

# Oracle高级安全性

*密钥管理，数据加密和整合性检查*

Oracle技术白皮书

2001年8月

# Oracle高级安全性

## 简介

本文将介绍在Oracle高级安全性选项中所实施的，Oracle所固有的网络加密和数据整合性保护。它检验相关的安全性威胁，解释加密和整合性检查处理的细节，并说明所需要的安装和配置工作。

除本文将介绍的加密和整合性功能之外，Oracle高级安全性还利用加密套接字协议层(SSL)保证Oracle网络安全，并利用SSL为Internet Inter-ORB Protocol (IIOP)提供保证。这些实施（它们不在本文涉及的范围之内）可以使用户在公共的关键性基础设施(PKI)中运转。这种选择余地意味着安全的网络遵循用于实施SSL的互联网工程工作小组(IETF)标准。而且，Oracle高级安全性可利用加密和数据整合性检查来维护瘦Java 数据库连接性(JDBC)客户端的通讯。本文并不涉及这些特点，它只限于介绍在Oracle高级安全性中检验Oracle所固有的网络加密和整合性检查的内容。

## 安全性威胁

在当今互联网的计算机时代，许多有价值的和敏感的数据通常是通过不安全的方式和渠道在传播。任何时候，当信息不论是互联网还是在一个组织的内联网等网络中传输时，它们都是易于受到攻击的。对数据的非法窃取正以加速的速率展开。一个可以通过以太网插口或一个布线室访问网络的个体可以监视和获得数据。硬件数据嗅探器不再是笨重和昂贵的了。PC卡插到PC机中可以有效地监视网络的通信量和信息包的内容。

然而更容易做到的，因此也是更具威胁的是基于软件的嗅探器。数据嗅探软件可以从互联网上下载，它可以把网络的通信量和个人信息包的内容暴露给网络中的任何人。在网络中任何人都可免费使用的一些软件中就有网络嗅探器，它可在几乎所有的协议中非法获得通讯信息。网络中不怀好意的用户可以将这种软件下载到他/她的PC中，然后立刻就能把它转化成信息包的嗅探器。一旦取得这种访问，攻击者可以偷窃敏感的归私人所有的信息，它甚至可以破坏一个企业的运转。

所有的数据都面临着被攻击的风险：所有的信息包都可被读，修正或删除。Oracle高级安全性的网络安全性特性注重了这些方面，并保证在网络中传输的所有数据的隐私性和整合性。Oracle高级安全性通过使用加密和数据整合性检查，保卫着所有通信量和信息内容的安全。

## 第一部分：加密和钥匙键管理

Oracle高级安全性中的加密功能将所有的明码文本数据转化为密码文本。一旦被加密，密码文本将以密码的方式在网络中传递，在这种方式中，如果没有正确的钥匙键就几乎不可能把密码文本转换回相应的纯文本。这些加密服务可以防止那些能够访问你的网络的人偷窥你的数据。Oracle高级安全性提供了三种著名的，经过验证的计算方法可以加密Oracle的网络通讯量：

- 数据加密标准（DES）
- Triple DES (3DES, 2Key 和 3Key)
- RC4 ( 40-, 56-, 128-, 256-)

每一种计算方法都允许选择钥匙键的长度。钥匙键的长度越长，将会产生更强大的加密。Oracle高级安全性提供了56位钥匙键的标准DES和40位钥匙键的DES。它也支持带有112位钥匙键的2-key Triple DES和168位钥匙键的3-key Triple DES。最后，它提供带有256-, 128-, 56- 和 40-位钥匙键的RSA Data Security Inc.的 RC4。

美国出口管理条例（EAR）已经做出变更，允许Oracle公司将Oracle高级安全性的一个版本在全球范围内销售。该产品的早期版本只提供较短钥匙键长度的版本，而且根据用户所在国家的不同而不同。目前版本包括了所有计算方法和钥匙键长度，以前该版本只适用于美国本土。而且，现在使用该产品早期版本的用户可以升级到该产品的国内版。

对Oracle高级安全性的正式评估已经开始实施，着重于它固有的加密实施。它可以在遵守U.S. Federal Information Processing Standard (FIPS) 140-1 Level 2（美国联邦信息处理标准（FIPS）140-1 第二级）的模式下进行配置。遵守FIPS 140-1标准表示该软件已经通过了独立专家的评估，可以保证加密技术运用适当。Level 2是一个软件产品所能获得的最高级别。

### 钥匙键的生成

加密技术是建立在产生加密钥匙键的基础之上的。“秘密”钥匙键只有发送方和接收方掌握。这些“秘密钥匙键”在网络事物处理的一端对信息加密，而在另一端解密信息。这种加密处理的安全性高度依赖于所生成的秘密钥匙键。

The process of key production and data encryption consist of the following steps:

钥匙键的生成过程和数据加密过程由以下步骤组成：

1. 计算方法的交流

2. 在客户端和服务端产生充足的随机数
3. 利用Diffie-Hellman公共密钥交换技术生成第一话路密钥
4. 利用Oracle密码协议和Diffie-Hellman密钥重叠，验证用户并生成第二话路密钥
5. 全数据流的加密

## 计算方法的交流

处理的第一阶段是客户端和服务端计算方法的交流。

在客户端和服务端的初始连接阶段，客户端将把所支持的计算方法的清单传递给服务器端，并通过交流决定将采用的计算方法。在初始配置系统时预先选定的计算方法将取代该处理。然后进行交流，以保证客户端和服务端安装有相同的计算方法。系统对安装的计算方法和在客户端与服务器的配置文件SQLNET.ORA中系统管理员所配置的顺序进行交涉。Oracle建议的配置是(i)首先是用户选择的计算方法，(ii)第二是由高到低密钥的顺序。Oracle高级安全性提供了下列的加密计算方法：

- RC4 256
- RC4 128
- RC4 56
- RC4 40
- 3DES 168
- 3DES 112
- DES 56
- DES 40

例如，考虑一个客户端/服务器连接，其中没有任何一方指定了必须使用的计算方法。假如客户端具有RC4 256，RC4 128和RC4 56，而服务器端只有RC4 56和DES，交流将缺省使用对两个处理所共同拥有的计算方法和密钥的长度。在本例中，它们将使用RC4 56。这种方法可以使具有不同计算方法集的各方，只要他们能共享一个公用的计算方法就能实现连接。

一旦计算方法被商定，Oracle高级安全性将产生秘密密钥以便对数据进行加密和解密。在生成足够的密码的过程中，密钥的产生过程是极为重要的。密钥越安全，破解密码的难度就越大。密钥强度的一个因素是用于生成密钥的随机数。

## 随机数的产生

Oracle高级安全性使用Diffie-Hellman的交换方法产生话路密钥。该处理由客户端和服务端双方生成随机数开始。根据重复的序列，随机性可有助于免遭攻击。用户定义的

SQLNET.ORA配置文件参数“sqlnet.crypto\_seed=n8xi6svxlsu3bc”就是生成随机数的种子之一。其他的种子包括系统时间和一个基于操作系统（OSD）的变量。

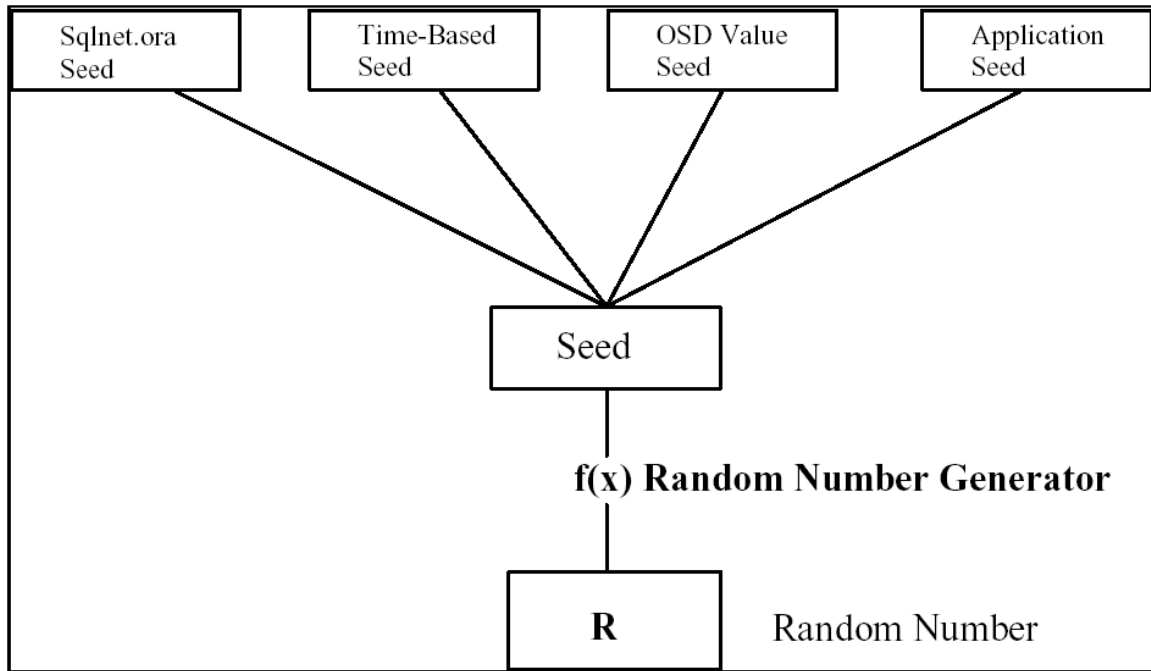


图2. 生成随机数的种子

OSD的值可能基于操作系统中有效的硬件生成器。一旦这些种子有效，随机数生成器将产生一个最终的随机数。由于硬件随机数生成器缺乏统一的标准，包括ORACLE在内的许多加密系统都依赖于PRNG生成加密钥匙键。该过程在相互连接的双方产生，从而使客户端和服务端都可以生成它们自己独特的随机数。

## 生成Diffie-Hellman话路钥匙键

一旦随机数被生成，Diffie-Hellman计算方法将产生话路钥匙键。该钥匙键将在客户端和服务端同时产生，无需通过网络传递。话路钥匙键可利用方程式 $g^c \bmod p$ ,  $g^s \bmod p$  and  $g^{sc} \bmod p$ 来产生，其中

**c** = 客户端生成的随机数

**s** = 服务器端生成的随机数

**g** = 已经在服务器端表示的数量级在300-512位的一个基数

**p** = 已经在服务器端表示的数量级在300-512位的一个模数

值**s**和**c**是通过三个种子，在客户端和服务端分别产生的随机数。值**g**和**p**是已在服务器端存在的值，但可以通过命令`naegen`进行选择。

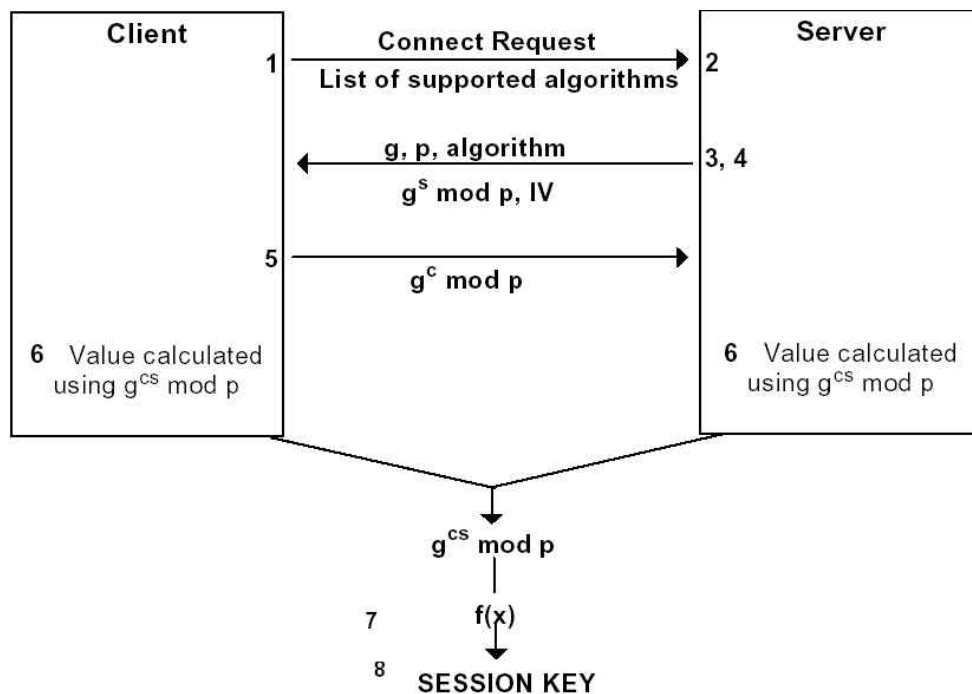


图3. Diffie-Hellman 公共密钥交换

密钥生成的传输顺序通过图3所示客户端/服务器端方框中的小数字来确定，如下所示：

1. 客户端根据所支持的计算方法的清单传输一个连接请求。
2. 服务器端根据第一符合的计算方法和客户端的密钥大小决定将采用的计算方法。
3. 服务器利用服务器端的随机数 $s$ 计算值 $g^s \bmod p$ 。
4. 服务器向客户端传输值 $g, p, g^s \bmod p$ ，即将采用的计算方法，以及初始向量IV。
5. 客户端利用随机数 $c$ 计算 $g^c \bmod p$ ，该值将被传递到服务器。
6. 现在双方都已经获得了值 $g, p, g^s \bmod p$ ，和 $g^c \bmod p$ ，因此可以计算 $g^{cs} \bmod p$ 。
7. 客户端和服务器端都对 $g^{cs} \bmod p$ 应用一个函数以生成话路密钥。该函数根据所选择的计算方法的不同而不同。选择Triple DES与选择RC4将导致不同的计算方法。
8. 话路密钥在客户端和服务器端被建立起来。

密钥交换的最终结果是主话路密钥。主话路密钥可被用于加密所有的网络通信。但是，该密钥容易受到网络中人的攻击，从而威胁到系统的安全性。

## 网络中人的攻击

如果有人正在监听网络通讯，他/她不可能捕获话路密钥，因为它不会通过导线来传送。

但是，在密钥交换设计中存在一个缺陷，即易受到“网络中人”的攻击的薄弱环节。在网络中的人唯一可以得到的信息是交换的元素： $g$ ， $p$ ， $g_s \bmod p$ ，和 $g_c \bmod p$ 。如果拥有Diffie-Hellman计算方法，他/她就能利用这些值去生成两个密钥，一个用于客户端另一个用于服务器端。尽管这两个密钥可能不同，但第三方仍然可以伪装成好似客户端之于服务器端或服务器端之于客户端。第三方可以侵入通讯中，并愚弄客户端和服务端以获得需要继续交流的信息。

如果第三方能够仿效客户端和服务器端并拥有Diffie-Hellman计算方法，它就能生成两个密钥，一个用于客户端另一个用于服务器端。尽管这两个密钥不同，第三方仍可愚弄客户端和服务端。

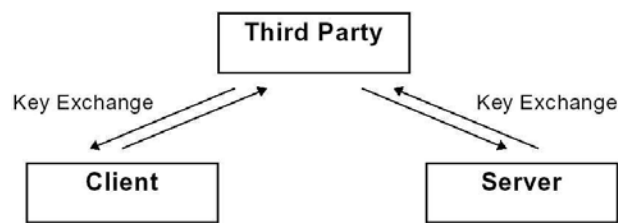


图4. 网络中人的攻击

为了防御这种情况的发生，Oracle高级安全性使用了一个认证密钥的重叠，它是将两个独立的话路密钥并入一个密钥中。

两个原始的密钥是：

1. Diffie-Hellman主话路密钥
2. 由Oracle密码协议（O3LOGON）生成的话路密钥

## O3LOGON 与Diffie-Hellman 密钥重叠

一旦计算方法的交涉和Diffie-Hellman公共密钥产生，用户就会作为登录处理的一部分被验证。该过程被称为Oracle密码协议或O3LOGON。O3LOGON是一种查询-响应协议，它利用DES加密技术保护用户的密码。它是Oracle数据库用于验证基于密码用户的标准协议。在本文当中，应特别注意O3LOGON的两个输出结果。首先，数据库已经验证了客户端的用户。第二，客户端和服务端双方都把随机数 $R$ 看作O3LOGON处理的结果。一旦O3LOGON处理产生， $R$ 值就会与话路密钥重叠以生成一个新的密钥。这种结合两个密钥的处理可用来抵御在前面“网络中人”章节中所述攻击。

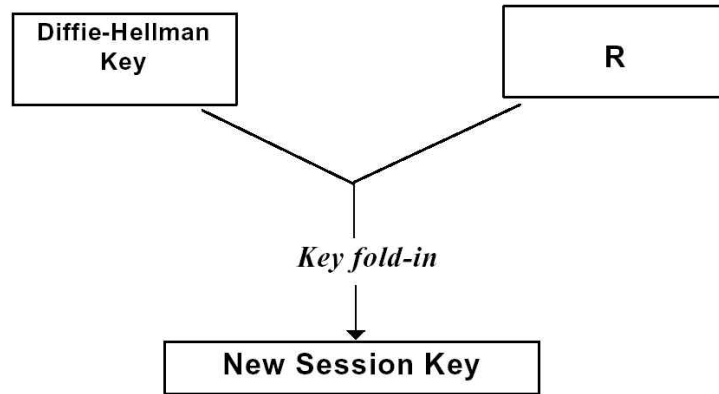


图5. 钥匙键重叠

“网络中人”的攻击允许第三方在线监听客户端和服务端，在利用Diffie-Hellman交换传输它们的钥匙键值时迷惑各方。一旦O3LOGON发生，数值R就被客户端和服务端掌握，而且不会被窃听者发现，因为R本身永远不会由它自身来传输。通过Diffie-Hellman话路钥匙键与参数R的结合，新的主话路钥匙键就不会通过网络来传输，因此可以防范“网络中人”的攻击。

## 全数据流加密

完成用户认证和产生Diffie-Hellman钥匙键重叠后，在服务器与客户端之间发送的信息就被加密。利用所生成的钥匙键和所选择的计算方法，在服务器端的软件就可以在通过网络发送数据前对它加密。同时，客户端的软件将产生钥匙键以完成对数据的解密。这些钥匙键只在该次通讯话路中有效。Oracle的高级安全性管理所有用于加密的钥匙键，而且该过程对用户是完全自动的和透明的。而且，该处理的发生同时得到优化，因此不会对性能产生大的影响。

## 系统表的攻击

如果用于Diffie-Hellman交换的数值p和g在较长的时间内保持不变，攻击者可以利用一个强化的系统表攻击生成一个带有可能钥匙键值得表。因此，我们强烈建议以月为单位利用“naegen”命令变更这些值。

该命令改变了数值p和g的位尺寸。对于40位的钥匙键，给定到naegen中的值必须大于300位，而对于56位的钥匙键则必须大于512位。否则，就使用缺省的位尺寸300和520，而攻击者就会猜到数值p和g。该命令的执行应该非常小心，因为变更数值p和g可能会花费4分钟。Oracle建议采用自动的方式变更这些值，在UNIX上，可以把它们装载到CRON job，在Windows上可采取生成批文件的方式运行。

## 第二部分：数据整合性检查

对数据流的攻击方式有三种：(i) 修正信息包，其中信息包的内容被改变，(ii) 破坏信息包，其中信息包被消除，以及(iii) 追加信息包，其中信息包被追加到数据流中。数据整合性检查，也被称为加密检查，提供了两种方式以保护信息包免受这种攻击的困扰从而保证数据的整合性：排序和散列。

在Oracle高级安全性中的两种数据整合性法则可以保护网络事务处理免遭这类攻击：

- MD5: 信息汇总
- SHA-1: 安全散列法则

### 排序

每一个通过网络传送的信息包都会有一个顺序标签，它可以保证信息包能够按照正确的顺序发送和接收，并且保证在传输过程中没有插入或删除信息包。例如，在网络中传递的第一个信息包可以被标注为A，下一个是B，再下一个是C，以此类推（顺序标签远比A，B，C复杂，但道理是一样的）。当信息包到达目的地，会检查顺序标签以察看是否有丢失，改变或追加信息包的情况。如果某些信息包的顺序被打乱或出现一些匿名的信息包，Oracle高级安全性将发出一条错误并终止连接。在SQLNET.LOG文件中会出现一条终止的记录。例如，一个合理的信息包增加了某人的薪水，而攻击者用更高的薪水替代了另一个信息包，排序机制将能够检测到欺诈信息包扰乱了顺序并终止连接。该处理利用了被称为keyed-MD5的keyscheduling机制。

事实上，排序并不受标签的影响，而是受由RC4计算方法所产生的一系列值的影响，这些值通过钥匙键的顺序来加密。在每一个新的连接中，利用有重叠处理所生成的主话路钥匙键可以客户端和服务端都生成一个同样的钥匙键清单。一旦钥匙键的顺序被建立，客户端和服务端都有同样的钥匙键清单： $k_1, k_2 \dots k_n$ 。这些钥匙键将依次地被使用，为每一个信息包生成顺序标签。

例如，第一个信息包使用 $k_1$ 生成顺序标签A，下一个信息包使用 $k_2$ 等等。

**$RC4(0000000)_{k_1} = \text{顺序标签 A (用于第一个信息包)}$**

**$RC4(0000000)_{k_2} = \text{顺序标签 B (用于第二个信息包)}$**

当信息包到达服务器端，服务器利用它的钥匙键清单为顺序标签解码。当第一个信息包到达服务器，服务器用 $k_1$ 为顺序标签A解码。当第二个信息包到达，服务器用 $k_2$ 为顺序标签B解码。钥匙键的数目是有限的，因此当最后一个钥匙键被使用，顺序重新从 $k_1$ 开始，并在

钥匙键的顺序中开始一个循环。如果解码不能完成，即用于加密的钥匙键不正确，服务器将判定该信息包是欺诈信息包，此时连接将被终止。

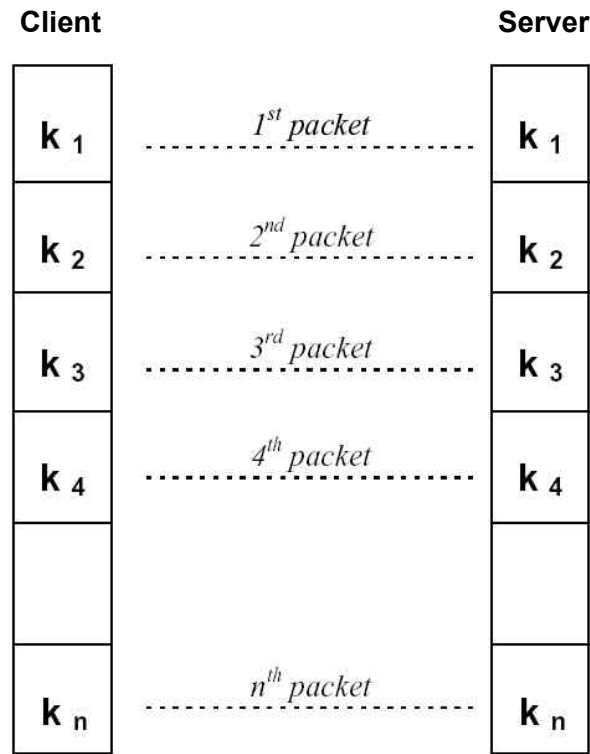


图6. 钥匙键顺序

## 加密散列

当信息包在网络中传递中，攻击者可以改变信息包的内容。例如，如果事务处理更新了表中的一个值，攻击者可能会改变包含有该值的信息包并把它重新放回数据流。

一旦信息包的顺序被建立，MD5的计算方法将保证在传输过程中实际信息包的内容不会以任何方式被改变。由MD5计算方法所执行的加密散列将信息包的内容和顺序相结合，并生成一个散列值，也被称作校验和或汇总，而且会被加密并被记录在Oracle网络信息包的最后。

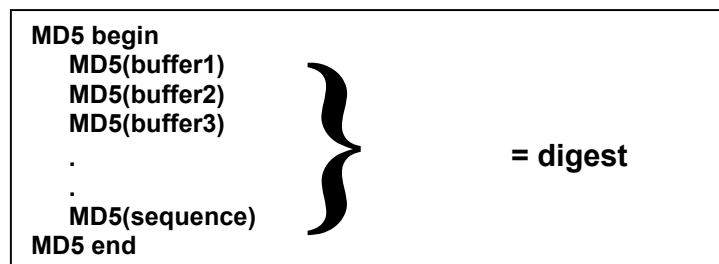
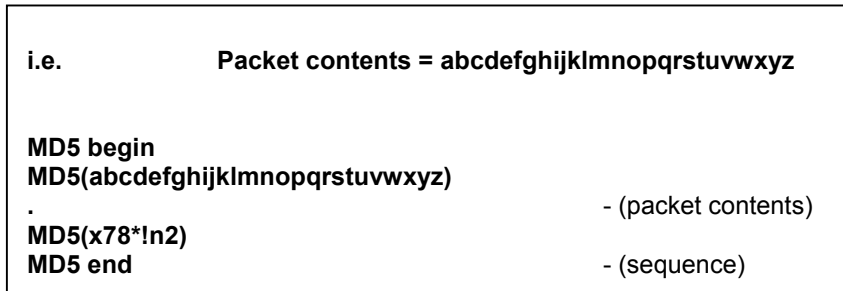


图7. Keyed-MD5处理



**图8. keyed-MD5 处理实例**

生成汇总的过程，按字节采用每个信息包的内容，并把它们装入一个缓存中。图8显示了一个带有“abcdefghijklmnopqrstuvwxyz”内容的信息包，从这个字符串中，每一个单独的字母都将被放到缓存器中。然后，MD5的规则将应用到整个缓存器中。如果缓存器不能容纳整个信息包，信息包的下一个小单位将被放到缓存器中，然后再次应用MD5的规则。一旦信息包的所有内容都被MD5的规则处理过，顺序将被追加上去，并且MD5的规则将应用于它。例如，在图8中，值“x78\*!n2”表示信息包在数据流中的顺序。因此，汇总是将MD5的规则应用于信息包内容的每一个字节以及它的顺序上的结果。

小单位信息包的处理也类似，利用顺序排列中的下一个顺序值。

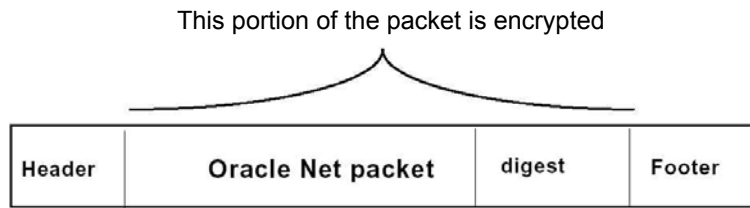


**图9. Oracle 网络信息包和它的汇总**

一旦信息包到达它的目的地，一个相应的程序将被应用于信息包。由于顺序是预先决定的，keyed-MD5功能可以计算汇总的值。如果计算结果与接收到的信息包的汇总相吻合，该信息将被接受，事务处理将继续。如果发生了攻击，信息包被插入，汇总的值与接收到的计算结果不吻合，在日志文件（SQLNET.LOG）中将会有一条输入的记录，而且连接被终止。

### **利用数据加密和整合性**

利用Oracle高级安全性，加密和校验和都能对数据流的通信提供最大的保护。当加密被起动，整个数据信息包以及汇总都被加密并在网络中得到传输。页眉页脚也作为一般网络路径处理的一部分被追加。在连接失败的情况下，一个跟踪工具或嗅探器可以帮助确定网络故障的方位。



**图10. 显示信息包与汇总加密的图形**

一旦信息包被接收，信息包将被解密，MD5的规则将被用于计算汇总值。

## 第三部分：安装与配置

尽管加密和整合性检查所利用的概念和数学逻辑是复杂的，但安装和配置Oracle高级安全性是简单的。一个管理员只需手工或利用网络管理器工具设定几个SQLNET.ORA参数，就能完成Oracle高级安全性的加密和整合性检查。

### 安装数据加密和整合性

为了安装Oracle高级安全性，可简单地安装Oracle数据库的企业版。Oracle高级安全性以及它的所有组成成分可以在数据库的“典型”安装模式中完成安装。但是，因为它是一个另外收费的选项，客户必须具有Oracle高级安全性的许可证才能配置和使用它的任何子项。

### 配置数据加密

建立Oracle网络加密有两种途径：(i) 利用网络管理器图形化用户界面工具，或(ii)手工编辑SQLNET.ORA文件。

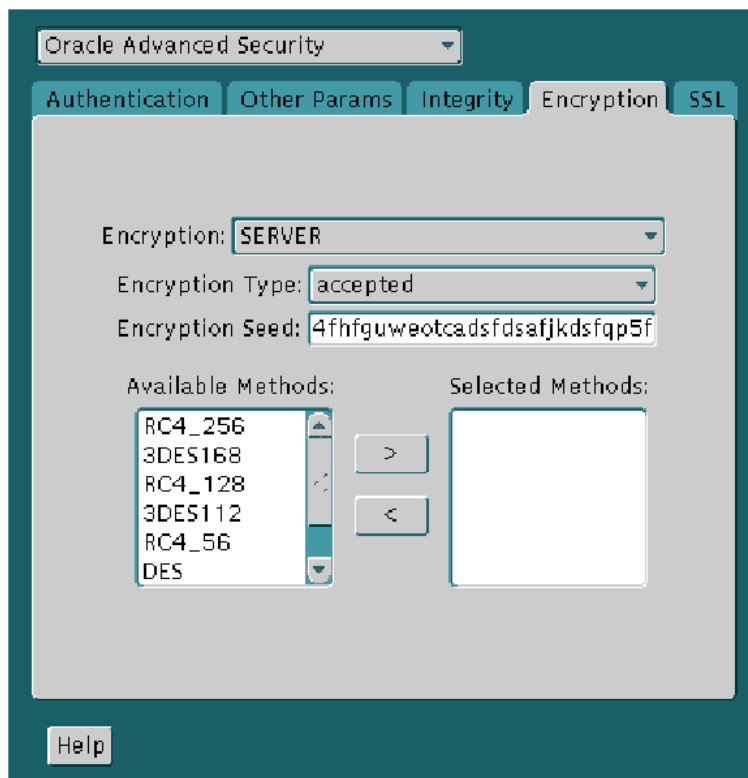


图11. 用于配置加密的网络管理器窗口

为了利用网络管理器配置加密，应在需要加密的每一台客户端和服务端运行该工具。从本地文件夹，选择Profile。然后，从Oracle高级安全性下拉菜单的“加密”标签中选择所需要的加密计算方法和密钥键长度，如图11所示。首先选择你将配置客户端还是服务器端，然后，设定加密值，如“要求”。最后，从左边面板中选择所有需要的计算方法和密钥键长度，并将它们移到右边的面板。

或者，你也可以在客户端和服务端手工编辑SQLNET.ORA文件。用于加密所需要的SQLNET.ORA的参数如下列：

#### 客户端：

```
sqlnet.encryption_client=[required | requested | accepted | rejected]
sqlnet.encryption_types_client=(RC4_256, 3DES168, RC4_128, 3DES112,
RC4_56, DES, RC4_40, DES40)
sqlnet.crypto_seed="j83hTCMDdo3boaoPAeh3basLGheu7asyi92kuvb23i9dhn3nNda
3
8dlkFdl32n9fhdfRhn"
```

#### 服务器端：

```
sqlnet.encryption_server=[required | requested | accepted | rejected]
sqlnet.encryption_types_server=(RC4_256, 3DES168, RC4_128, 3DES112,
RC4_56, DES, RC4_40, DES40)
sqlnet.crypto_seed="2z0hslkdharUJCFtkwbjOLbgwsj7vkqt3bGoUylihnhvkhgkdsbd
s
kkKGhdkl4p78hcpZr4"
```

不论你是手工编辑SQLNET.ORA还是利用网络管理器，你所选择的计算方法和密钥键的长度的顺序都是重要的。客户端和服务端在交流采用哪种计算方法时，将**从左向右**读取SQLNET.ORA文件的参数。以最优先的顺序输入计算方法。相似地，利用网络管理器时，以从上而下的优先顺序选择计算方法。我们推荐配置是首先用最长的密钥键设定计算方法，利用较长的密钥键不会对性能产生影响，但却可以显著增加安全性。想了解更多的信息，请参见*Oracle高级安全性管理员手册*的第一、二章和附录A。

加密的种子可最长达到70个字符。Oracle建议字符串包含70个字符，而且用户应该定期地改变加密的种子。这两条建议都会对生成加密的密钥键产生积极作用，如本文前面所述。用于决定是否对话路的加密起作用的参数有四个值：

- 要求 – 双方必须都能加密，否则连接失败。
  - 要求 – 如果另一方允许，可以加密。
  - 接受 – 如果另一方需要或要求可以加密。
  - 拒绝 – 即使被另一方要求，也不能加密。
- 缺省值是接受。

Oracle高级安全性所固有的Oracle网络加密可以在遵守美国联邦信息处理标准 (FIPS) 140-1 第二级安全性级别的模式下被配置。参照*Oracle高级安全性管理员手册*的附录D, 有关在该模式下的配置的介绍。

## 配置数据整合性

如同Oracle高级安全性管理员的其他子项一样, 由两种方法可以建立Oracle网络整合性: (i) 使用网络管理员图形化用户接口工具, 或(ii) 手工编辑SQLNET.ORA文件。

为了利用网络管理器配置数据整合性, 应在需要整合性检查的每一台客户端和服务端运行该工具。从本地文件夹, 选择Profile。然后, 从Oracle高级安全性下拉菜单的“整合性”标签中选择所需要的加密计算方法<sup>1</sup>。首先选择你将配置客户端还是服务器端, 然后, 设定整合性值, 如“要求”。最后, 从左边面板中选择所有需要的计算方法, 并将它们移到右边的面板。

或者, 你也可以在客户端和服务端手工编辑SQLNET.ORA文件。用于整合性所需要的SQLNET.ORA的参数如下列:

### 客户端:

```
sqlnet.crypto_checksum_client=[required | requested | accepted | rejected]
sqlnet.crypto_checksum_types_client=(MD5, SHA)
```

### 服务器端:

```
sqlnet.crypto_checksum_server=[required | requested | accepted | rejected]
sqlnet.crypto_checksum_types_server=(MD5, SHA)
```

不论你是手工编辑SQLNET.ORA还是利用网络管理器, 你所选择的计算方法的顺序都是重要的。客户端和服务端在交流采用哪种计算方法时, 将**从左向右**读取SQLNET.ORA文件的参数。以最优先的顺序输入SHA和MD5计算方法。因为只能用到一个。

与加密参数相似, 用于决定是否对话路的整合性起作用的参数有四个值:

- 要求 – 双方必须都要求整合性, 否则连接失败。
- 要求 – 如果另一方允许, 可以整合。
- 接受 – 如果另一方需要或要求可以整合。

---

<sup>1</sup>在当前版本中, 你必须手工编辑SQLNET.ORA文件才能利用SHA-1。

- 拒绝 – 即使被另一方要求，也不能整合。

缺省值是接受。

想了解更多的信息，请参见*Oracle高级安全性管理员手册*的第一、二章和附录A。

## 总结

Oracle高级安全性可以保护在数据库和事实上所有的客户端以及其他服务器之间的通讯，不管他们之间的协议和基础结构实怎样的。通过保证在网络中传输的所有数据的隐私性和整合性，它可以是用户在私人网络和互联网中传递敏感的和有价值的的数据，保证它免遭入侵者和黑客的攻击。当这些加密和整合性测定被结合到其他安全性的实践中时，如强大的认证和访问控制，用户将体会到端到端的安全性。本文所详细介绍的测定将对系统增加强化的安全性，能够对以数据库为中心的内联网和互联网的事务处理提供安全可靠的环境。