SQL Server 数据库

安全策略指南

SQL Server 数据库安全策略指南

数据库安全威胁和数据偷窃的发生率在不断增加，而行业专家认为如果想要在 2010 年及以后继续保护敏感信息，大多数组织需要重新考虑他们的数据库安全性策略。在本次的技术手册中，我们将讨论 SQL Server 数据库安全的话题，其中包括数据加密、SQL 注入攻击、角色控制以及安全通信方面的内容，希望对 SQL Server 安全有个更全面了解的读者，赶紧下载阅读吧。

数据库安全策略分析

IT 行业分析师认为数据库管理现在所面对的最大问题是恶意分子使用越来越复杂的攻击手段，而数据库安全防御的发展仍然停滞不前。他们说更多的公司应该利用高级数据库安全技术来保证客户和公司机密信息的安全，避免其被盗。

- 2010：数据库安全策略关注度需提升
- 未来的数据库安全性技巧
- 需要考虑的数据库相关安全政策

SQL Server 数据加密

随着数据安全需求的不断增加，不论以何种方式，都不要忽略对你的数据库备份文件的安全保证。SQL Server 加密能够为您的数据库的安全提供额外的保护。

- 为增强安全性 SQL Server 加密备份
- 理解 SQL Server 2008 透明数据加密（一）
- 理解 SQL Server 2008 透明数据加密（二）
- 能否在不同数据库服务器上进行不对称加密解密

- SQL Server 密码破解工具简介

SQL 注入攻击

SQL 注入攻击可能是黑客攻击面向互联网的 SQL Server 数据库最常用的方式。任何使用动态 SQL、允许未经检验的用户输入提交到数据库的应用程序都面临 SQL 注入攻击的风险，无论你的网络有多安全，或者你安装了多少防火墙。

- 如何保证 SQL Server 免遭 SQL 注入攻击？

SQL Server 访问控制

数据库角色允许我们定义一个特定的权限集。然后，服务器登录（不管是在 AD 组、AD 用户或者 SQL Server 登录）都将添加到服务器上和数据库角色中。

- 通过数据库角色控制访问（上）
- 通过数据库角色控制访问（下）

SQL Server 数据传输安全性

从一个实例向另一个实例发送 SQL Server Service Broker 信息最安全的方法是使用证书以确保通信安全。虽然这是最安全方法，但它比普通 SQL Server 或 Windows 身份验证技术要复杂得多。

- 设置 SQL Server Service Broker 进行安全通信（上）
- 设置 SQL Server Service Broker 进行安全通信（下）

SQL Server 安全审计

SQL Server 2008 引进了一种新的审计性能，它能使 DBA 追踪数据库的使用并进行详细审计。我们能在服务器和数据库上安装审计，在单独的数据库对象上激活并用各种不同的格式保存，如二进制文件或 Windows Application 日志。

- 在 SQL Server 2008 中安装安全审计（一）
- 在 SQL Server 2008 中安装安全审计（二）
- SQL Server 安全审计中的常见疏忽

进一步加强 SQL Server 安全性

SQL Server 的安全性是由多个层次构成的：网络、操作系统、服务器以及数据库。这些层次需要分别进行加强，因为它们其中任何一个出现问题都会导致整个方案的失败。

- 加强网络安全
- 加强操作系统安全
- 加强服务器与数据库安全

2010：数据库安全策略关注度需提升

数据库安全威胁和数据偷窃的发生率在不断增加，而行业专家认为如果想要在2010年及以后继续保护敏感信息，大多数组织需要重新考虑他们的数据库安全性策略。

IT行业分析师认为数据库管理现在所面对的最大问题是恶意分子使用越来越复杂的攻击手段，而数据库安全防御的发展仍然停滞不前。他们说更多的公司应该利用高级数据库安全技术来保证客户和公司机密信息的安全，避免其被盗。

同时，分析师认为Oracle目前提供给用户高级安全功能方面做得好于它的竞争对手——IBM和Microsoft。然而，他们补充道，同时运行来自不同供应商的多个数据库管理系统(DBMS)的公司可能需要借助第三方支持来实现更全面的数据库安全性策略。

“我们已经在数据库系统中存储的数据及与数据相关的风险都同时在增加，”Jeffrey Wheatman，来自Stamford, Conn.-based Gartner Inc.的主管信息安全和隐私的研究主管说道。“但不幸的是，我们并没有在客户中看到我们所预期的更多保护这些数据存储的措施出现。”

(作者：Mark Brunelli 译者：曾少宁 来源：TT中国)

原文标题：2010：数据库安全策略关注度需提升

链接：http://www.searchdatabase.com.cn/showcontent_30400.htm

未来的数据库安全性技巧

随着内部数据偷窃和更狡猾黑客攻击的增加，诸如认证、授权和访问控制等基本的数据库安全方法已经不够了，Noel Yuhanna，来自 Forrester Research 的一个长期数据库管理系统(DBMS)分析师说道。

Yuhanna 说这些基本的方法需要用更广泛的数据库安全性策略来加强，这些策略需要结合各个方面深刻的理解，其中包括每个所保护数据库的原因，以及所有监管需求的最新信息，正确的高级数据库安全技术，如加密和数据置乱或屏蔽、审计、监控和变更管理。

Forrester 说建立一个强大的数据库策略的第一步是建立包括认证、授权、访问控制、数据恢复和分类的基本要素的基础架构——可能还有最重要的、稳固的补丁管理实践方法。

“大多数组织没有很好地在他们的系统上安装补丁，”最近撰写了一篇关于 2010 年数据库安全性策略的文章的 Yuhanna 说道。“大约 65%到 70%的组织并没有定期安装补丁。”

他说，现在许多组织有大量缺乏跟踪的数据库，他们根本不知道数据库中的内容是什么，这就是为什么数据库恢复和分类之所以重要的原因。分析师说每个公司都应该定期清查生产和非生产数据库，然后根据是否包含敏感信息及它们的安全规定来对这些数据库分类。

创建一个稳固的安全性策略的下一步就是使用加密、数据屏蔽和变更管理来保护数据。Yuhanna 说加密应该主要使用在生产数据库中，非生产数据库则进行数据屏蔽，因为这些非生产数据库常用于测试、开发和培训。这两者之间的区别在于加密的文件可以被有相应权限的用户解密，而数据屏蔽或置乱一般是永久地使数据变杂乱。Yuhanna 说两种方法将有利于保护敏感数据不受窥探。

“仅有 16%的组织正在做数据屏蔽，但是这个数据在过去的 2 年里翻了一番，”他说。[数据屏蔽]是一定需要的，而且我们强烈建议客户实施一个数据屏蔽策略。

受采访的 DBA 和其它信息技术专家都说他们认为在可能的地方伪装数据是很好的做法。DBA 和应用开发人员解释说，他们经常拷贝一份生产数据库数据，然后将信息移到非生产数据库进行测试。而当数据进入非生产环境后，它们就变得更容易被内部数据盗取。

“一个 DBA 不需要去查看数据的内容，”来自 Alexandria, Va.的一个长期 DBA(姓名隐去)说道。“他们要做的就是确认数据库是操作和提供服务的。”

变更管理是一个处理 IT 体系架构内部变更的系统方法，它也是保护生产数据库免受攻击的好方法。Yuhanna 说公司应该要求模式结构的变更遵循正式的规程，其中包括文档和批准过程。

一个稳固的数据库安全性策略的最后一个主要的组件是强大的入侵检测功能的实现，包括审计、监控和持续的漏洞评估，Yuhanna 说。

他解释说审计——收集关于系统资源使用方式的过程——是非常重要的，因为它能告诉管理员谁在访问数据、数据被访问的时间以及数据作了什么修改。分析师说组织应该尽快彻底调查有哪些重要数据修改不是预期的。监控技术也可以在这方面有所帮助，因为它们可以在出现任何可疑活动发生时进行“实时”通知。

漏洞评估报告可以帮助公司识别数据库安全环境的缺陷，包括弱密码和不当的访问权限，Yuhanna 补充说。

(作者：Mark Brunelli 译者：曾少宁 来源：TT 中国)

原文标题：未来的数据库安全性技巧

链接：http://www.searchdatabase.com.cn/showcontent_30404.htm

需要考虑的数据库相关安全政策

在能够真正影响每个行业的所有的政府和行业规则中，总有一天会出现这么一条，要你真正地应用一些信息安全政策。你也许已经对密码和数据备份有了一些基本政策。但是还有更多内容。所以，如果你的企业现在才刚刚把它的安全政策放在一起，或者你已经意识到了是时候需要更新一些东西了，那么这里有几个你需要了解的数据库安全相关的问题。

从技术上来说，为了准确判断需要哪个安全政策，你需要执行信息风险评估。然而，我理解，现实情况经常会导致其它内容。就是说，我可以考虑得很少，如果有的话，但是几乎很少有不再要求我思考如下数据库相关安全政策的环境出现：

• 可接受的利用率——什么可以/不可以在数据库服务器上完成，例如网络浏览和安装/卸载中间件，以及个人防火墙保护，还有 MSDE, SQL Server Express 2005，以及其它非服务器系统上的数据库软件的安装。

• 认证控制——对于数据库，以及有关应用程序和操作系统来说，包括密码的要求，多种成分的使用等。

• 业务合作——应对外部承包人、审计人员、寄存提供者商等，包括合同规定和应用的服务级别的协商。

• 业务连续性——灾难恢复和/或业务连续性计划要求可以帮助你的数据库保持运转状态，可以访问。

• 变更管理——记录谁、为什么、在什么时候，如何，以及所有相关的拆除过程。

• 数据备份——什么，什么时候，以及使用的方法。

• 数据保持和破坏——什么，为什么，使用的方法和时间线。

• 加密——不仅覆盖了静止的数据，例如加密特定的行，还覆盖了传输的数据，例如数据库服务器和网络应用程序之间的 TLS。数据备份也是覆盖的一部分。

• 信息分类——为信息贴上公共、内部、机密等标签。

• 物理安全——构建，数据中心和服务器的安全。

• 财产的删除——服务器，驱动，磁带，和其它财产。

• 安全测试和审计——什么，如何，什么时候，以及谁执行的测试，用的什么工具。

• 职责的分隔——用户，数据库管理员，安全审计员，以及其它与数据库管理有关的人的角色/职责。

• 系统维护——打补丁，系统清理/清楚和中间件的更新。

• 系统监控和意外事件的响应——谁，为什么，什么时候，以及如何进行实时监控和记录审计日志，还有为了维护正式的意外响应计划所需要的特殊需求。

• 用户授权——添加/删除用户，授予谁，为什么，什么时候，多长时间的管理权限。

我认为这里有几个关键点需要与数据库中心的安全政策相关。首先，如果管理层没有明确表示他们的支持(例如，他们将会接受并强制政策的执行)，你兴许也只能一起放弃这个计划。所以首先要把他们也拉上船。其次，尽可能地把你的政策提到最高级别上，这样你就可以最大化覆盖的部门和系统。最大化规则的数量，你可以真正地实施它。换句话说，如果你可以帮助它，不在你的数据库中发生以上所有的政策，并且在无线网络、存储系统等等中拥有另一套规则。同样，至少也要理解你的企业对解决 PCI, GLBA, HIPAA, SOX 等问题上的规则上的调整需求。这一点在你有一个依从性或者 IT 管理委员会来审视和现实安全政策的时候，可以带来最好的帮助。如果你能帮助它的话，你不会想要你自己管理一套政策。

最后，确保你尽可能地以一种可以直接带来最大理解和管理的方式记录你的政策。以下政策中的组成部分对于保证政策的简单性和长期的可管理性是至关重要的。

• 简介——对题目的简单概括，例如加密

• 目的——简单列出最高的目标和方针的策略。

• 范围——说明覆盖了哪些雇员、部门和数据库系统。

• 角色和职责——列出与谁有关，以及要他们支持政策的话，需要他们做什么。

• 政策的陈述——你的真正的政策的陈述。你应该对每个主题有一篇陈述。换句话说，为前面列出的你选择使用的每个政策创建一个单独的模版。

• 例外情况——强调没有包括在政策里面的人，部门，数据库等。

• 过程——政策如何实现和在你的环境内强制执行的详细步骤。你可能会想要参考这个信息，并且把它放在不同的文件中。

• 遵循——列出如何衡量是否遵循了这个政策的过程，包括所有涉及的衡量标准。

• 制裁——列出当违反了政策时候会发生的事情。这可能包括了第一次违反的时候发生什么事情，第二次违反的时候发生什么事情，以及第三次违反的时候发生什么事情。

• 回顾和评估——说明什么时候政策必须检查其准确性、实用性，以及遵循的目的(例如，. SOX, HIPAA, GLBA, PCI 等)。

• 参考——指出调整代码部分河信息安全标准(ISO/IEC 17799, ITIL, COBIT 等)

• 相关文档——指出其它政修订——记录针对这个政策文档进行的修改。

• 修订——记录针对这个政策文档进行的修改。

• 注意事项——最重要的注意事项，贴士等。它可以帮助以后的政策管理和加强。

如果你以正确的方式做完了以上所有内容来构建你的政策，它会在很长一段时间内节省你大量的时间和烦恼，让你的审计员很高兴。良好的回报值得你的努力。

(作者：Kevin Beaver 译者：曾少宁 来源：TT 中国)

原文标题：需要考虑的数据库相关安全政策

链接：http://www.searchdatabase.com.cn/showcontent_9025.htm

为增强安全性 SQL Server 加密备份

随着数据安全需求的不断增加，不论以何种方式，都不要忽略对你的数据库备份文件的安全保证。在本地的 SQL Server 备份中，备份文件中的数据是以普通文本格式存储的，仅仅用文本编辑器就可以轻松阅读。根据表中使用的数据类型，一些数据比另外的一些数据更容易查看。

试试这个你几乎从来没有进行过的试验。对 Northwind 数据库进行备份，或者任何其它小型数据库，然后用任意的文本编辑器打开备份文件。你将会看到数据自身有一点难以理解，但是只要你看到存储过程的注释，然后通读一下文件，你就会看到你的备份文件的真正价值所在。如果你采取行动，将用户 ID 和/或密码存储在你的存储过程中，首先这可不是一个好主意，这个数据现在就可以被任何能够接触到备份数据库的人所访问。如果你有其他藏有秘密信息的文本数据类型，你也会让这些数据非常有意义了。

备份密码

SQL Server 中的一个选项就是创建用密码创建备份。这是你在创建备份的时候可以使用的另一个选择，但是在企业版管理器或者 SQL Server 管理套件中，并没有提供这个选项。这里是一个使用密码选项备份的例子:

```
backup database northwind to disk=’c:\northwind.bak’ with mediapassword = ’Backup2006’
```

这个过程需要密码来重新存储文件，但是使用文本编辑器，这些数据仍然是可以访问的。还有，重新存储不能使用 GUI 来完成，所以它必须通过 T-SQL 重新存储命令和密码一起完成任务。

加密存储过程

一种防止你的存储过程被用于查看的方法就是在创建你的存储过程的时候使用“带加密”的选项。这样的话，备份文件中的数据也是经过加密的了。要使用加密来创建一个存储过程，如下所示:

```
create procedure dbo.testEncryption
        with encryption
              as
select * FROM products
```

加密数据

另一个选择就是在你把数据存储到你的数据库表中的时候，对数据进行加密。在 SQL Server 2000 中没有本地的方法来完成，但是有很多工具你可以使用:

针对 SQL Server 的 NetLib Encryptionizer

使用 XP_CRYPT 加密 SQL Server

SQL Server 2005 中存在本地加密功能。看看微软的文章<>如何:加密一列数据，那里解释了这个过程。在你加密了数据库中的数据之后，当你创建备份的时候，数据仍然是经过加密的。

保证文件系统的安全

保卫你的备份文件的安全的另一个方法就是在你的服务器或者网络中使用安全目录。你可以限制访问这个目录的权限，这样就只有一小部分受限制的人能够访问你的备份文件。通过在安全目录上使用上述的技术，你就可以创建另一个级别的安全措施了。这仍然不会消除加密的需求，但是它提供了额外的安全措施。

直接备份到磁带

备份的另一个选择就是直接备份到磁带上，以便在你的网络中保证备份文件的安全。这种方法减少了对你的备份文件的不正当访问的问题。用这种方式有一个大问题:我写入的大部分关于备份的内容都是首先写入磁盘的，以便在必要的时候能够快速重新存储，然后再为了长期的存储而归档到磁带上。这种方式消除了你的备份落入坏人之手的机会，但是，不幸的是，它让其他的处理过程变得困难。

加密备份

如果你真的需要保证你的备份文件的安全，最好的方式就是在你创建备份文件的时候对其进行加密。不幸的是，SQL Server 中没有工具可以让你完成这个任务，但是看看以下的产品，它们可以让你创建各种级别的密码和加密技术加密备份。

Idera 的 SQLsafe

Quest 的 SQL LiteSpeed

Red-Gate 的 SQL Backup

有几种不同的方式来保护你的数据库备份文件，以及备份文件内容。查看你的数据库，找出哪个数据库有需要保护的信息或者代码。然后实现一个或者几种上述的技术来确保你的数据远离图谋不轨的眼睛。

（作者：Greg Robidoux 译者：曾少宁 来源：TT 中国）

原文标题：为增强安全性 SQL Server 加密备份

链接：http://www.searchdatabase.com.cn/showcontent_10984.htm

理解 SQL Server 2008 透明数据加密（一）

透明数据加密（TDE）是 SQL Server 2008 中的一个新特征，被设计用来对数据库文件、数据库备份以及临时数据库进行加密。通过透明数据加密，当你从数据库中读取数据时，数据将被实时加密，且不会阻止任何合法用户进入数据库访问并读取表格数据。

透明数据加密 - 为什么要使用它？

PCI DSS（支付卡产业数据安全标准）要求每一个数据和备份都必需是安全的，而透明数据加密一开始就是用来帮助正在使用 SQL Server 2008 的公司来满足符合此标准（PCI DSS）中的各项条款的要求。

记住，TED 本身不能满足你所有的安全要求。它只是 SQL Server 2008 帮助 DBA 遵从法规时提供的一整套特征中的一部分。DBA 仍需确保敏感数据已通过加密算法被加密，且网络管理员与系统管理员必须确保 Windows 服务器、网络、网页以及应用服务器之间的链接都是安全的。开发人员仍需确保从客户端到 Web 服务端的通信是安全的或者已经被加密了的。

使用 TED 之前需要考虑的

在 SQL Server 中使用透明数据加密之前，你应该考虑几个因素。

例如，任何正在使用 SQL Server TDE 的公司可能会发现有细微的性能衰减，因为数据在向磁盘写入的过程中被加密，在从磁盘中读取出来的过程中被解密。这就增加了 CPU 的消耗。但是此时数据文件，事务日志文件以及备份文件的大小与没有使用 TDE 的数据库中的文件大小是一样的。

加密后的数据库，其备份的压缩比远比没有加密的数据库的备份要小。这就导致备份时需要更多的存储设备，且在传递那些加密备份文件时需要花费更多的费用。

虽然在 SQL Server 中，一开始也可以通过密码实现安全的备份，但这被认为是一个不太可靠的选择。现在大多数磁带备份方案在向磁带设备写入数据时都包含备用的加密算

法。虽然磁带加密技术这项技术在过去发展很慢，但在最近几年还是有很多发展。尽管如此，这些发展阻止不了黑客访问你的 SQL Server,也阻止不了他们试图分离你的数据库文件、将它们拷贝到另一个 SQL Server 中，附加它们并读取数据库内容。因此，数据库文件加密技术对于大多数常规使用是必需的。

以下是在实现透明数据加密之前要考虑的一些其它的重要因素：

使用 TDE 要有数据库密钥（DEK）以及任何所需的证书。在存储备份时需要用到此证书。

如果你正在使用 TDE,即时文件初始化会被禁用。即时文件初始化是 Windows Server 2003 的一个特征，SQL Server 2005 可以在数据增涨非常快的时候利用此特征，使得文件系统中的潜在空间不需要被清空。

如果你正在记录一个透明数据加密的数据库的日志或为其做镜像,二级或镜像服务器需要启动 TDE。

文件流数据将不会被加密。文件流是 SQL Server 2008 的一个特性，varbinary 列可以存储在文件系统中并异步流向客户端。

数据库中的只读文件组必需被设为可写，才能激活 TDE 来为数据库内容加密。它们之后又可以设为只读。

为透明数据加密激活数据库可能会花费一些时间，且一些数据库操作在转换过程中不会被激活。参考微软的关于理解 TDE 的网页,来获取更多关于这些局限的信息。

复制是 TDE 的盲区,复制后的数据不会被加密。换言之，复制网络通信经常会是纯文本，也可能会是复制后的快照文件。DBA 需要考虑到这一点。

在生成索引的过程中,全文索引将会从 varbinary 列与 image 列即时抽取文本数据至文件系统。这些数据是无格式文本，并且没有被加密。微软建议不要将全文索引数据存存储在 varbinary/image 列中。

(作者：Hilary Cotter 译者：张峰 来源：TT 中国)

原文标题：理解 SQL Server 2008 透明数据加密（一）

链接：http://www.searchdatabase.com.cn/showcontent_22649.htm

理解 SQL Server 2008 透明数据加密（二）

激活 SQL Server 2008 中的透明数据加密

要激活 TDE,你首先必须创建一个服务主密钥（SMK）。为此，需在你的主数据库中使用以下语句：

```
Create Master Key Encryption By Password = 'MyPassword'
```

然后，你需要通过一个证书来保护 DEK。通过此证书，你可以将此证书转移到另一个你需要存储 TDE 的受保护的数据库服务器上。你可以通过使用以下语句来实现此功能：

```
CREATE CERTIFICATE MyCertificate WITH SUBJECT = 'My Certificate'
```

你也许要将证书连同私有密钥备到文件系统中。确保把这两类文件保存在一个安全的，已知的位置。如果你丢失了这些文件，你将不能恢复数据库并读取其中的内容。

```
BACKUP CERTIFICATE MyCertificate TO FILE = 'c:\temp\MyCertificateBackup.bck'
                                WITH PRIVATE KEY  (
                            FILE = 'c:\Temp\MyPrivateKey.key',
                          ENCRYPTION BY PASSWORD = 'MyPassword') ;
```

你现在需要创建一个通过上面的证书加密的数据库密钥。

```
                        CREATE DATABASE ENCRYPTION KEY
                          WITH ALGORITHM = AES_256
                    ENCRYPTION BY SERVER CERTIFICATE MyCertificate
```

现在你可以通过以下命令激活数据库的透明数据加密功能:

```
ALTER DATABASE myDatabase SET ENCRYPTION ON
```

最后，你可以通过查询下边的 DMV 来监控加密转换的过程或状态:

```
Select db_name（database_id）, encryption_state from
sys.dm_database_encryption_keys
```

关于SQL Server中的透明数据加密,要记住有一点很重要,就是它不是一个静止的加密方案。它也不会对你数据库中的敏感数据进行加密,但是数据文件及备份除外。你仍需通过加密个别列来保护敏感数据，从而允许通过认证的用户来查看它们。

（作者：Hilary Cotter 译者：张峰 来源：TT中国）

原文标题：理解SQL Server 2008透明数据加密（二）

链接：http://www.searchdatabase.com.cn/showcontent_22650.htm

能否在不同数据库服务器上进行不对称加密解密

问：在 SQL Server 2005 中，能否在一个数据库服务器上使用不对称加密算法进行数据加密，然后将数据复制到另一个数据库服务器，再进行解密?

在这个项目中，我们想要将遗留数据从一个数据库服务器复制到另一个数据库服务器上，并在发送到目标系统上之前进行转换。数据的加密与解密是在两个不同的服务器上进行的，貌似我可以使用密码进行对称加密，但非对称加密的话就不行。理性的情况下，我想旨在解密的时候输入密码，而不要加密时也输入密码。

答：如果你想要用密码进行解密的话，你就需要在加密时也用密码。为了让这一过程自动进行，我建议使用证书加密，导出证书然后将其导入到目标 SQL Server。或者另过程更加透明，在服务器端设置 IPSec，两端的数据传输就都加密了。这样的话，在 SQL Server 上就无需做任何改动：数据简单地从 SQL Server 之间传送，操作系统会自动对传输进行加密解密。

如果你的 NIC 支持，你甚至可以利用 NIC 处理器进行数据加密解密，取代了 CPU，可以让系统整体性能得到提升。

(作者：Denny Cherry 译者：孙瑞 来源：TT 中国)

原文标题：能否在不同数据库服务器上进行不对称加密解密

链接：http://www.searchdatabase.com.cn/showcontent_29577.htm

SQL Server 密码破解工具简介

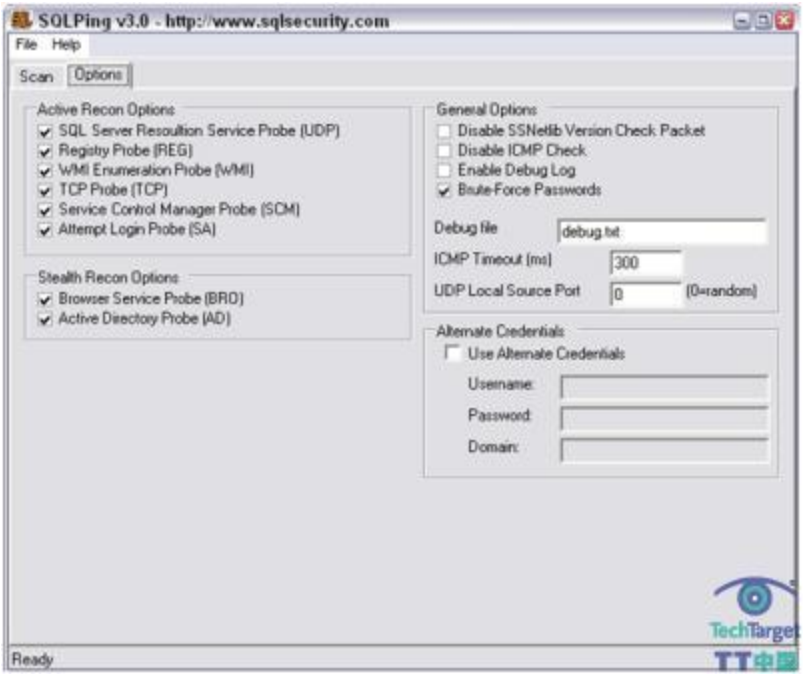
在对 SQL Server 系统执行入侵测试或者更高级别的安全审计时，有一种测试不应该被忽略，那就是 SQL Server 密码测试。这一点看起来显而易见，但是很多人都会忽略它。

密码测试可以帮助检查恶意入侵者或者外部攻击者，测试他们要强行进入数据库有多容易，而且还可以确保 SQL Server 用户对他们的账号负责。此外，测试密码的漏洞在 SQL Server 混合模式认证的情况下尤其重要，这种模式比其他 Windows 认证模式安全性要差一些。

密码测试的第一步是确定要测试的系统。虽然你对你的环境可能了如指掌，但是找出那些可能被遗忘的服务器，或者有未经你知晓有人就连接到网络的服务器的情况是没有坏处的。

SQLPing3 是一个免费的 SQL Server 查找和密码破解工具，可以帮助你开始测试。该工具有多个选项可以供你搜索活动状态的 SQL Server 系统，如图 1 所示：

图 1：用 SQLPing3 搜索活动 SQL Server 系统的选项界面(点击放大)。

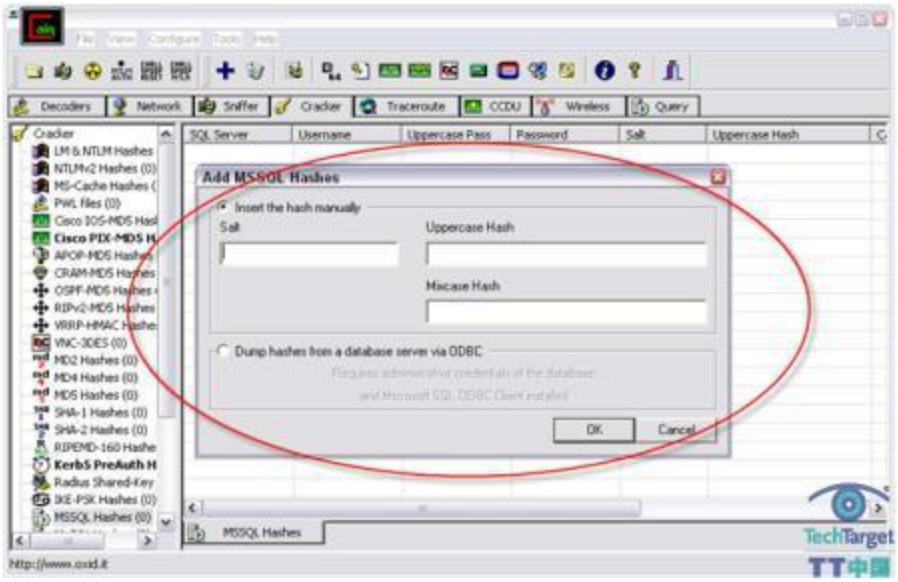


此外，SQLPing3 可以扫描到那些通过约定俗成的端口扫描可能扫描不到的 SQL Server 数据库实例，而且它可以找到那些“sa”密码为空的系统。SQLPing3 还可以针对 SQL Server 数据库运行字典攻击，这种做法就像加载你自己的用户账号和密码列表一样简单。

因为这是最基础层面的 SQL Server 搜索和密码破解，所以我们从这里开始非常合适。

另一个免费工具是 Cain&Abel，它支持你转存并攻击 SQL Server 数据库密码哈希，如图 2 所示：

图 2：使用 Cain&Abel 转存并破解哈希(点击放大)。

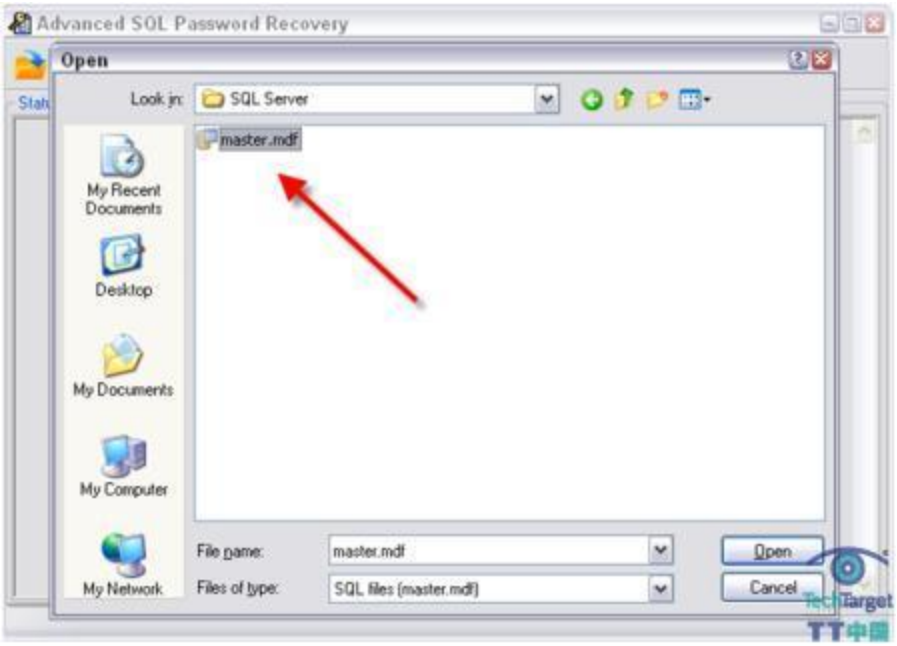


使用 Cain&Abel，你可以插入你自己的哈希或者通过 ODBC 连接到数据库并把它们一股脑转存下来，以便后续破解使用。

在商业软件里面，NGSSQLCrack 和 AppDetective Pro 都是很好的工具，他们可以执行字典破解和暴力密码破解。

我最喜爱的新的商业 SQL Server 密码破解工具是 Elcomsoft 公司开发的 Advanced SQL 密码恢复工具。使用 Advanced SQL Password Recovery，你可以立即从“master.mdf”文件中恢复密码，如图 3 所示：

图 3：使用 Advanced SQL Password Recovery，从“master.dbf”指向并直接点击密码破解(点击放大)。



这看起来似乎是不可思议的或者是完全不可能，因为 SQL Server 系统被认为在网络范围内是锁定的。然而，我经常会碰上管理员级别的密码或者发现丢失的补丁，一试之下，发现可以很容易地以全部权限访问数据库服务器。从这一点上看，系统中的一切就都是暴露着的游戏而已。

一定要记住的，SQL Server 密码破解不应该被忽视。要把它当成正式的安全评估，得到管理方面的支持，周密地规划。因为你不想碰到麻烦。

尽管如此，密码破解也存在一些缺点要记住：

- 密码破解会消耗宝贵的系统资源，包括 CPU 时间，内存和网络带宽，积累到一定量就会给系统造成拒绝式服务攻击。
- 字典型攻击和暴力攻击会花大量时间，有时候你可能得不到结果(时间太长)，尤其是如果你只能在特定的时间窗口内测试系统。
- 字典攻击的效果取决于你使用的字典，所以确保你操作时拿到的字典是可靠的。我发现 BlackKnight List 是最全面的字典。

最后，可能也是最重要的，一定要跟进你的发现。这可能意味着与管理层和你的 IT 部门同事分享你的发现，调整你的密码策略并传播安全意识，来说明安全对企业来说是多么重要的一个问题。

（作者：Kevin Beaver 译者：冯昀晖 来源：TT中国）

原文标题：SQL Server 密码破解工具简介

链接：http://www.searchdatabase.com.cn/showcontent_29941.htm

如何保证 SQL Server 免遭 SQL 注入攻击？

SQL 注入攻击可能是黑客攻击面向互联网的 SQL Server 数据库最常用的方式。任何使用动态 SQL、允许未经检验的用户输入提交到数据库的应用程序都面临 SQL 注入攻击的风险，无论你的网络有多安全，或者你安装了多少防火墙。最近有一项关于 Web 黑客攻击的报告显示，SQL 注入攻击呈上升趋势，不仅仅导致数据盗窃和数据丢失，而且在最近一连串的自动注入攻击中，数据库被破坏，向用户提供恶意 Java 脚本。这种渗透导致 Web 服务器令客户端电脑感染上其它的病毒。报告对于已经遭到攻击的网站数量统计不一，不过，哪怕是最低的数字都上万。在感染的高峰期，甚至联合国的网站也无法幸免。

你可能会认为 SQL Server 平台既然这么不安全，不如考虑更换平台。但事实上，所有的数据库平台都受到这种攻击的困扰。由于在主机环境中更多部署的是 SQL Server，所以，针对 SQL Server 的攻击也就相应更为普遍。开发人员进行网页开发工作，他们不知道如何防范这样的攻击。由于这种攻击的成功率很高，所以，在恶意软件的社区内颇为流行。

SQL 注入攻击是如何工作的？

容易遭受 SQL 注入攻击的 Web 应用程序具有以下特点：

1. 你的网站使用动态 SQL。这不是指应用程序动态生成 select 或者 insert 语句。它是指任何代码是动态生成的，包括应用程序在执行语句之前，动态生成一个存储过程。

2. 当从客户端应用程序取值的时候，这些值没有经过验证——也可能没有对语法或者转义符进行验证。

SQL 注入攻击是这样进行的：攻击者修改现有命令行，通过在字符串值内插入一个单引号，或者在数字后面添加分号，在转义符后面写入 SQL 语句。命令行看上去类似这样：

```
exec sel_CustomerData @CustomerId=47663; truncate TABLE Customer
```

这样会执行 sel_CustomerData 过程，然后运行 truncate TABLE 命令，删除 Customer 数据表的内容。如果有 Foreign Key 约束这个数据表，数据库将会返回错误，向黑客提供

受约束的数据表名称。一名聪明的黑客可以利用这个技巧，查找到数据库里面每一个表的名称。然后，黑客能够将数据插入你的数据库，或者从你的表中选择数据（取决于数据库给了应用程序什么样的权限）。当黑客获取了数据表中的数据，就可以使用 xp_sendmail 或 sp_send_dbmail 向他们自己发送电子邮件。即使你已经禁用这些过程，黑客可以轻易启用这些过程，或者使用 sp_OA 过程添加他们自己的过程。

如何确保 SQL Server 数据的安全，避免 SQL 注入？

有多种办法可以保护你的数据库，避免这样的攻击。

首先，我们需要采用数据库安全的最佳实践方法加固数据库安全。这包括将数据库的操作许可设置为最低级别（setting up the database security with the lowest set of permissions possible.）。同时，应用程序不要直接访问数据表。所有对数据表的访问应该通过存储过程进行，而且那些存储过程不应该包括任何动态 SQL。

避免对表的直接访问，你可以很大地减少受攻击的层面。但是，这并不是唯一必须做的事情。存储过程也存在受到攻击的可能性。虽然对存储过程进行攻击需要花费更多的时间，但是，仍然有可能利用你的存储过程对数据库进行破坏。存储过程就是用来向数据库中插入、更新和删除数据。一个聪明的黑客可以利用你自己的存储过程攻击你。

这是需要应用开发人员和你共同工作的方面，以确保被执行的代码（the code being executed against the database）是安全的。如果不确保应用层的安全，防范 SQL 注入攻击，其他的工作都将是徒劳的。数据只要进入数据库，基本上不可能在数据库内进行验证。这需要在应用层对数据进行验证。

应用程序与数据库最简单配合使用的方法就是在应用程序内动态生成 SQL 命令。下面的示例中，.NET 代码从前台应用程序中调用（populate）v_Input 变量：

```
…
Dim v_Conn As New SqlConnection(p_Connectionstring)
v_Conn.Open()
Dim v_cmd As New SqlCommand
v_cmd.Connection = v_Conn
v_cmd.CommandType = CommandType.Text
v_cmd.CommandText = "exec sel_CustomerData @CustomerName='" & v_Input &
```

″, ″

```
    Dim v_DR As SqlDataReader
    v_DR = v_cmd.executeReader
    v_DR.Close()
    v_DR = Nothing
    v_cmd.Dispose()
    v_cmd = Nothing
    v_Conn.Close()
    v_Conn = Nothing
    v_DR.Close()
```

如果你不在 v_Input 变量中对数据进行验证，就相当于为 SQL 注入攻击敞开大门。如果你对输入不进行验证，就会允许攻击者传入一个单引号和一个分号，这样，将告诉 SQL Server 结束当前的语句，转而执行另一段 sql 语句 pass in a single quote, and a semicolon, which tells the SQL Server to end the value and the statement moving on to the next statement in the batch。例如像这样的值“Smith ’ ; truncate table Customer; declare @myV =”。通过 SQL Server 执行的 SQL 语句会变成：
The resulting SQL statement executed against the SQL Server
exec sel_CustomerData @CustomerName=’ Smith’ ; truncate table Customer; declare @myV = ’ ’

当应用程序（calling application runs the code）运行这段代码的时候，将运行存储过程，然后表被清空。你需要进行一些基本的验证，将变量中的任何单引号替换为两个单引号。这样，就可以中止 SQL Server 运行删除语句 truncated statement，因为现在它是值的一部分。通过这样的简单变化，我们的数据库调用现在就会像这样：

exec sel_CustomerData @CustomerName=’ Smith’ ’ ; truncate table Customer; declare @myV = ’ ’ ’

一种更好更安全的解决方式就是把存储过程代码用参数来表示。这让.NET 来处理变量的数据净化，这样，任何注入代码都不会执行。

.NET 代码从你的前台应用程序中调用 v_Input 变量。

```
...
Dim v_Conn As New SqlConnection(p_Connectionstring)
v_Conn.Open()
Dim v_cmd As New SqlCommand
Dim v_Parm As New SqlParameter
v_cmd.Connection = v_Conn
v_cmd.CommandType = CommandType.StoredProcedure
v_cmd.Parameters.Add("@CustomerName", SqlDbType.NVarchar, 255)
v_cmd.Parameters.Item("@CustomerName").Direction =
ParameterDirection.Input
v_cmd.Parameters.Item("@CustomerName").Value = v_Input
v_cmd.CommandText = "sel_CustomerData"
Dim v_DR As SqlDataReader
v_DR = v_cmd.executeReader
v_DR.Close()
v_DR = Nothing
v_cmd.Dispose()
v_cmd = Nothing
v_Conn.Close()
v_Conn = Nothing
v_DR.Close()
```

如果网站的前台应用程序和后台数据库的安全无法正确得到全面的保障，你的系统和数据就很容易受到 SQL 注入攻击。这些攻击可能表面看上去没有造成什么影响，但如果把你的所有客户数据暴漏给攻击者，问题就大了。破坏程度可以导致所有的数据被删除，或者你的网站和应用程度被用来向客户发送病毒。短期而言，这会让客户的电脑受到感染。长远来看，你的公司可能会列入不安全站点名单。

(作者：Denny Cherry 译者：Shirley 来源：TT 中国)

原文标题：如何保证 SQL Server 免遭 SQL 注入攻击？

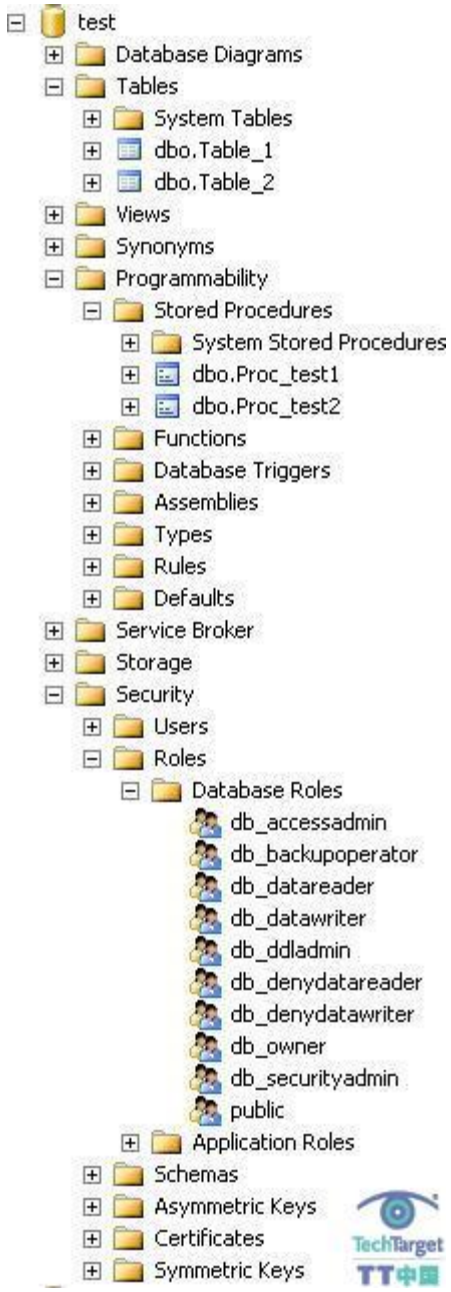
链接：http://www.searchdatabase.com.cn/showcontent_10911.htm

通过数据库角色控制访问（上）

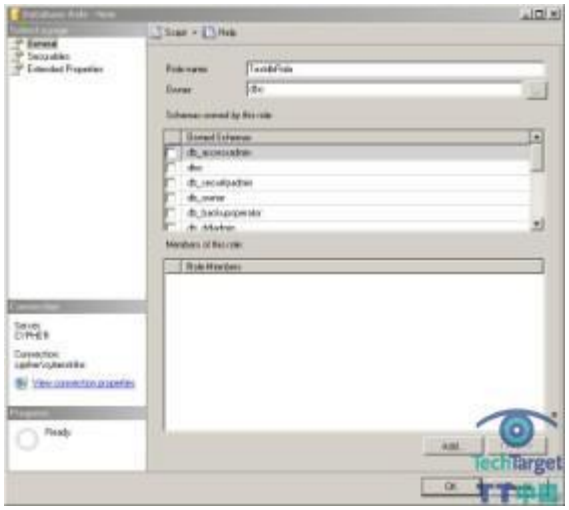
安全性在当今世界中是非常重要的，但是如果不了解现有的控制选项，我们是很难保证SQL 数据安全。在本文中，我将阐述如何设置数据库角色并解释它们是如何帮助我们巩固企业安全性。

数据库角色

数据库角色允许我们定义一个特定的权限集。然后，服务器登录（不管是在 AD 组、AD 用户或者 SQL Server 登录）都将添加到服务器上和数据库角色中。



第一步：右键点击“Database Roles”，选择“New Database Role”。



第二步：点击“Securables”，选择指派到该角色的其它数据库角色、对象等。然后点击“OK”。现在数据库角色就出现在“Database Roles”下的数据库/安全/角色清单中。

第三步：指派权限到数据库角色上。只指派确定需要的权限到数据库角色上。我强烈提倡所有数据库访问都通过存储过程，因为这样能让公司获得如下的好处：

1、编译存储过程可以减少破坏操作的 SQL Server 入侵攻击。

2、编译存储过程允许缓存和重用执行计划，从而减少 IO/CPU/RAM 使用率，因为一次执行计划将以参数化语句存储而不是分别为每次执行存储一个计划。

3、更简单的访问控制。不要对某一个进程可能被访问的表（很可能是 10+表格）授予 select/delete/insert/update 访问权限，我们只需要对合适的存储过程授予执行权限，然后存储过程会给需要的对象授予正确的权限。

下面是这个例子的一些权限授予示例：

- grant execute on [dbo].[Proc_test1] to [TestdbRole]
- grant view definition on [dbo].[Proc_test1] to [TestdbRole]

第一个语句授予"TestdbRole"数据库角色成员执行 Proc_test1 存储过程的权限。第二个语句授予 TestdbRole 对存储过程的查看定义权利。基本上，这就意味着该角色成员现在可以执行程序 Proc 同时查看它的内容。然而，它们没有其它的权利，因为 Proc_test1 查询表 Table_1 时就已经隐含地授予表查询访问权限了。但是，TestdbRole 成员则不能在 Table_1 进行查询、查看定义或者执行任何活动。

如果选择授予直接对象访问权限，那么只需简单地执行“grant select on [object] to [TestdbRole]”。然而，正如前面所声明的，我并不推荐这样的方法。

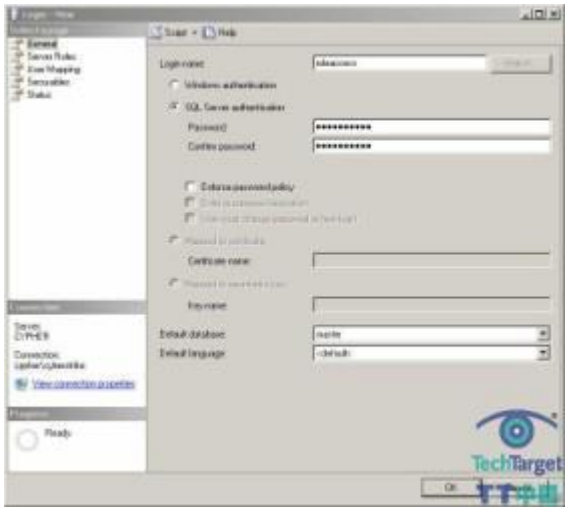
(作者：Matthew Schroeder 译者：曾少宁 来源：TT 中国)

原文标题：通过数据库角色控制访问（上）

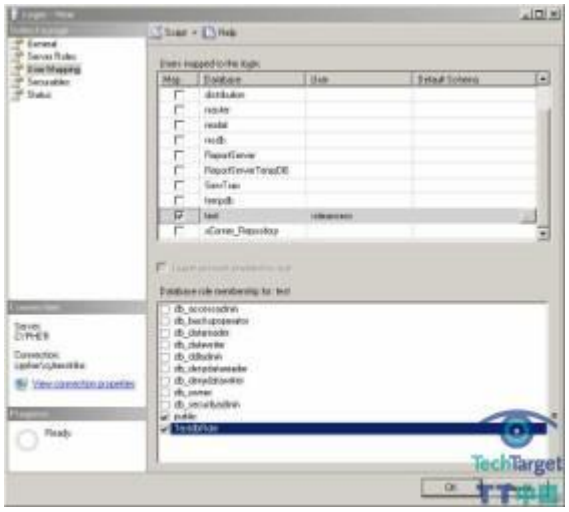
链接：http://www.searchdatabase.com.cn/showcontent_20567.htm

通过数据库角色控制访问（下）

第四步：创建/使用添加到数据库角色的用户/组。在我们所列举的例子中，我们将创建如下图所示的注册/用户"roleaccess"。

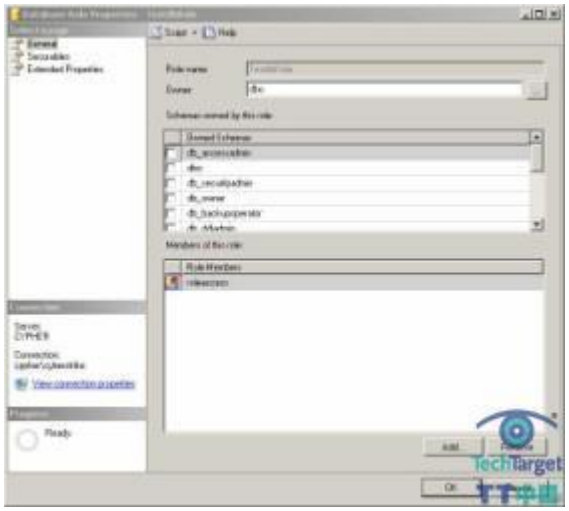


在这里，我们也可以将用户指派到数据库和已经创建的角色上，如下所示。

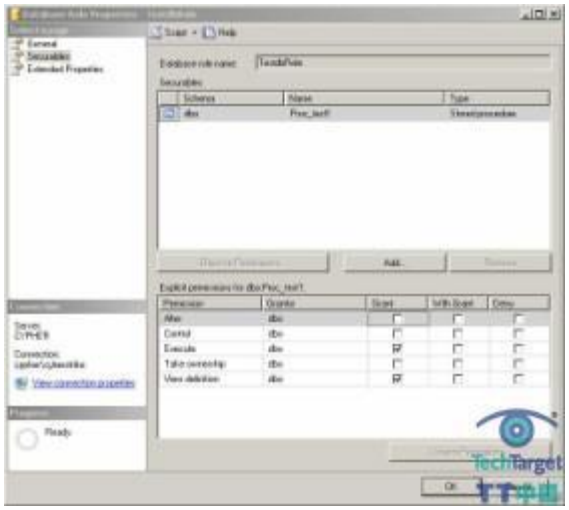


验证角色访问：

我们可以通过检查数据库中的数据库角色来验证访问，如下图所示。右击数据库角色，选择属性，就会显示如下图所示的新添加角色成员的窗口。

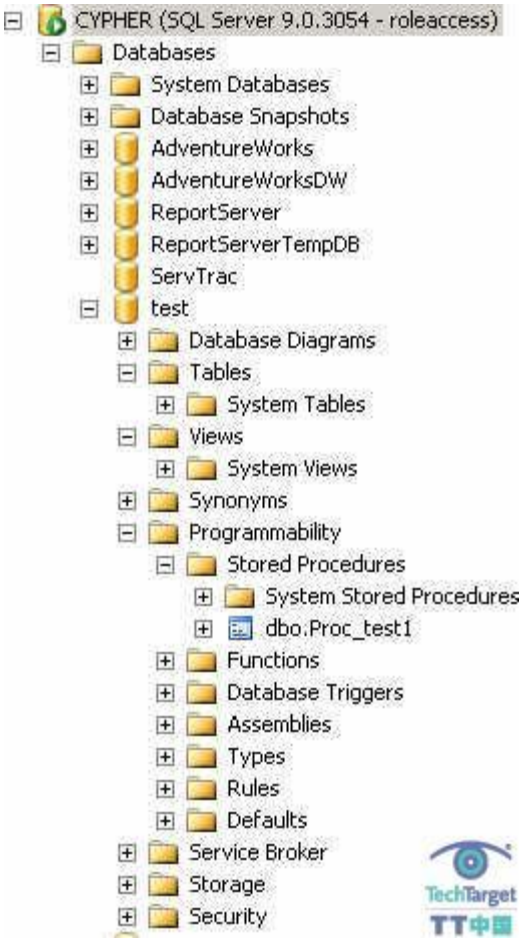


我们可以通过查看 securables 来检查存储过程/其它安全对象，如下所示。



检测数据库角色的安全性

为了查看我们设置的角色访问的实际作用，我们可以使用"roleaccess"用户登陆 SSMS。如下图所示，我们可以看到所有的数据库，但是无法访问它们。我们也只能看到我们拥有访问权限的表或存储过程，而无法看到所有其它的表或存储过程。



我们也可以右击 Proc_test1 并点击修改来查看代码，但是我们无法修改它，如下图所示。

最后，我们可以执行存储过程，但是我们无法查询 Table_1 表。

我希望在本文中你能够学到一些东西。我已经解释了为什么通过存储过程来控制所有数据库访问是最佳方法，以及为何该方法可以帮助提高安全性/性能。同时，我还阐述了如何定义数据库角色、数据库角色是如何提高安全性/可管理性的、如何为数据库角色授予对象权限、如何创建一个登陆、如何指派一个用户到数据库和数据库角色以及如何验证已定义的权限。最后，我还演示了它是如何限制用户访问数据库。请记住，在项目的一开始定义数据库安全性比在之后添加要容易得多。

(作者：Matthew Schroeder 译者：曾少宁 来源：TT 中国)

原文标题：通过数据库角色控制访问（下）

链接：http://www.searchdatabase.com.cn/showcontent_20570.htm

设置 SQL Server Service Broker 进行安全通信（上）

从一个实例向另一个实例发送 SQL Server Service Broker 信息最安全的方法是使用证书以确保通信安全。虽然这是最安全方法，但它比普通 SQL Server 或 Windows 身份验证技术要复杂得多。

为实现本文的目的，我们将从购物车数据库向购票系统数据库发送信息。购物车数据库安装在 SQL1 服务器上而购票系统数据库安装在 SQL2 上，两者在不同的服务器上。所有的这些声明全部运行在每个实例的主数据库上。

首先需要设置每个实例的主密钥。使用 CREATE MASTER KEY 声明可以实现。这个密码是实例加密的关键，所以要用一个复杂的。创建主密钥成功后，你需要修改并配置它以便由服务主密钥加密。服务主密钥是由 SQL Server 实例来生成和维护的。这就允许 SQL Server 实例在不设密码的前提下为主数据库打开任何数据库主密钥。

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD ='YourVeryLongSecurePassword!'
                                    GO
                                ALTER MASTER KEY
                            ADD ENCRYPTION BY SERVICE MASTER KEY
                                    GO
```

主密钥创建成功后，你就可以用 CREATE CERTIFICATE 声明来创建证书了。你要为证书来指定名称、主题、起始日期和终止日期。这将基于数据库主密钥来创建一个一个证书而不是基于密码。如果实例被许多应用程序共享，那么你应该用一些更加通用的名称。

```
                CREATE CERTIFICATE ServiceBrokerCert_SQL1
            WITH SUBJECT = 'Service Broker Certificate',
                    START_DATE = '1/1/2009',
                EXPIRY_DATE = '12/31/2099'
                                GO
```

在第二个实例中创建不同名称的证书。

```
CREATE CERTIFICATE ServiceBrokerCert_SQL2
WITH SUBJECT = 'Service Broker Certificate',
START_DATE = '1/1/2009',
EXPIRY_DATE = '12/31/2099'
GO
```

证书创建的工作完成后，你需要创建并配置 Service Broker 端点以便验证。你可以用 CREATE ENDPOINT 声明来完成这一动作，指出要用 FOR SERVICE_BROKER 选项并给出指定证书。在这个例子中，要求通信加密，并使用 RC4 加密法则。

可供选择的加密方案有以下几种：

RC4——在实例中使用 RC4 加密法则配置端点。

AES——在实例中使用 AES 加密法则配置端点。

AES RC4——先使用 AES 加密算法配置端点，然后使用 RC4 加密。

RC4 AES——先使用 RC4 加密算法配置端点，然后使用 AES 加密。

```
CREATE ENDPOINT [ServiceBrokerEndpoint]
AUTHORIZATION [sa]
STATE=STARTED
AS TCP (LISTENER_PORT = 1234, ;LISTENER_IP = ALL)
FOR SERVICE_BROKER(MESSAGE_FORWARDING = DISABLED,
MESSAGE_FORWARD_SIZE = 10,
AUTHENTICATION = CERTIFICATE ServiceBrokerCert_Cart,
ENCRYPTION = REQUIRED ALGORITHM RC4) GO
```

（作者：Denny Cherry 译者：孙瑞 来源：TT 中国）

原文标题：设置 SQL Server Service Broker 进行安全通信（上）

链接：http://www.searchdatabase.com.cn/showcontent_22325.htm

设置 SQL Server Service Broker 进行安全通信（下）

尽管 RC4 不如 AES 算法强大，但是速度更快。这意味着在加密解密时不需占用太多的 CPU 资源。一定要注意二者的区别，因为在高负荷环境中使用 AES 算法将占用更多 CPU 资源，同时除非迫不得已，RC4 算法不适合在数据十分宝贵的高风险系统中使用。

设置完所有服务器端点后，你就可以执行证书交换了。先备份证书的公共密匙到一个文件，然后将公共密匙输入到远程服务器。你首先需要使用 BACKUP CERTIFICATE 声明来备份证书。

在这个例子中，我们已经备份了 ServiceBrokerCert_SQL1 证书到 SQL1 服务器 C 盘的一个文件。

```
BACKUP CERTIFICATE ServiceBrokerCert_SQL1 TO
    FILE='C:\ServiceBrokerCert_SQL1.cer'
```

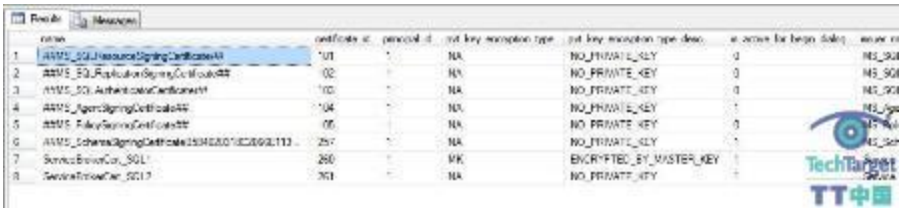
之后我们将 ServiceBrokerCert_SQL2 的证书备份到 SQL2 服务器的 C 盘中。

```
BACKUP CERTIFICATE ServiceBrokerCert_SQL2 TO
    FILE='C:\ServiceBrokerCert_SQL2.cer'
```

文件备份完成后，该把他们移动到其它服务器的 C 盘上了。然后你需要把证书导入到其它服务器。CREATE CERTIFICATE 声明可以实现此功能，但是需要使用 FROM FILE 参数来代替 SUBJECT, START_DATE 和 EXPIRY_DATE 参数了。

```
CREATE CERTIFICATE ServiceBrokerCert_SQL1
FROM FILE='C:\ServiceBrokerCert_SQL1.cer'
```

然后重连 SQL1 并将证书从 SQL2 服务器中导入。通过查询 sys.certificates 系统目录视图，你可以看到证书已经成功导入。请参看下图：



证书导入后，你就可以使用 SEND 声明和 Service Broker 线路在加密的连接中发送消息了。

（作者：Denny Cherry 译者：孙瑞 来源：TT 中国）

原文标题：设置 SQL Server Service Broker 进行安全通信（下）

链接：http://www.searchdatabase.com.cn/showcontent_22327.htm

在 SQL Server 2008 中安装安全审计（一）

SQL Server 2008 引进了一种新的审计性能，它能使 DBA 追踪数据库的使用并进行详细审计。我们能在服务器和数据库上安装审计，在单独的数据库对象上激活并用各种不同的格式保存，如二进制文件或 Windows Application 日志。

在 SQL Server 2008 里安装审计，步骤如下：

- 给每个 SQL Server 2008 具体实例创建一个 SQL Server 审计
- 创建服务器审计规范、数据库审计规范或者其中的一个
- 激活 SQL Server 审计
- 查看审计数据

在这一技巧中，我将复习其中的每一步并举例说名它们怎么进行的。注意大多数情况下这些例子是基于 T-SQL 语句的。但是，这些步骤也能够用 SQL Server Management Studio 界面来执行。如果想了解更多有关 SQL Server Management Studio 用法的信息，请查看 Microsoft SQL Server 联机丛书。

1、创建 SQL Server 审计

第一步你应该在 SQL Server 2008 的一个实例上创建审计，这样就能创建一个 SQL Server 审计。审计就是为与数据库引擎相关的具体事件集合配置的安全对象。你可以在 SQL Server 2008 里的一个实例上创建多个审计。

在创建审计时，你必须给它指定一个名称和事件输出的目标位置。目标文件可以是二进制的文件、Windows Security 日志或 Windows Application 日志。你还可以给审计对象指定一个或更多个可选参数。

你可以用 CREATE SERVER AUDIT 语句，如下所示：

```
USE master
GO
CREATE SERVER AUDIT SrvAudit
```

```
TO FILE (FILEPATH='C:\Data', MAXSIZE=5 MB)
WITH (QUEUE_DELAY = 3000)
```

注意，你必须在主数据库中创建一个审计。由于审计是和 SQL Server 实例相联系的，因此你不能在用户数据库中创建。

第一行的 CREATE SERVER AUDIT 语句仅规定审计名称（如 SrvAudit）。第二行的 TO 子句确定事件输出时的目标位置。例如，我想将输出结果保存在文件中，所以我必须指定 TO FILE 并规定 FILEPATH 值。注意这只是一个路径名。SQL Server 将自动命名输出文件，如下所示：

```
<audit_name>_<audit_GUID>_<partition_number>.sqlaudit
```

在上面的例子中，TO FILE 语句还包括了 MAXSIZE 参数，MAXSIZE 参数将文件大小限制为 5 MB。该参数是 TO FILE 子句可选参数之一。如果你将审计数据迁到 Application 日志或 Security 日志，你就只需要指定日志名选项，示例如下：

```
CREATE SERVER AUDIT SrvAudit2
TO APPLICATION_LOG
WITH (QUEUE_DELAY = 3000)
```

就像你看到的一样，TO FILE 子句已经被 TO APPLICATION_LOG 子句所替代并且还没有另外规定其他的参数。

最后一行在 CREATE SERVER AUDIT 语句中的就是一个 WITH 子句。该子句支持很多个选项，限制了创建审计的方法。在这种情况下，我使用的是 QUEUE_DELAY 参数并将它的值设为 3000。这个参数指定了在创建审计之前要耗费的毫秒数并且。默认数字为 1000 毫秒（即 1 秒）。

要了解所有 CREATE SERVER AUDIT 语句可选择项和本篇文章中的其他语句，请查看 Microsoft SQL Server 联机丛书。

（作者：Robert Sheldon 译者：April 来源：TT 中国）

原文标题：在 SQL Server 2008 中安装安全审计（一）

链接：http://www.searchdatabase.com.cn/showcontent_15650.htm

在 SQL Server 2008 中安装安全审计（二）

2、创建服务器审计规范

你创建 SQL Server 审计之后，必须创建一个服务器审计规范或者是一个数据库审计规范或者是其中的每个。一个服务器审计规范就是和具体的 SQL Server 审计相关的一个或多个服务审计。活动组就数据库引擎暴露出来的一组相关的事件，例如，我们在进行安全审计操作时，SERVER_OPERATION_GROUP 行动组就出现了，如当用户在改变服务器设置时。你可以在每个审计上只创建一个服务器审计。但是，你可以对审计规范增加多个活动组。创建一个服务器审计规范，你需要在主数据库上运行 CREATE SERVER AUDIT SPECIFICATION，如下所示：

```
USE master
GO
CREATE SERVER AUDIT SPECIFICATION SrvAuditSpec
FOR SERVER AUDIT SrvAudit
ADD (SUCCESSFUL_LOGIN_GROUP),
ADD (FAILED_LOGIN_GROUP)
WITH (STATE=ON)
```

第一行 CREATE SERVER AUDIT SPECIFICATION 语句规定了审计规范名（SrvAuditSpec）。第二行 FOR SERVER AUDIT 子句指定了与审计规范相关的审计名（SrvAudit）。第三行和第四行为 增加了规范活动组的 ADD 子句。在这种情况下，我增加了 SUCCESSFUL_LOGIN_GROUP 和 FAILED_LOGIN_GROUP 活动组，跟踪试图登录到 SQL Server 实例的安全主管。

最后一行 CREATE SERVER AUDIT SPECIFICATION 语句为 WITH 子句。WITH 子句包括激活规范的在 STATE 参数。默认值不能激活审计规范（STATE=OFF）。如果你在创建时不能激活审计规范，你就必须过段时间再激活，在你能够审计活动组之前进行激活。

创建数据库审计规范

和服务器的审计规范不一样，数据库审计规范是具体针对数据库的。但是它和服务器审计规范相同的是，你可以增加审计活动组，但是它们仅仅针对数据库。此外，你可以给规范增加单独的审计活动。审计活动就是数据库具体的活动，如删除数据或运行存储程序。

创建数据库审计规范，在目标数据库中运行 CREATE DATABASE AUDIT SPECIFICATION 语句，例如：

```
USE AdventureWorks2008
GO
CREATE DATABASE AUDIT SPECIFICATION DbAuditSpec
FOR SERVER AUDIT SrvAudit
ADD (DATABASE_OBJECT_CHANGE_GROUP),
ADD (SELECT, INSERT, UPDATE, DELETE
ON Schema::HumanResources BY dbo)
WITH (STATE=ON)
```

第一行 CREATE DATABASE AUDIT SPECIFICATION 语句指定了规范（DbAuditSpec），第二行为 FOR SERVER AUDIT 子句，用它可以判断和规范相关的审计。接下来，我增加了一个审计活动组，在这里就是 DATABASE_OBJECT_CHANGE_GROUP。在对 AdventureWorks2008 数据库执行 CREATE、ALTER 或 DROP 语句时就会出现这个活动组。

第二个 ADD 子句制定了单独审计活动，而不是一个活动组。这样，审计活动就是 SELECT、INSERT、UPDATE 和 DELETE。但是你要注意，下面一行包含一个 ON 子句指定的 HumanResources schema 和 dbo 安全主管。结果，只要 dbo 在 HumanResources schema 中查询一个对象或在 AdventureWorks2008 数据库中插入、更新或删除，SQL Server 就会将事件记入日志。

最后，CREATE DATABASE AUDIT SPECIFICATION 语句中的最后一个子句就是 WITH 子句。跟上次一样，你可以在操作完之后就激活审计规范或者过一段时间之后再进行激活。

3、激活 SQL Server 审计

和我们刚刚回顾的审计规范一样，CREATE SERVER AUDIT 语句中的 WITH 子句并不支持 STATE 参数。也就是说你必须在单独的一步中激活审计，如下面的语句中所示：

```
USE master
GO
ALTER SERVER AUDIT SrvAudit
WITH (STATE=ON)
```

你可以看到，我使用的是 ALTER SERVER AUDIT 语句更改我之前创建的 SQL Server 审计（SrvAudit）。ALTER SERVER AUDIT 语句中的 WITH 子句支持被我设置成 ON 的 STATE 参数，SQL Server 会审计这些具体事件。

4、查看审计数据

你可以在 SQL Server Management Studio 中用 Log File Viewer 查看审计数据。另外如果你创建 SQL Server 审计将事件保存到 Application 日志或者 Security 日志，这样你就可以用 Event Viewer 查看这些数据。我认为回顾事件信息最简单的方法就是将审计数据保存到一个二进制文件中，然后用 Log File Viewer 回顾这些数据。

要访问 Log File Viewer，就要打开 SQL Server Management Studio，扩展 Security 节点。接下来选择你要复习的审计，然后点击 View Audit Log 发送 Log File Viewer。图 1 表示以上的例题中 SQL Server Audit (SrvAudit)事件实例。

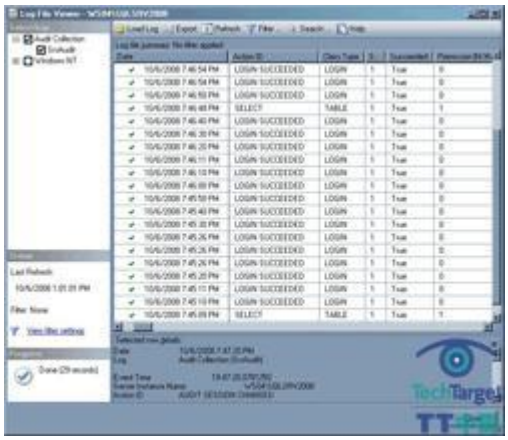


图 1：在 Log File Viewer 中查看审计数据

以上就是安装审计以及回顾审计数据的步骤。SQL Server 2008 让这些步骤比以前版本执行更加简单。你只需要简单创建 SQL Server 审计，并附上一到两个审计规范，然后

激活。其余的工作都由 SQL Server 完成。要了解更多有关审计方面的信息，请查看微软 SQL Server 联机丛书。

（作者：Robert Sheldon 译者：April 来源：TT 中国）

原文标题：在 SQL Server 2008 中安装安全审计（二）

链接：http://www.searchdatabase.com.cn/showcontent_15651.htm

SQL Server 安全审计中的常见疏忽

你的 SQL Server 系统在应对安全漏洞时的表现如何?你是否已经打了相应的补丁?它们是否因此而变得“坚不可摧”?在密码方面呢?它们是否可以满足复杂的需求?你是否会有规律地修改密码?你是否启动了审计日志功能呢?

针对以上的这些问题，如果你的回答大部分是肯定的，也千万不要自我感觉良好。事实上，针对这样的情况，十个人中会有八个人会盲目乐观，但不幸的是，问题还存在，而且很多很严重。

你肯定不会相信，我见过多少漏洞百出的 SQL Server 等待着被攻击，其中还都自称是开启了安全审计功能的。这其实都是清单审计的问题，虽然听上去是很安全的，但只要你稍微深入一点观察，就能发现丑陋的真相。下面就是一些 SQL Serce 的安全漏洞，只需一点点时间，这些漏洞就可以让你的系统面目全非：

- C 盘共享的服务器。我经常能看到 SQL Server 系统提供 C 盘的完全共享以及全部用户的 NTFS 访问权。如果 SQL Server 没有启动，那么 master.mdf 文件就可以被访问，而这个文件是第三方工具用来进行 SQL Server 密码破解或重置时所需要的。

SQL Server 启动之后，黑客往往会利用其他的漏洞来访问系统(比如密码弱或补丁不完整)，而不是访问 master.mdf 文件。

- 未受保护的磁盘镜像和整个 SQL Server 系统的基础文件夹级备份。这是同上面一个问题相关的，不安全的共享以及 NTFS 许可。
- 未受保护的 SQL Server Express 实例以及工作站中运行的 MSDE。这个看上去并不重要的数据库其实存储了十分重要的数据，而且不应该暴露给内部人士。想要找到网络中活动的 SQL Server 系统，唯一的方法就是使用一个好的漏洞扫描工具，比如 QualysGuard 或 LANguard。

一个更加简单直接的方式就是使用 SQLPing3 工具，它的功能不仅仅可以找出默认的 SQL Server 安装，而且还可以找出异常配置并锁定 SQL Server 实例。

- Web 应用没有正确验证输入并有利于 SQL 注入。即使是最安全的 SQL Server 控件也没有办法避免这类攻击。在最近的一次 Web 应用安全鉴定中，我发现一个 SQL 注入漏洞如果拥有 SQL Server 的完全访问权，就可以轻松地让大部分数据库安全控件失效。

想要攻击一个看似安全的 SQL Server 系统，可以有无数种选择。现实的情况是，在 SQL Server 上执行更高级别的安全审计实际上是失去先机。如果你想要看清楚你的数据库安全状态，就必须检查每一个级别。

从操作系统到 Web 应用，以及其他所有的东西，你都必须考虑到。否则你可能就离被攻击不太远了。

（作者：Kevin Beaver 译者：孙瑞 来源：TT 中国）

原文标题：SQL Server 安全审计中的常见疏忽

链接：http://www.searchdatabase.com.cn/showcontent_30524.htm

加强 SQL Server 安全性：网络安全

很多组织都存在安全漏洞，不管他们采取怎样的措施来保障他们的环境。在数据库系统中，整体系统上任何一个地方出现漏洞都能够被利用来获取重要的信息。

为了正确的保障 SQL Server 安全性，构成 SQL Server 安全性的以下几个层面都是需要重点考虑的：

1. 网络安全（防火墙、端口、加密）

2. 操作系统安全性（Windows 安全性）

3. 服务器层安全性（终端点、服务器登录、端口、协议以及其它外部领域配置）

4. 数据库层安全性（授予权限给登录身份/角色、加密选项以及确定恰当的权限）

由于这些层面的任何一个位置出现安全错误都意味着整个解决方案的失败，因此，假如攻击者只需要嗅探到某些“通过网线”的数据了，或者是操作系统出现漏洞了，那么也就无法保证每个对象层面的安全性。因此，所有层面都必须确保安全。

这个系列的第一部分阐述与 SQL Server 相关的网络和操作系统安全性。数据库和服务器层安全性将在第二部分进行探讨。

网络安全性

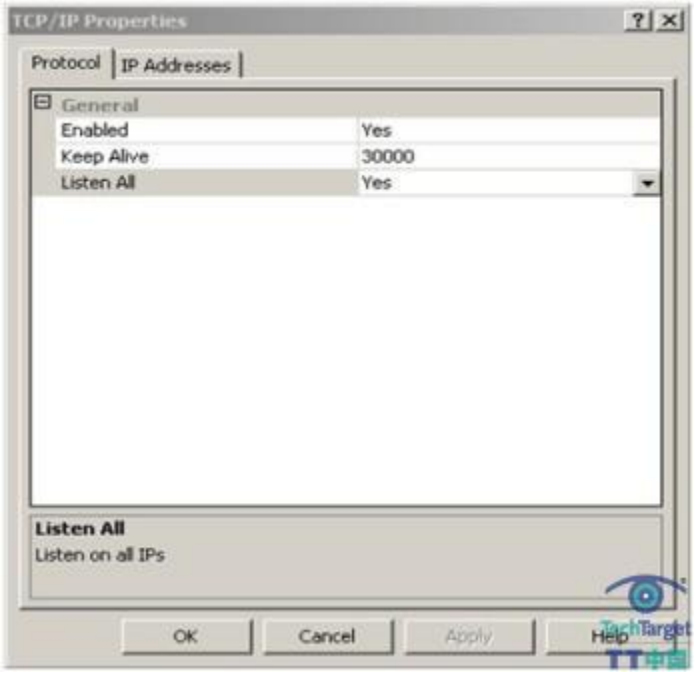
虽然网络安全专家主要负责网络安全，但是有部分 SQL Server 配置选项也与这一层有关系。

网络安全专家主要关注哪些端口打开了以及使用了哪些协议。网络安全专家和 SQL Server 管理员必须探讨是否每个 SQL Server 必须具备不同的端口或者它们必须共享一个端口。（我强烈建议不使用标准 1433 端口，因为该端口有很高的受攻击系数。）

在 SQL Server Configuration Manager （图 1）中，SQL Server Network Configuration 选项卡包含实例（S2K8 是一个实例）所使用的协议/端口的控制。

这是开发数据库的默认配置。对于 SQL Server Enterprise Edition，唯一应该打开的协议是 TCP/IP。所有其它协议都应该禁用，除非有特别的应用需求要求在客户端打开它们。

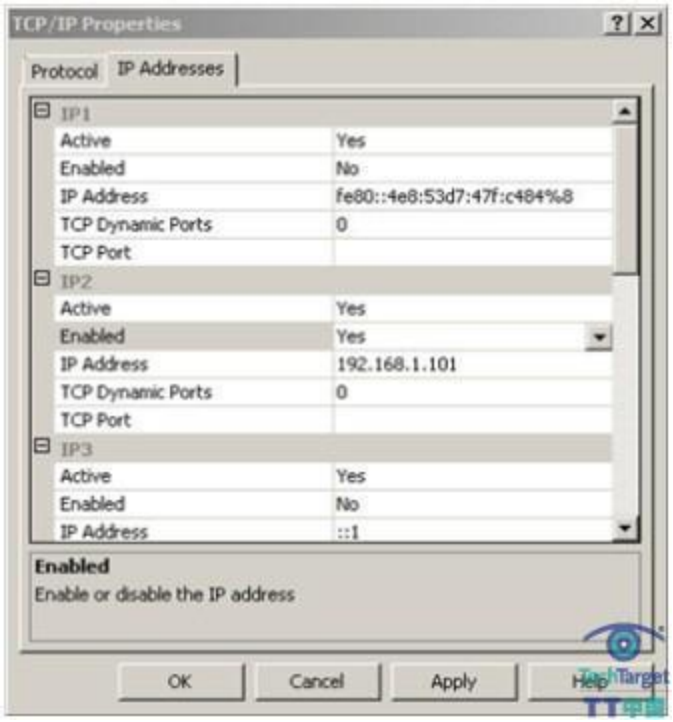
在 TCP/IP Properties 选项卡中有几个重要的属性。图 2 显示的“Listen All”属性，决定 SQL Server 实例应该监听的端口号。



如果属性设置为“Yes”并且有多个 IP 地址，SQL Server 实例将监听所有的 IP。如果只有一个 IP 地址，那么就置空这些设置。

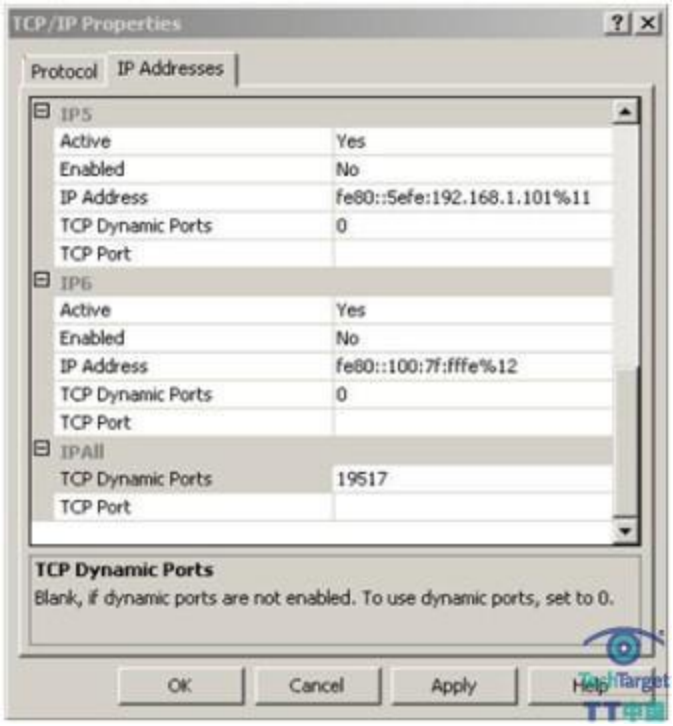
然而，如果服务器有多个 IP 地址，“Listen All”属性应该设置为“No”，SQL Server 实例应该设置监听需要的 IP。一个实例监听的 IP 地址越多，可能被攻击的概率越大。

图 3 显示如何限制 SQL Server 实例到一个具体的 IP 地址。如果“Listen All”属性设置为“No”，那么“Enabled”属性就会应用。“Enabled”应该只有在需要的 IP 地址上设置为“Yes”。



监听的具体端口号可能由 TCP Dynamic Ports 或每一个 IP 或 IPAll 上的 TCP Port 设置来设定。

在图 4 中显示的是设置 IPAll 以及动态端口。这允许端口号在 SQL Server 启动时根据可用的端口进行变化。



设置动态端口时，SQL Server Browser Service 会监控使用的端口并将入口连接导向到一个特定实例的当前端口。

图 5 是关闭了动态端口，并在 IPAll 定义了静态端口。

注意 IP 端口也可以为每一个 IP 地址设置，并且每一个 IP 地址可以监听多个端口，每个端口用逗号隔开（1600，1700）。

（作者：Matthew Schroeder 译者：曾少宁 来源：TT 中国）

原文标题：加强 SQL Server 安全性：网络安全

链接：http://www.searchdatabase.com.cn/showcontent_28603.htm

加强 SQL Server 安全性：操作系统安全

操作系统安全性

虽然大多数 Windows OS 安全性是由 Windows 自身处理的，许多 SQL Server 都默认具备 SysAdmin 的 BUILTIN\Administrators 组。这给每一个物理设备上的 Windows 管理员提供了 SQL Server 的 SysAdmin 权限。

通常，这不是最好的方法，因为一般生产 DBA 团队是与维护生产 Windows 服务器的团队分开的，甚至可能来自不同的公司。更好的做法是在确定知道 SysAdmin 密码或有其它拥有 SysAdmin 权限的帐号后删除默认的 Windows 组。

总之，一定要考虑在一个环境中不同的系统是如何交互以及在整个环境中是如何使用的。

(作者：Matthew Schroeder 译者：曾少宁 来源：TT 中国)

原文标题：加强 SQL Server 安全性：操作系统安全

链接：http://www.searchdatabase.com.cn/showcontent_28604.htm

加强 SQL Server 安全性：服务器与数据库安全

SQL Server 的安全性是由多个层次构成的：网络、操作系统、服务器以及数据库。这些层次需要分别进行加强，因为它们其中任何一个出现问题都会导致整个方案的失败。

网络和操作系统安全已经在本系列文章的第一部分讨论过。本文将探讨数据库和服务器层面的保护。

服务器安全属性

在服务器属性选项卡中，可以通过 xp_cmdshell 指定一个“服务器代理帐号”。我再次强烈推荐，如果使用 xp_cmdshell，我推荐给代理帐号优先较低的权限。

特别地，服务器属性中有 2 个值得注意的设置：Enable C2 审计跟踪和 Enable Common Criteria 规范。

Enable C2 审计跟踪允许跟踪数据驱动的对象访问——包括成功或失败的，除非有专门的控制或服务器关闭，它会随着时间一直增长。

Enable Common Criteria 规范会导致内存在被重新分配前被重写。虽然这能够增强系统的安全性，但内存的申请会减慢。而且，这个选项也会改变登录审计和权限设置。因此，在打开这个选项之前，应该要小心设计数据库权限，以及性能需求。

SQL Server SSL 安全性

SQL Server Configuration Manager 控制所有 SSL 使用。

如果安装了证书，右击“Protocols for S2K8”，然后选择“Properties”，就会出现下面的屏幕。

在“Flags”和“Certificate”选项卡中，可以指定是否需要强制加密。

在一个高度安全的生产系统，最好是选择强制加密。我推荐在生产系统中设置“Hide Instance”为“Yes ”。这是因为在一个生产系统中允许 SQL Server 浏览服务定位一个实例会使定位和攻破一个生产 SQL Server 变得更容易。

而且，在生产环境中，最好避免使用自有签名证书，因为它可能被中间人攻击所攻破——中间人攻击是通过单独连接每一个目标并延迟它们之间的消息来进行窃听的攻击方式。在开发/QA 环境中，这将取决于系统所需要的工作安生性级别。

SQL Server 终端安全性

当一个终端创建后，要保证状态只有在需要时才启动，同时使用一个非标准端口(通知网络组端口号以及使用的 TCP 协议)和允许/要求的加密。

只有需要的端口才应该打开，并且只打开要求的协议。假定网络不是加密流量，我建议设置加密为“required”。在下面的情况中，其它终端将必须设置为必需的(或支持的)。

```
CREATE ENDPOINT endpoint_mirroring
```

```
STATE = STARTED

AS TCP ( LISTENER_PORT = 8035 )

FOR DATABASE_MIRRORING (

AUTHENTICATION = WINDOWS KERBEROS, ENCRYPTION = SUPPORTED,

ROLE=ALL);
```

下一步是给终端赋予权限。考虑到安全性问题，我建议不要使用基本的认证。同时还应该限制连接访问到最少的登录可能。更好的做法是，使用仅限于这个终端或在企业中使用范围很小的登录身份。

```
GRANT CONNECT on ENDPOINT::endpoint_mirroring TO [Domain\ConnectingUser];

GO
```

SQL Server 服务帐号安全性

每一个 SQL Server 服务应该用不同的 Active Directory (AD)帐号安装。绝不应该给一个 SQL Server 服务单独一个 AD 帐号域管理员。

这是因为人们都会有懒性而牺牲安全性来简化使用方法。每一个 SQL Server 服务都应该使用不同的帐号，因为每一个服务都使用要求锁定不同权限的不同的功能领域。另一方面，你的 SQL Server 服务 AD 帐号也可能不需要访问 Internet 或交互登录的权限。

在数据库系统中，整个系统的任何一个方面的缺陷都可能被利用来获取重要系统的访问。因此，重要的是要考虑各个系统层——网络、操作系统、服务器和数据库——是互联的。

(作者：Matthew Schroeder 译者：曾少宁 来源：TT 中国)

原文标题：加强 SQL Server 安全性：服务器与数据库安全

链接：http://www.searchdatabase.com.cn/showcontent_28698.htm