

# ***It's Not Paranoia If They're Really Out To Get You.***

## **A beginning DBA's overview of Database security.**

By William Thater, Principal Consultant Keane, Inc.

### **Introduction:**

With today's growing tendency to use Oracle databases as information providers to web applications, there is an increased exposure to malicious programmers and malware. Here I hope to provide the beginning DBA with an overview of Oracle features that affect the security of the data stored in his or her databases.

### **A bit about me:**

I began programming in 1968 [yes we really did have computers back then] and have been working with Oracle as a developer and/or DBA for the past 18 years. I'm currently a Principal Consultant for Keane, Inc. working with the Data warehouse support team at Carrier, a large air conditioning manufacturer.

### **Non-overview:**

Some of the things I won't be covering here:

- 1) Exact SQL statements/syntax: First, security policies vary so much from place to place that my answer would have to be "It Depends" and second, many other more talented people than I have already explained them in detail, for explanations see the books listed in the Thank You section.
- 2) Names and/or configurations of any network software: again this is highly dependent on an individual site's set up.

### **Overview:**

What I will be covering here:

- 1) Using what you've got: Oracle's built in behaviors and facilities

- 2) Using what you're given: interacting with network software and set up.
- 3) Application Design: things to consider when designing applications that connect the Internet to your database.

## **Using what you've got:**

### **Users and schemas:**

How many of you have done the following:

Created a user and given them default and temporary tablespaces, then granted them connect. You even log on with the new user no problems, and then turn it over. About 10 minutes later you get a call saying, "I can't create anything. Help!" Of course you immediately realize that you forgot to grant them any quota on the tablespace. And yes, my hand would be one of the first ones up. Now while this is not so good for the user that actually has to create tables and such, it may not be bad for a user that the application connected to the internet uses to read data from the database. Especially if you combine this with some of the other things we'll talk about in a little bit. That way if one of the bad guys manages to scoff the username and password from the application, when he logs on with it he can't do much. Will this stop him from stealing your data? No, but it makes it harder for him to trash your database.

How often have you had Tim call up and say "Dan has a table with data in it that I can use and he said to go ahead but I can't get to the table."? Right away you know that Dan hasn't granted any rights on his table to Tim. That's the way Oracle works. You call Dan up and explain it to him and maybe even grant the rights to Tim for him. Of course then you get the call saying "Hey, I still can't see the table, what did you do to my account!" You explain to Tim that he has to precede the table name with 'Dan.' and he'll be fine. "But I don't want to have to remember that." To solve that problem you create a synonym for him so he can just use the table name.

All of this is an example of Oracle security that we use all the time, but if we combine it with issuing the grants on tables to a role, then when the bad guy gets into the database, he's not going to see much of anything.

## Roles and Profiles:

Roles are Oracle's way of managing privileges for groups of users. In the example I gave above if other people wanted to use the same table of Dan's then you would create a role and assign the privileges on that table to the role. Granting the role to a user will give them the rights needed on the table, and when Dan decides to drop and recreate the table you only have one place you need to give the grants back to. When you grant the role to the user you have the option to make it a default role, that way when the user logs on the role is active. What's that got to do with security? Well making the role default is an optional step. If it's not default, then you have to activate it to get the privileges. This means that if you have a userid that your web application uses to access the database and the role to allow that userid to see the data is not default two things apply. First the application has to activate the role in some manner, and second if the bad guy logs on with that userid, he can't see much of anything. Remember that a role can't own synonyms and you will need to either set up public synonyms or prefix the object names with the owner. It's not always practical to have this situation, but it is a consideration when looking at your application planning.

Oracle also allows you to assign a profile to a userid either when you create the account or later on. Profiles allow you to manage a lot of the behavior of the user account: you can lock an account after a set number of log in failures, you can set resource limits so a user can only tie up the system for so long with the query from hell, and you can activate password management. These are only some of what can be controlled with profiles. Using the SYSTEM owned table PRODUCT\_USER\_PROFILE you can deny the userid for the web application the ability to sign on to the database with any program other than the application, so if the bad guy manages to get the user name and password he still can't log on to the database using SQL\*Plus or another program. It's easy to see how these actions can improve your security, but it does take planning and management. Locking an account can be a problem if the web application is being used by a large number of people and you end up locking them out. Planning the implementation of resource limits and account locking among other things needs to be coordinated with your network administrators, security people and developers.

## **Account and Password management:**

While we're talking about locking accounts, Oracle's account management is basically locking and unlocking the accounts. Why would you lock an account? Well, maybe because of intrusion detection by multiple login failures, or maybe if you separate the schema owner from the data managers by setting up procedures to insert, update and delete data. If that's the case you may wish to lock the owner account. If you have a Data warehouse environment where updates or refreshes are run at set intervals you might want to lock the data owner account in between them. Most of these decisions will depend on both your local environment and your local security policy.

Oracle's password management has become very flexible. It allows you to expire passwords at set intervals, and keep a history to prevent password reuse. You can also create your own customized procedure to ensure passwords comply with a set of rules. This will make your security people happy and your developers unhappy because they'll have to parameterize the log on. The trade off between those two groups needs to be assessed depending on the local security policies and environment.

## **Auditing:**

Oracle provides a system for auditing every type of transaction in the database. This is an excellent tool for gathering information. But like any intelligence gathering operation it can gather both too little and too much information and skew the decision process. The use of auditing should be planned out carefully to make sure that you are auditing information that will be of use to you, that you have space for the audit tables, and that you are reviewing the information on a timely basis. Also you need to be aware that the writing of the audit transactions will give you a performance hit. The size of the hit is dependent on the kinds of transactions being audited. You should also know that if you run out of space in the audit table tablespace, the database will hang.

## **Using what you're given:**

## **Firewalls:**

The primary interaction between the database and the network security is most likely taking place at the firewall. The firewall is the basic separator of the internal network from the Internet. Oracle has worked with many of the firewall vendors to allow firewalls to recognize and pass OracleNet traffic. This allows the network administrator to set up the network traffic between nodes with a simple rule, making them happier than having to open a large number of ports. SQL\*Net or Net8 or OracleNet, works by having a listener watching a TCP/IP port [1521 is the default, not using the default is recommended.] for a connection request. It will then authenticate the user and pass off the communication to another process using a random numbered higher port. Before Oracle made their protocols available to vendors that meant that the network administrator had to open a large number of these ports to allow the traffic to pass through causing a large exposure to attempted intrusion.

Now this can be set up with a rule to pass the traffic between nodes and can even pass it using secure tunneling. It also allows for encryption of the traffic.

## **Encryption:**

To encrypt or not to encrypt is a question that depends on both the local security policy and network architecture. Those factors also determine at which level the encryption takes place. The installation of Oracle's Advanced Security Option allows SQL\*Net to encrypt traffic by interfacing with a Kerberos or RSA server on the network. Some firewalls also will encrypt traffic across the network. The use of encryption, SSH tunneling, or other methods of securing network traffic is used for communications between the various zones of the network. Each zone defines a different level of security. The outer zones are directly connected to the Internet; this is where the web servers, PHP servers and such sit. The inner zones are where the intranet exists and usually where the database servers sit. In many cases a method of encryption is used when passing data through the firewall between nodes in different zones to secure the communication and prevent someone gaining any information by "sniffing" the network traffic.

## **Intrusion detection:**

Besides Oracle's built in intrusion detection, many firewalls include intrusion detection. Intrusion detection is also available in a separate software package that may be running on the network. These various network packages monitor log in attempts and will sound an alarm if it detects repeated failures of any one account. However if you have a web application going against a well-tuned database with a large number of users who are performing small, quick queries, this can generate a false positive and trigger an alarm for a denial of service attack.

## **Application design:**

Knowing how Oracle operates we can see how combining the options we discussed can lead to more secure application design. First the userid that the application connects to the database with should be kept separate from the owner of the data. If any updating or inserting of data is to be done by the application user it should be done through a stored procedure created under the data owner and having execute granted to the application user. A word of warning here, unless set up to run with the privileges of the calling userid the procedure runs with the authority and privileges of the owner of the procedure. Because of this I generally do not recommend allowing the application user to directly delete data. I try to set it up so that the application user marks data for deletion, but another user does the actual deletion after the data has been reviewed in some manner. The privileges on the data should be granted to the application userid through a role, and that role should not be default if possible. The profile for the application userid should restrict the resource use and the application userid should be denied access to the database with any program besides the application. All these combine to make the data more secure. But remember all security is compromises between letting the user get to the data in the manner they need and making sure the data remains intact.

**Thank You:**

Like any project this one was not conducted in a vacuum. As references I'd like to recommend two books, both from Oracle Press:

Oracle9i DBA 101 by Marlene Theriault, Rachel Carmichael and Jim Viscusi

Oracle Security Handbook by Marlene Theriault and Aaron Newman

I'd also like to thank Rachel for thinking I could pull this off and helping me to do just that. And Jen, for showing me what courage really is.

And most of all I'd like to thank you for making it all the way here. I hope I've managed to at least start you thinking about the questions you need to ask about your database security.

And may the force be with you!

Bill "Shrek" Thater, [bthater2@netscape.net](mailto:bthater2@netscape.net)