控-蜜罐数据-异常报警。日志前面已经提过，不再赘述。数据库里可放置一些蜜罐数据的帐号和密码。我们这有一个服务，不过也是从日志入手，发现请求恶意异常时，自动转发蜜罐。比如用户表里，前几十行里，做些用户名和密码进行，实际上没有人用，一旦被登录，立马报警。不管怎么说，日志监控，必须得做起来，而且做一个WAF和数据库防火墙非常有用。

豌豆妹

有白帽子问到一个问题：希望可以补充下SQL注入的发现思路。能答疑解惑一下吗？

哆啦A梦

个人建议白帽子们多练练白盒的测试，也能提高自己。白盒很有意思，不仅能让你挖到

漏洞，长期关注后，也能了解一个产品的实现思路：
1. 快速方式是，关键词匹配，找到SQL语句后，往回溯，看看这些SQL语句在哪被调用，哪里被带入了变量，变量的值来自哪里，是否有安全校验，安全校验是否匹配当前SQL操作的具体场景（字符集编码等也要留意），沿着变量和函数的调用，一直回溯查到输入点就好。还有一种思路是，从输入开始一个个往逻辑里看，个人觉得，后者没有前者高效，因为所有输入不一定会带入SQL，而有SQL的大部分都会直接或间接的有输入变量。
2. 有意思的方式是，了解代码使用的框架或者代码结构，如看看代码对请求进行路由和分发的方式，这个路由分发方式的设计和实现是否存在隐患，记录一下，再看看是否有一些统一的安全filter，记录下他的特性（任何统一的安全filter都会因为不了解后端调用的场景而产生绕过），然后再看看是否有基础的DB库，这个库是否实现了安全的SQL操作，最后结合这些因素和方法1，可以轻易发现SQL注入点，更能站在防御者视角有效的给出举一反三的修复建议。
搞安全的，知识面很广，在于的不是黑盒的方式一个个发现漏洞。而是因为了解业务实现的方式，设计的思路，这样黑盒看到同类网站，就可以知道怎么下手，知识面越广也越容易发现更多的技巧。黑盒方式上，我觉得查询SELECT型的注入很容易被发现，其实可以多考虑考虑insert update里的注入点发现，关系型数据库里，结合这个接口的功能，提交的返回值对比，response code 5xx ，其实也容易黑盒发现注入点。只是发现注入点后如何利用，这个也是技巧。

豌豆妹

哈~干货多多，受益匪浅！非常感谢本次的特邀嘉宾和各位核心白帽子的大力支持！下期再见！