Oracle Database

INTEGRIGY

November 12, 2006

Security Analysis

Spoofing Oracle Session Information

OVERVIEW

INTRODUCTION

Oracle Database session information includes database user name, operating system user name, host, terminal, IP address, module, program, timestamps, session ID, and other details. These values are critical to auditing and identifying the actual end-user. Many of the database session values can be "spoofed" by an attacker either to mask the true identity or to circumvent security and auditing measures.

This paper will look at four common stores and uses of database session information related to security and auditing: (1) V$SESSION view, (2) SYS_CONTEXT function, (3) Database Session Auditing, and (4) Fine Grained Auditing (FGA). The V$SESSION view contains one row per current database session. The SYS_CONTEXT function returns information regarding the current database session and is often used with database logon triggers. Database session auditing (AUDIT SESSION;) records all database logons and logoffs. Fine Grained Auditing is used to audit SQL statements executed for specific database objects and can be configured based on columns or other criteria.

This paper is the first comprehensive examination of the key database session elements and the ability to spoof these values – there is neither new information nor new security vulnerabilities discussed in this paper. Much of the information has been previously discussed to some extent in Oracle-related security mailing lists and other forums [1].

Even though experienced database administrators know about the ability to spoof certain session values, we consistently observe programs and applications relying on "spoofable" session information within database logon triggers and in database auditing. With the emphasis on auditing in recent years due to legislative requirements, we are actually seeing this issue much more often now than three years ago. This is an auditing implementation issue found especially in large complex ERP or CRM implementations where auditing has been configured to satisfy Sarbanes-Oxley (SOX), HIPAA, Payment Card Industry Data Security Standard (PCI DSS), or other legislative and regulatory requirements. Due to the complexity of an application and large

transaction volume, often some level of filtering must be enabled to sift through the enormous amount of audit data generated. In many cases, the filtering is applied to differentiate between normal transaction processing and anomalous transactions by either an attacker or a privileged user (i.e., Super User or DBA) when auditing has been enabled due to SOX requirements.

The goal of this paper is to build awareness of this issue since Integrigy consultants performing security assessments consistently see absolute reliance on "spoofable" values in auditing and security solutions. Even a quick search of Oracle Metalink shows some support notes with caveats about the ability to spoof these session values (e.g., 281229.1), while other notes provide solutions or samples that rely on these modifiable values (e.g., 336254.1, 136145.1).

SUMMARY

With the exception of the database username, session ID, and timestamps, most session-level values can be easily manipulated by the user and are not reliable for security protections or audit trails. Using sample code provided by Oracle, any skilled Oracle database administrator or developer should be able to create a simple Java program that sets all session values to arbitrary values with the exception of the database username, session ID, IP address, and timestamps. The lack of reliability of the database session information must carefully considered when designing security protections (e.g., database logon triggers) and auditing.

DATABASE SESSION INFORMATION

Database session-level information is stored and referenced in a number of places within the Oracle Database. The V$SESSION view contains all current sessions and the SYS_CONTEXT function displays information about the current session. Both are often used in database logon triggers and other real-time auditing. Database session auditing (i.e., AUDIT SESSION;) captures an audit of all database sessions and Fine Grained Auditing (FGA) can be used to audit SELECT statements (10.2 supports INSERT, UPDATE, and DELETE as well). There are other stores and uses of session-level information, however, these four are the most frequently utilized.

V$SESSION

V$SESSION is a dynamic performance view that contains information for each database session. This view contains all the key session values. V$SESSION is really a public synonym to the view SYS.V_$SESSION.

By default in Oracle 9.2, 10.1, and 10.2, SELECT privilege on SYS.V_$SESSION has only been granted to SELECT_CATALOG_ROLE. Some applications and database components may grant SELECT privilege on SYS.V_$SESSION to other users or roles as part of the installation process.

SYS_CONTEXT

SYS_CONTEXT is a standard Oracle Database function used to retrieve session-level information. The function is defined in SYS.STANDARD and documented in the Oracle Database SQL Reference manual. Refer to the documentation for how SYS_CONTEXT works and the use of namespaces with the function.

SYS_CONTEXT can be used for multiple purposes, but most often is used to reference the "USERENV" namespace objects related to session information. Rather than querying V$SESSION, database triggers - especially database logon triggers - mostly use SYS_CONTEXT to retrieve session information. This paper will focus exclusively on the "USERENV" namespace values and all references to SYS_CONTEXT assume the "USERENV" namespace.

SYS_CONTEXT can be used in SQL statements or PL/SQL. An example usage is as follows –

```
select SYS_CONTEXT('USERENV', 'OS_USER') from dual;
```

DATABASE SESSION AUDITING

This paper will focus on session auditing (i.e., AUDIT SESSION;), although all the information is also relevant to other database auditing since all the audit data is stored in a single table (SYS.AUD$).

In order to make this paper as user friendly as possible, we will reference the commonly used audit views rather that the underlying tables. For instance, DBA_AUDIT_SESSION will be referenced instead of the base table SYS.AUD$. Session information is stored in all the audit session views based on SYS.AUD$ - DBA_AUDIT_TRAIL, DBA_AUDIT_SESSION, DBA_AUDIT_EXISTS, DBA_AUDIT_OBJECT, and DBA_AUDIT_STATEMENT.

FINE GRAINED AUDITING (FGA)

Fine Grained Auditing is used to audit SQL statements executed for specific database objects and can be configured based on columns or other criteria. Audit data is stored in the SYS.FGA_LOG$ and usually accessed via the DBA_FGA_AUDIT_TRAIL view.

DATABASE SESSION INFORMATION

The database session values are not consistently named throughout the database. Table 1 provides a mapping of the session values to the column names for the key views and functions referenced in this paper. Table 2 provides a brief description of each session values and typical values that will be found in most databases when using common tools and drivers like SQL*Plus, TOAD, JDBC, etc.

Table 1 – Session Value to Column Mapping				
Session Value	**V$SESSION View**	**SYS_CONTEXT Function**	**DBA_AUDIT_* **	**DBA_FGA AUDIT_TRAIL**
User Name	USERNAME	SESSION_USER	USERNAME	DB_USER
Schema Name	SCHEMANAME	CURRENT_SCHEMA		
OS User	OSUSER	OS_USER	OS_USERNAME	OS_USER
Machine	MACHINE	HOST	USERHOST	USERHOST
Terminal	TERMINAL	TERMINAL	TERMINAL	
Program	PROGRAM			
IP Address		IP_ADDRESS	COMMENT_TEXT[1]	
Process ID	PROCESS			
Module	MODULE	MODULE		
Action	ACTION	ACTION		
Client Info	CLIENT_INFO	CLIENT_INFO		
Client ID	CLIENT_IDENTIFIER	CLIENT_IDENTIFIER	CLIENT_ID	CLIENT_ID

[1] The COMMENT_TEXT column of the DBA_AUDIT_TRAIL view contains the IP address embedded in an authentication string.

Table 2 – Typical Session Values[1]	
Value	**Description**
User Name	Database username used to authenticate to the database.
Schema Name	Current database schema name as set by "ALTER SESSION SET CURRENT_SCHEMA". Upon initial login, the current database schema name will be the user name.
OS User	Client operating system user name which will be dependent on the client operating system (Windows, Linux, Unix, Macintosh, etc.), Oracle drivers, and database authentication method.
Machine	• For Unix and Linux, the fully qualified host name. • For Windows, (1) the domain name or workgroup and Windows computer name (e.g., DOMAIN\COMPUTER) or (2) the Windows computer name.
Terminal	• For Unix and Linux, (1) "unknown", (2) NULL, or (3) fully qualified host name. • For Windows, (1) the domain name or workgroup and Windows computer name (2) the Windows computer name, or (3) NULL.
Program	• For Unix and Linux, (1) user and host name for OCI programs (e.g., "oracle@host.domain.com (J001)") or (2) often NULL or an application defined value for Java programs using JDBC. • For Windows, (1) the Windows executable name for programs using OCI or (2) often NULL or an application defined value for Java programs using JDBC.
IP Address	The IP address of the client connection to the TNS listener.
Process ID	Client operating system process ID in the following format – • For Unix and Linux, the operating system process ID (e.g., 12345). • For Windows, in the format x:y where x is the Windows process ID.
Module	Set during the database session using the database procedure DBMS_APPLICATION_INFO.SET_MODULE or using the OCI attribute OCI_ATTR_MODULE.
Action	Set during the database session using the database procedure DBMS_APPLICATION_INFO.SET_MODULE or DBMS_APPLICATION_INFO.SET_ACTION or using the OCI attribute OCI_ATTR_ACTION.
Client Info	Set during the database session using the database procedure DBMS_APPLICATION_INFO.SET_CLIENT_INFO or using the OCI attribute OCI_ATTR_CLIENT_INFO.
Client ID	Set during the database session using the database procedure DBMS_SESSION.SET_IDENTIFIER or using the OCI attribute OCI_ATTR_CLIENT_IDENTIFIER (OCI) or OracleConnection.setClientIdentifier (JDBC). "Client ID" is most often used to identify application level users when application authentication is handled by the application and a common database username is used.

[1] The example values are typically found in many databases and are only representative of commonly used applications like SQL*Plus, Toad, JDBC drivers, etc. As will be described later, most of the values can actually be any value set by the client program.

SPOOFING SESSION INFORMATION

SUMMARY

From an auditing perspective, the only information that is completely reliable is the database user name – all other information regarding the client can be spoofed. Key session information regarding the client and end-user can be readily spoofed. Other session and audit data like session ID and timestamps can not be spoofed using the methods described in this paper.

The session values shown in red in the following table can be spoofed. The IP address can be spoofed, but requires more advanced techniques.

Table 3 – Spoofable Session Values

Session Value	V$SESSION View	SYS_CONTEXT Function	DBA_AUDIT_*	DBA_FGA AUDIT_TRAIL
User Name	USERNAME	SESSION_USER	USERNAME	DB_USER
Schema Name	**SCHEMANAME**	**CURRENT_SCHEMA**		
OS User	**OSUSER**	**OS_USER**	**OS_USERNAME**	**OS_USER**
Machine	**MACHINE**	**HOST**	**USERHOST**	**USERHOST**
Terminal	**TERMINAL**	**TERMINAL**	**TERMINAL**	
Program	**PROGRAM**			
IP Address		***IP_ADDRESS***	***COMMENT_TEXT***[1]	
Process ID	**PROCESS**			
Module	**MODULE**	**MODULE**		
Action	**ACTION**	**ACTION**		
Client Info	**CLIENT_INFO**	**CLIENT_INFO**		
Client ID	**CLIENT_IDENTIFIER**	**CLIENT_IDENTIFIER**	**CLIENT_ID**	**CLIENT_ID**

[1] The COMMENT_TEXT column of the DBA_AUDIT_TRAIL view contains the IP address embedded in an authentication string.

ASSUMPTIONS

The focus of this paper is much more on auditing of properly authenticated end-users than on an attacker hacking the database without any database credentials. In some of our testing, we even assumed the end-user has access to privileged database accounts that had system privileges like SELECT ANY TABLE. With legislative compliance requirements from Sarbanes-Oxley and HIPAA, auditing of privileged database accounts is required and is an important aspect of an organization's IT controls.

The information and conclusions in this paper assume an Oracle Database running 9.2 or 10.2 hosted on either Linux or a common Unix variant (Sun, HP, IBM). The database is assumed to be using database authentication rather than single sign-on, external authentication using Advanced Security Option (ASO), operating system, network, or proxy authentication. Other authentication modes and operating systems may not permit certain session values to be spoofed, particularly

operating system user name. Although, using JDBC drivers on the client, the client operating system has no impact on the ability to spoof the session values.

This paper only looks at methods used to spoof the session values that would be in the knowledge realm of an experienced Oracle database administrator or developer. Specifically, techniques to manipulate the Oracle SGA [2], to spoof IP addresses, and to circumvent normal security measures by exploiting security bugs like SQL injection were not investigated. Other advanced techniques for unique scenarios, such as having BECOME USER system privilege and using the OCI function upicui were not investigated.

SESSION VALUES

DATABASE USER NAME - NO

The database user name is the most critical session value and can not be readily spoofed by the client.

A privileged account with certain permissions (like BECOME USER system privilege) may be able to spoof the database user name in the audit trail. Also, by exploiting known security bugs, an attacker with permissions to vulnerable database packages can execute SQL statements as SYS or other database accounts.

CURRENT SCHEMA NAME - YES

The current schema name is set using the ALTER SESSION SET CURRENT_SCHEMA statement. Upon initial logon, the current schema name and database user name should be equal. At anytime after logon, the user is able to change the current schema name to any valid database account. Setting the current schema name provides no additional privileges or other benefits, rather allows the user to access database objects without fully qualifying the name.

OPERATING SYSTEM USER - YES

The operating system user is supplied directly by the client and can be set to any value by an attacker. The OS user value is set during the session initialization. Using the Oracle JDBC driver, the value can be set prior to the connection initialization. There may be limitations if non-database authentication is being used.

MACHINE - YES

The machine name is supplied directly by the client and can be set to any value by an attacker. The machine name value is set during the session initialization. Using the Oracle JDBC driver, the value can be set prior to the connection initialization. There may be limitations if non-database authentication is being used.

TERMINAL - YES

The terminal name is supplied directly by the client and can be set to any value by an attacker. The terminal value is set during the session initialization. Using the Oracle JDBC driver, the value can be set prior to the connection initialization. There may be limitations if non-database authentication is being used.

PROGRAM – YES

The program name is supplied directly by the client and can be set to any value by an attacker. The program value is set during the session initialization. Using the Oracle JDBC driver, the value can be set prior to the connection initialization. With the Oracle OCI driver, simply renaming the "exe" file will allow you to manipulate the program name.

IP ADDRESS - MAYBE

The IP address is supplied to the database from the TNS listener. Using standard Oracle clients (i.e., OCI or JDBC), there is no simple method to manipulate the IP address. The easiest way to change the IP address in many networks is to simply connect in another location like a conference room or to change the MAC address on the computer to obtain a new IP address through DHCP.

More sophisticated methods of IP address spoofing or using a proxy may be effective, although these methods are out of the realm of the typical Oracle DBA or developer. Spoofing of IP addresses in an Oracle environment will be the topic of a future whitepaper.

PROCESS ID - YES

The client operating system process ID is supplied directly by the client and can be set to any value by an attacker. The process ID value is set during the session initialization. Using the Oracle JDBC driver, the value can be set prior to the connection initialization.

MODULE - YES

The module name can be set by the user through either the PL/SQL package DBMS_APPLICATION_INFO.SET_MODULE or the OCI attribute OCI_ATTR_MODULE. The module name can be set to any value by an attacker after the session is initiated. This is standard Oracle functionality and the application should be setting the module name as part of normal processing.

ACTION – YES

The action value can be set by the user through the PL/SQL packages DBMS_APPLICATION_INFO.SET_MODULE and DBMS_APPLICATION_INFO.SET_ACTION or the OCI attribute OCI_ATTR_ACTION. The action value can be set to any value by an attacker after the

session is initiated. This is standard Oracle functionality and the application should be setting the action value as part of normal processing.

CLIENT INFO – YES

The client info value can be set by the user through the PL/SQL package DBMS_APPLICATION_INFO.SET_CLIENT_INFO or the OCI attribute OCI_ATTR_CLIENT_INFO. The client info value can be set to any value by an attacker after the session is initiated. This is standard Oracle functionality and the application should be setting the client info value as part of normal processing.

CLIENT ID – YES

The client ID value can be set by the user through the PL/SQL package DBMS_SESSION.SET_IDENTIFIER or the OCI attribute OCI_ATTR_CLIENT_IDENTIFIER. The client info value can be set to any value by an attacker after the session is initiated. This is standard Oracle functionality and the application should be setting the client ID value as part of normal processing.

SAMPLE CODE

There are various methods that can be used to programmatically set the session values. The simplest method is to use the Oracle JDBC drivers and the properties referenced in Table 4.

Table 4 – Setting Session Values

Session Value	Database Session	OCI (OCIAttrSet)	JDBC
Schema Name	ALTER SESSION SET CURRENT_SCHEMA		
OS User			OracleConnection property v$session.osuser
Machine			OracleConnection property v$session.machine
Terminal			OracleConnection property v$session.terminal
Program			OracleConnection property v$session.program
Process ID			OracleConnection property v$session.process
Module	DBMS_APPLICATION_INFO.SET_MODULE	OCI_ATTR_MODULE	OracleConnection.END_TO_END_MODULE_INDEX or OracleConnection property v$session.module
Action	DBMS_APPLICATION_INFO.SET_MODULE or DBMS_APPLICATION_INFO.SET_ACTION	OCI_ATTR_ACTION	OracleConnection.END_TO_END_ACTION_INDEX
Client Info	DBMS_APPLICATION_INFO.SET_CLIENT_INFO	OCI_ATTR_CLIENT_INFO	
Client ID	DBMS_SESSION.SET_IDENTIFIER	OCI_ATTR_CLIENT_IDENTIFIER	Oracle.jdbc.OracleConnection.setClientIdentifier or OracleConnection.END_TO_END_CLIENTID_INDEX

Oracle Metalink Notes 369236.1 and 147413.1 provide fairly complete sample code for setting most of the session values using the Oracle JDBC driver [3,4]. The Oracle JDBC drivers across database versions (i.e., 9.2 vs 10.2) are inconsistent in their knack for correctly setting the values.

We performed most of our testing using the Oracle JDBC 9.2.0.5 Thin drivers rather than the Oracle JDBC 10g drivers.

REFERENCES

1. Pete Finnigan, "Can application names be changed to spoof logon triggers?", Pete Finnigan Oracle Security Blog, www.petefinnigan.com
2. Miladin Modrakovic, "Reading and Storing Data Directly From Oracle SGA using Pro*C/C code"
3. Oracle Corporation, "Method setClientIdentifier() Not Found In Interface oracle.jdbc.OracleConnection for JDBC 10g", Oracle Metalink Note ID 369236.1, 11 May 2006
4. Oracle Corporation, "Example: How to set V$SESSION Properties Using the JDBC Thin Driver", Oracle Metalink Note ID 147413.1, 14 November 2002

HISTORY

November 12, 2006 – Initial Version

INTEGRIGY

About Integrigy

Integrigy Corporation is a leader in application security for large enterprise, mission critical applications. Our application vulnerability assessment tool, AppSentry, assists companies in securing their largest and most important applications. AppDefend is an intrusion prevention system for Oracle Applications and blocks common types of attacks against application servers. Integrigy Consulting offers security assessment services for leading ERP and CRM applications.

Integrigy Corporation
P.O. Box 81545
Chicago, Illinois 60602 USA
888/542-4802
www.integrigy.com

Authors: Stephen Kost and Jack Kanter

If you have any questions, comments or suggestions regarding this document, please send them via e-mail to alerts@integrigy.com.