Official Publication of the Northern California Oracle Users Group

NoCOUG JOURNAL

Vol. 20, No. 2 · MAY 2006 $15

Add Color to Your Career with NoCOUG

Down Memory Lane

An interview with Bill Schwimmer. *See page 4.*

Is 24x7 a Myth?

Six well-known Oracles answer our questions. *See page 7.*

Hello, World!

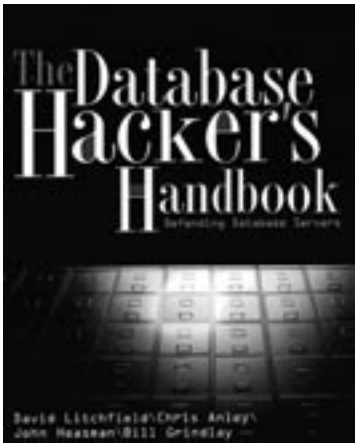
A new column on programming languages. Guest columnist Krishna Kakatur introduces us to SQLJ. *See page 19.*

Big discount on Steven Feuerstein seminar! ***See page 3.*** ***Much more inside . . .***

Database Hacker's Handbook

A Book Review by Brian Hitchcock

Summary



Overall review: Excellent; a book you must read.

Target audience: Anyone responsible for the security of corporate data.

Would you recommend this book to others? Without question.

Who will get the most from this book? Anyone with basic RDBMS experience.

Is this book platform specific? No.

Why did I obtain this book? I saw it on Amazon and wanted to read it. NoCOUG reimbursed me for the cost of the book.

Overall review

There are many books published about RDBMS technologies that you really don't need to know about. This isn't one of those books. You must read this book, or at least the first five chapters, which cover security issues in Oracle databases.

The authors do throw in some things that they don't explain but this is rare. I found myself wanting to know what a "0 day exploit" is, and I found out with a simple Internet search. (It means that someone takes advantage of a security vulnerability on the same day that the vulnerability becomes generally known.)

A reasonable person might question the morality of publishing this information. The reality is that the "bad guys" know all this and so should you. When you have an idea how the hacking happens, you have to deal with the magnitude of the problem and just how poorly protected your systems really are.

Chapter 1: Why Care About Database Security?

I think it is an excellent point that it doesn't really matter which vendor's database you use, it all really boils down to the specific implementation of your database in your environment. It is also excellent that they break down all of the database security flaws into eight groups.

Unauthenticated flaws in network protocols—These are issues in which a hacker can gain access to your database simply by being on the same network. The best protection for this is up-to-date patching.

Authenticated flaws in network protocols—Here the hacker has been able to connect to your database through normal means before the hacking begins. The recommended solution for this class of problems is to have a strong password policy.

Flaws in authentication protocols—The hacker uses a database user password that is detected as it is transmitted in plain text or by other means, and again the recommended solution is to be up-to-date with your patches.

Unauthenticated access to functionality—Features within the database can be executed by a database user who isn't supposed to be able to execute to them. The solution for this class of problems is to use a listener password.

Arbitrary code execution in intrinsic SQL elements—These issues involve causing buffer overflows simply through using SQL statements in unusual ways, and the recommended defense is to be up-to-date with patches.

Arbitrary code execution in securable SQL elements—These problems are caused by functions within Oracle that can be executed in specific ways to cause buffer overflows that allow a hacker to gain access. Patching is the best defense.

Privilege elevation via SQL injection—Here the issue is how to execute PL/SQL stored procedures in specific ways to get privileges that the database user shouldn't have. Patching is the best defense.

Local privilege elevation issues—The hacker executes functions within the database to go out and execute operating system utilities. The recommended solution for the user is to run your database as a low-privilege operating system user. I'm not clear how to do this since I always install Oracle as Unix user oracle.

There are many books published about RDBMS technologies that you really don't need to know about. This isn't one of those books.

Chapter 2: The Oracle Architecture

The vulnerabilities of Oracle Enterprise Manager are explained, among other things. This is a great example of

why I wanted to read this book and why I want you to read it as well. This chapter presents a new and very different view of Oracle. I would not have thought of Oracle Enterprise Manager as a security problem.

Simply from reading this chapter, I believe this book has the most information per page of any Oracle book I've ever seen.

The section on Oracle auditing is fascinating. The authors point out that if database auditing is on, any ALTER USER statement that changes the password is logged in plain text. Other things I learned in this chapter are that Oracle stores passwords in uppercase and this makes it easier for a brute force attack on the SYS.USER$ table.

Chapter 3: Attacking Oracle

The discussion of PL/SQL invoker rights, wrappers, and injection is very interesting, and very valuable. Simply from reading this chapter, I believe this book has the most information per page of any Oracle book I've ever seen. There are several sections in this chapter for which my comment is simply WOW! For example, the explanation of how a hacker can execute a procedure that can assume or inherit the privileges of the SYS user and how to grant DBA_ROLE to PUBLIC using autonomous transactions is just great. How easy it is to use DBMS_SQL also rates a WOW!

The authors explained in detail what conditions are necessary for an anonymous PL/SQL block to be used by a hacker; once that is understood, you can start looking through all of the PL/SQL blocks in Oracle to find others that meet the same conditions and can be exploited in the same way. This also illustrates how software could (and should?) be written to prevent many security issues in the first place.

Now I understand how hackers can find security issues. Once they know what conditions are necessary for them to exploit a piece of PL/SQL code, they can download and install Oracle on their own system and spend as much time as it takes looking for vulnerable pieces of code. This also illustrates why even the Oracle user SCOTT can be exploited. Once you know how to exploit pieces of code supplied by Oracle, even an Oracle database account as innocent as SCOTT can be used to break in. I used to wonder how an account like this could be a problem.

Chapter 4: Oracle: Moving Further into the Network

This chapter explains how the Oracle RDBMS itself can be seen as a "shell" because it can be used to execute operating system functions. In this manner a hacker can access the entire filesystem of the server machine, execute operating system commands, and access the network as well. A specific example explains how the Oracle listener was used to execute operating system functions as the operating system user oracle. One wonders why Oracle didn't cover all these issues at once. Surely these issues are known to those who wrote the software. This example doesn't give you a good feeling about the quality of commercially available software.

Chapter 5: Securing Oracle

While the previous chapters have been very educational and almost entertaining, this chapter attempts to tackle the really important and most difficult issue of all. Once you appreciate the depth, breadth, and complexity of the security issues, what exactly do you do about them?

Oracle is hard to secure because it is so big and so complex. A reasonable person would say that this argues for using simpler RDBMS software. Take MySQL for example. A simpler RDBMS to be sure, but what is the MySQL vendor doing? They're adding "features" as fast as they can, which adds to the size and complexity of their product. And why are they doing this? To meet market demands! Perhaps the enemy is us?

Perhaps the enemy is us?

The authors offer a list of recommendations on how to improve the security of our Oracle installation. While all of them make sense, and all of them have probably been heard before, I'm not convinced that most of them are really practical. Encrypting network traffic is a good idea, but our license agreement didn't include that specific (and expensive) Oracle functionality. The authors recommend that new account creation should be approved by a security officer. My group supports about 2000 databases; how many security officers should we hire? They also recommend that we use a strong password policy. Again I agree completely, but tell users of my mission critical systems that every 30 days one or more of the databases supporting their systems will lock out the database user that the application server uses to connect to the database. And, of course, I would need to maintain an up-to-date list of all the applications that might access all of the databases in my environment, so I can make sure they'll have up-to-date passwords. And none of the application code will have hard-coded passwords, correct? Auditing is certainly a good idea, but who will review the audit trails for all of the databases in my environment? We are told that we should check all PL/SQL modules to make sure that they include AUTHID CURRENT_USER, for example. Can you imagine reviewing all of the modules in Oracle Applications? We will also need someone to create and maintain a secure PL/SQL coding standard and do code reviews.

All of this ignores the threat from within.

The hard part in all of these recommendations is balancing the business need for security against the cost of the security while taking into account the benefits of the complex database features that you want for your business. I would have found this chapter much more valuable if it told me how all these suggestions were put in place for a large IT organization, and how they evaluated the costs versus the benefits. It's much easier to make these recommendations than it is to implement them.

The hard part in all of these recommendations is balancing the business need for security against the cost of the security.

Chapters 6–26: DB2, Informix, Sybase, MySQL, SQL Server, PostgreSQL

These chapters cover the same issues for each of these vendors' RDBMS products. I did read them all and I think they are worth your time, if only to see just how differently each of these products is architected. Since this is primarily a review for users of Oracle, I didn't review these chapters.

Conclusion

The real issue with security is balancing the cost of implementing improved security against the cost of a successful break-in. We can all agree that more security is better, but at what cost? And how do you determine the cost to your business of a security breach?

The complexity of Oracle invites security issues, but your business benefits from the advanced features available. Simpler software may be more secure, but then we want more features. More features (complexity) breed more security flaws, which require more patching, and the patching becomes more complex—what's wrong with this picture?

And, no matter how technically proficient we think we are, all of this ignores the threat from within. If those who are approved for full access decide to embark on their own nefarious schemes, none of what is discussed here will matter. The internal threats are much harder to deal with and this book doesn't offer any advice on these issues.

About the Authors

All are members of Next Generation Security Software Ltd.

David Litchfield holds the unofficial world record for finding major security flaws, including a major breach in Oracle 9 database servers.

Chris Anley has published white papers and security advisories on SQL Server, Oracle, and other database systems.

John Heasman has authored security advisories on high-profile products including Microsoft Windows, RealPlayer, and Apple QuickTime.

Bill Grindlay has worked on a generalized vulnerability scanner as well as NGSSquirrel database security scanners. ▲

Brian Hitchcock has worked at Sun Microsystems in Newark, California, for the past 11 years. He is a member of a DBA team that supports 2400+ databases for many different applications at Sun. He frequently handles issues involving tuning, character sets, and Oracle applications. Other interests include Formula One racing, finishing his second Tiffany Wisteria lamp, Springbok puzzles, Märklin model trains, Corel Painter 8, and watching TV (TiVo rules!). Previous book reviews by Brian and his contact information are available at **www.brianhitchcock.net**.