

# **Search Engines Used to Attack Databases**

**White Paper**

**Aaron C. Newman, CTO & Founder, Application Security, Inc.**

**APPLICATION  
SECURITY, INC.**

[www.appsecinc.com](http://www.appsecinc.com)

Tel: 1-866-9APPSEC

E-mail: [info@appsecinc.com](mailto:info@appsecinc.com)

Introduction..... 3

Searching for an Oracle database ..... 4

    iSQLPlus ..... 4

    SQL Injection in Demo Applications..... 11

    Looking for Directory Indexing ..... 16

Mitigating the Threat..... 18

Conclusion ..... 19

Bibliography..... 20

About Application Security, Inc. .... 20

## INTRODUCTION

Database security has recently become the victim of misused search engines. Over the last year or so, Hackers have begun to use search engines to find potentially vulnerable web applications to attack. The search engine doesn't actually execute any attacks, rather it is used to quickly locate "soft targets" among the vast number of sites on the internet. The hacker then targets the vulnerable sites with attacks designed to exploit the specific holes discovered by the search engine.

More recently, hackers have started to use search engines to find web facing database interfaces that can be used to mount attacks on databases placed behind a firewall. This is a significant new development, completely exposing previously "protected" databases to outside attack. As we will demonstrate in this white paper, an attacker can data mine any of the commonly used search engines to find target databases to attack.

So, what makes search engines such a powerful tool for an attacker? It is their ability to provide a thorough inventory of all the pages within a web application. Before launching an attack, the attacker needs to gather two key pieces of information. Where to mount the attack, and what vulnerabilities to target. A search engine can provide all of this information. Most popular search engines are advanced enough to provide robust capabilities to search for very targeted items. You can search within a single domain, you can search for URLs that match a given criteria, and you can search for certain words or phrases in the content of a web page, the title of a page, even the URL of the page.

The first use of search engines as attack tools grabbed the public's attention in early 2004, when an article in Wired showed how to use Google to search for FileMaker Pro database interfaces. Here is a link to the article:

- <http://www.wired.com/news/infrastructure/0,1377,57897,00.html>

At the same time, many presentations were given at security and hacking conferences on using search engine attack techniques to search sensitive data exposed on the internet. All of a sudden, hackers were using search engines to find exposed Excel spreadsheets with the words like "finance" and "confidential" in them. This proved to be a powerful and incredibly easy method for discovering sensitive data that had been crawled by various search engines after companies had inadvertently exposed sensitive files to the public Internet. We are now starting to see freely available tools designed to specifically to use Google to look for vulnerabilities in web sites. See the link below:

- <http://johnny.ihackstuff.com/index.php?module=prodreviews>

This new form of attack is troublesome for many reasons - the biggest being that these search engine repositories reduce the need for the attacker to probe the victims of an attack. This makes detecting the actual breach much more difficult; since the characteristic early warning signs no longer appear, and the forensic evidence is significantly reduced. In the past, attacks usually involved preliminary reconnaissance activity, which can be picked up on and monitored. If the attacker can plan their attack without ever visiting the victim site, the element of surprise they gain during the actual breach gives them a significant advantage over the target.

The fact that we have seen a new focus on attacking databases through these search engines should be a wake up call to both the database industry and to the security industry. The traditional approach to securing databases has been to create a hardened perimeter – this strategy is no longer viable. 'The perimeter is dead.' Organizations that rely solely on perimeter security are accepting a significant level of risk that their mission critical data will be compromised.

“Isn’t perimeter security enough? We bought all those firewalls – aren’t they keeping us safe from attackers?” The answer is that perimeter security is necessary but is unequivocally not sufficient. The key to successful, economical security is proper resource allocation. Today we see organizations placing about 95% of their security effort securing the perimeter, and only about 5% on securing their data where it lays. Why is this a problem? Think about it – what are we protecting? Are we protecting firewalls? No! Are we protecting routers? No! Are we protecting hardware? No! We are protecting data. It’s a simple concept – the data in your organization is the most valuable asset it has (except for perhaps people). We need to look at protecting data right where it lays; in the database.

## SEARCHING FOR AN ORACLE DATABASE

Firewalls are a tremendously important component of any organization’s perimeter security effort. Everyone’s got a firewall, and it’s almost unheard of for a database to be exposed to the internet without a firewall in place to restrict access. Protecting a database with a firewall is a good thing, but sometimes DBAs put too much stock in the security provided by the firewall. This results in many DBAs leaving low hanging fruit exposed - meaning that often even simple security holes go unfixed. This situation is greatly exacerbated by Oracle Corporation’s stance on what is considered “Risk to exposure”. The following excerpt is taken from Oracle’s website:

- <http://www.oracle.com/technology/deploy/security/pdf/2003Alert58.pdf>

### **“Risk to exposure**

Unless you connect the database directly to the Internet (e.g., no intervening application server or firewall), a remote buffer overflow attack via the Internet is, in Oracle’s opinion, unlikely.”

Hmmm – so Oracle’s stance is that if you have a firewall, remote buffer overflows are unlikely to occur. I may be twisting what they are saying a little, but this comment is naïve and demonstrates a genuine lack of knowledge of good security practice. Oracle is in effect, telling DBAs not to worry about quickly applying patches or fixing security issues because they are protected by a firewall. Security requires paranoia. Attackers are going to find a way past that firewall; they always do. If you make the firewall your last line of defense, even unsophisticated attackers will have an easy time having their way with your databases once they get past the perimeter.

Of course, the significance of the search engine is that the databases thought to be protected behind the firewall, but in fact exposed to the world are now easy to find. Let’s take a look at how this works.

## ***iSQLPLUS***

Oracle has long been packaged and delivered with a tool called SQLPlus. This is a standard query tool used to run SQL statements against the database and return results and error messages. The tool was updated to iSQLPlus with Oracle8i to reflect the company’s new focus on Internet technology. Other than a facelift, little actually changed in the product. iSQLPlus continues to be an important tool for DBAs. Initially iSQLPlus was a client application installed on a workstation. It was designed as a Java application that ran on a fat client and connected over the network to the database. iSQLPlus required that the Oracle SQL\*Net drivers be installed on the client as well as the iSQLPlus program itself.

Oracle9i introduced a new architecture for iSQLPlus – delivering it as a web application. This was a great idea since it removed the need for a client application, and the need for SQL\*Net drivers. Effectively, any platform with an HTML browser could now connect to iSQLPlus and run database queries. This was a welcome change in how iSQLPlus was used. Anyone could administer and run queries against Oracle through any platform that supported a browser - even a PDA.

iSQLPlus was implemented as a Java servlet running in the standard Oracle HTTP Server. The iSQLPlus web application was included by default with the Oracle9i database, listening on port 7777 under the URL /isqlplus. The tool is also included in most versions of Oracle Application Server using this same URL. Oracle10g has made some minor adjustments to the tool's default configuration, dropping the HTTP server listening on port 7777, and instead created an Oracle HTTP Server running on port 5560. The iSQLPlus application now runs on this new port.

If one of these ports has been exposed to the Internet, several results will occur:

1. The web server along with the default applications included isqlplus will be inventoried by search engines
2. A route to directly access an internal Oracle database from the public internet is created

Knowing about this utility, an attacker can start looking for databases by getting a list of websites running iSQLPlus. This is easily accomplished using Google's (<http://www.google.com>) "Advanced Search" option.

Google Advanced Search - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Media Refresh Print Mail Wordpad Help

Address [http://www.google.com/advanced\\_search?hl=en](http://www.google.com/advanced_search?hl=en)

Google allinurl: "/isqlplus" Search Web Search Site Options isqlplus

**Google** **Advanced Search** [Advanced Search Tips](#) | [About Google](#)

**Find results** with **all** of the words  10 results

with the **exact phrase**

with **at least one** of the words

**without** the words

**Language** Return pages written in

**File Format**  return results of the file format

**Date** Return web pages updated in the

**Numeric Range** Return web pages containing numbers between  and

**Occurrences** Return results where my terms occur

**Domain**  return results from the site or domain

**SafeSearch**  No filtering  Filter using [SafeSearch](#)

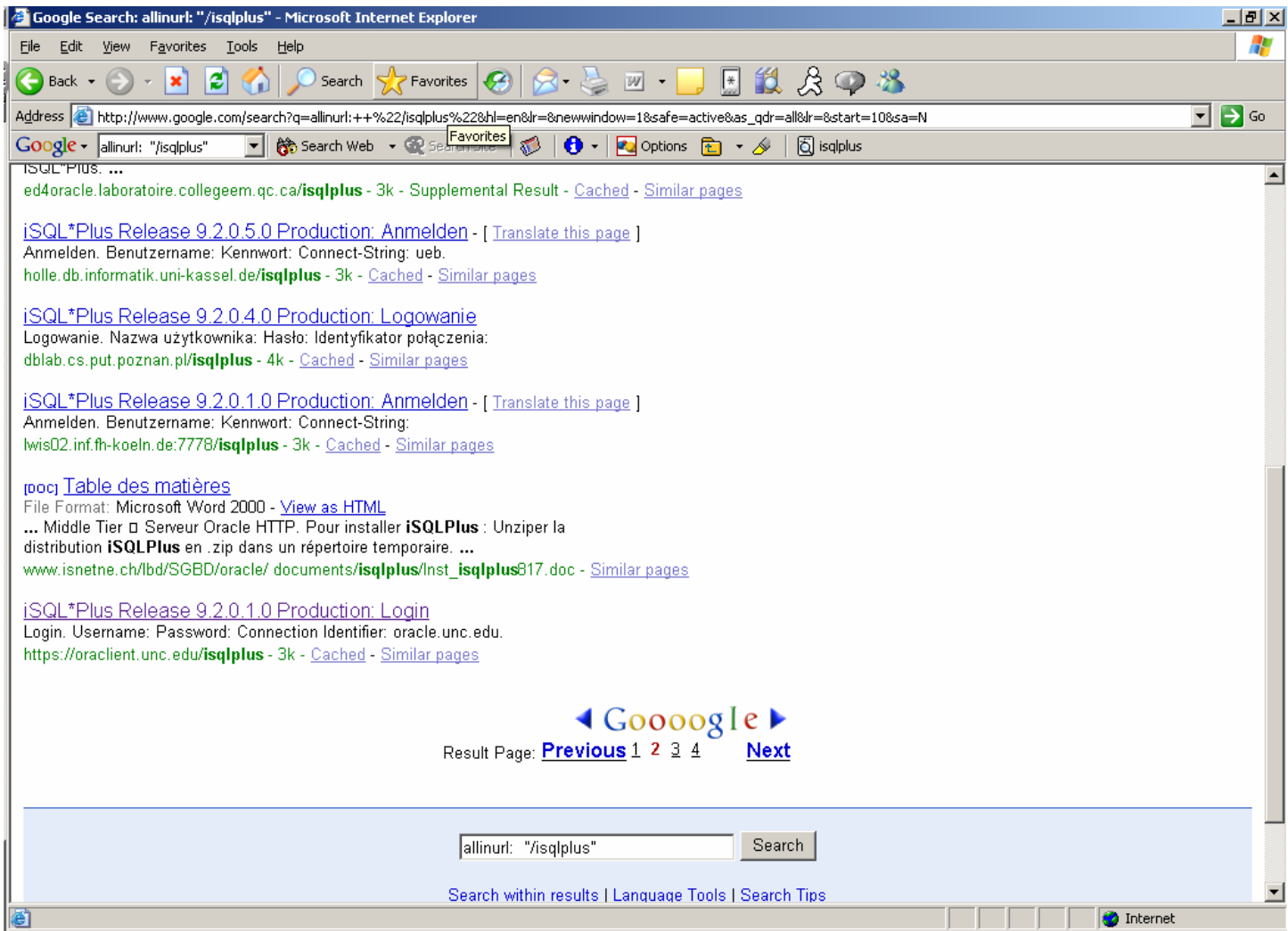
**Froogle Product Search (BETA)**

**Products** Find products for sale

To browse for products, start at the [Froogle home page](#)

**Page-Specific Search**

As you can see in the figure above, the attacker sets the “with the exact phrase” field to the value “/isqlplus”. The attacker also sets the “Occurrences” field to “in the URL of the page”. Google is now set to retrieve a list of websites with a URL including /isqlplus.



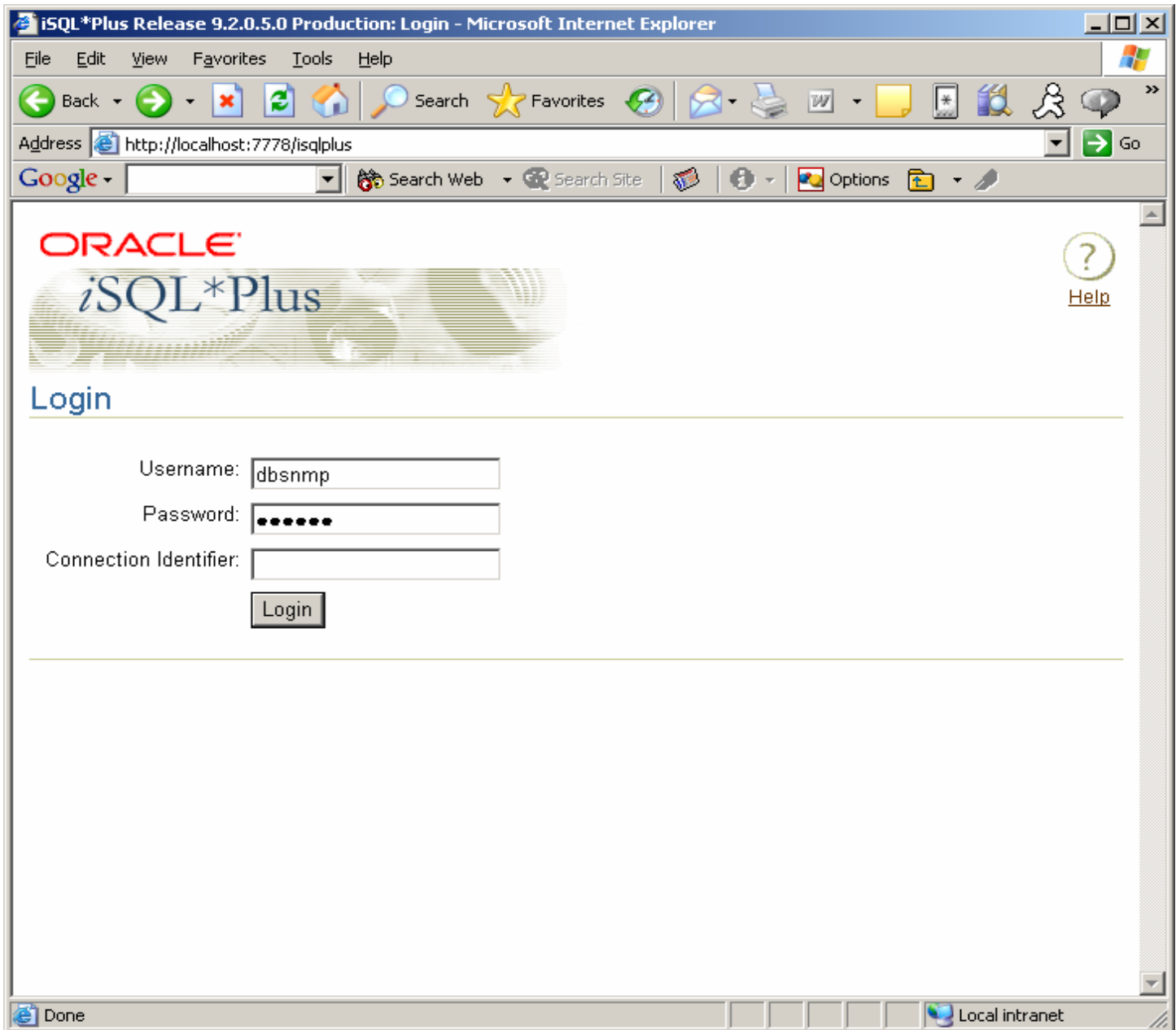
The screenshot above shows several of the websites which were found to be running iSQL\*Plus exposed to the Internet. We can try this same search on Yahoo as well. This time rather than using Google's allinurl option, we will simply search for text we know exists on the iSQLPlus webpage. In this case we will search on "iSQL\*Plus Release".

The screenshot shows a web browser window with the Yahoo! search engine. The search query is "iSQL\*Plus Release". The search results are as follows:

- [iSQL\\*Plus Release 9.2.0.1.0 Production: Login](#)  
Help. Login. Username: Password: Connection Identifier:  
[gettysburg.wccnet.edu:7777/isqlplus](#) - 3k - [Cached](#) - [More from this site](#)
- [iSQL\\*Plus Release 9.0.1](#)  
Script Location: Enter statements:  
[student.cob.ohiou.edu/jb250299/sqlweb.htm](#) - 20k - [Cached](#) - [More from this site](#)
- [iSQL\\*Plus Release 9.0.1](#)  
Script Location: Enter statements:  
[student.cob.ohiou.edu/jb250299/sarasql.htm](#) - 23k - [Cached](#) - [More from this site](#)
- [iSQL\\*Plus Release 9.2.0.5.0 Production: Login](#)  
Help. Login. Username: Password: Connection Identifier:  
[isqlplus.it.swin.edu.au:7777/isqlplus](#) - 3k - [Cached](#) - [More from this site](#)
- [What's New in SQL\\*Plus?](#)  
... Any user customizations can be manually merged into the default **iSQL\*Plus Release 9.2** configuration file ... There are several new parameters for sizing and tuning **iSQL\*Plus Release 9.2** ...  
[cs.utah.edu/classes/cs6530/oracle/.../server.920/a90842/whatsnew.htm](#) - 30k - [Cached](#) - [More from this site](#)
- [iSQL\\*Plus Release 10.1.0.2](#)  
\* Indicates required field. Username. Password. Connect Identifier. Help. Copyright © 2003, Oracle. All rights reserved.  
[www.onlinecreation.com:5560/isqlplus](#) - 9k - [Cached](#) - [More from this site](#)
- [SQL\\*Plus FAQ](#)  
SQL\*Plus and iSQL\*Plus Frequently Asked Questions  
[otn.oracle.com/sunnet/tech/sql\\_plus/htdocs/runtime.html](#) - 47k - [Cached](#) - [More from this site](#)

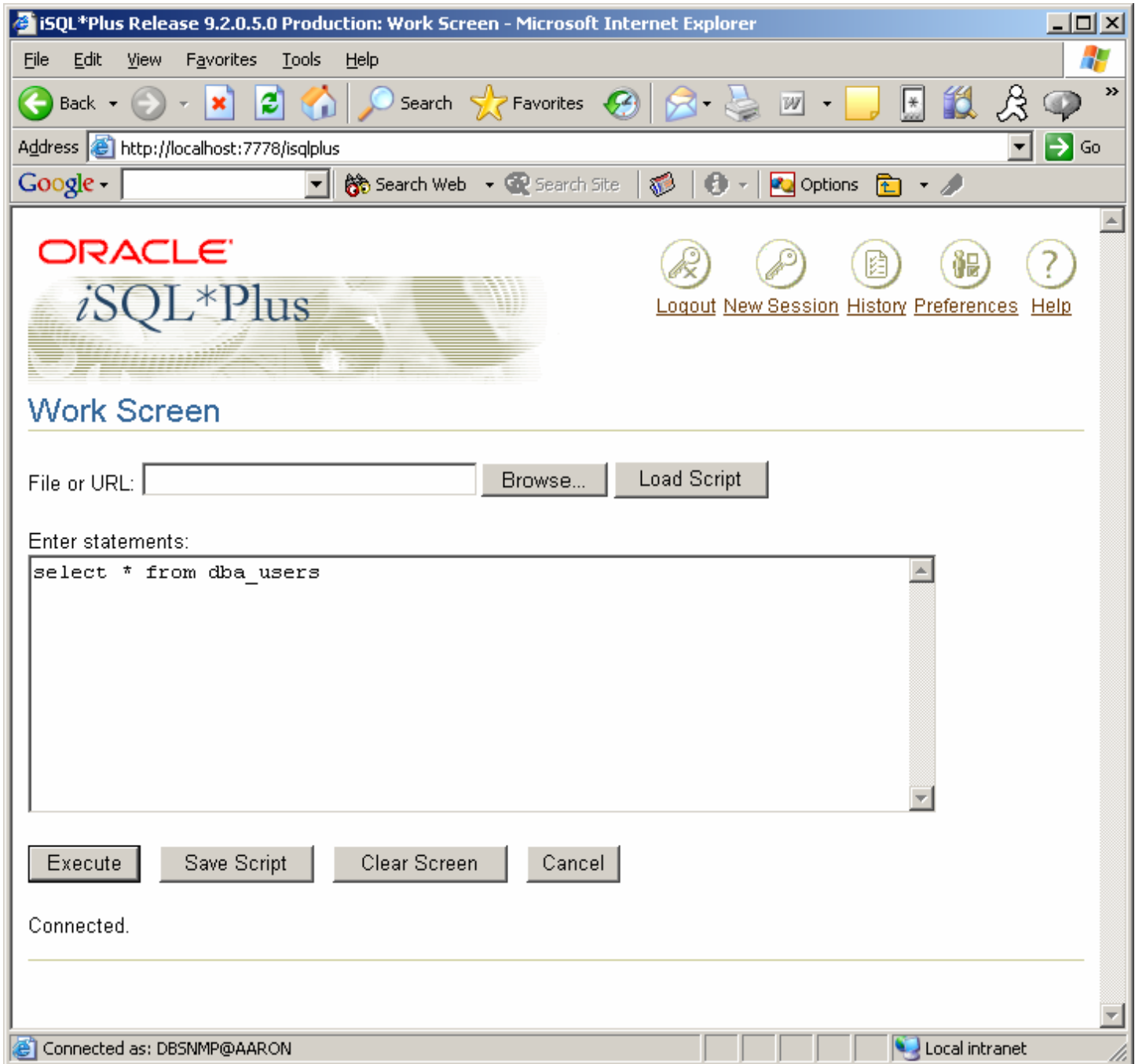
Once again many sites exposing versions of Oracle's iSQL\*Plus to the internet are discovered. The combinations of text strings to search for are endless, as are the number of search engines that can be queried. Clearly there is a large number of Oracle databases exposed out there. Of those databases, it's very likely that a significant portion will have unpatched security vulnerabilities, including default accounts and passwords.

To demonstrate how simple it is to exploit this security problem, we will be simulating an attack on a machine in the Application Security, Inc. test lab. Below is an example of iSQL\*Plus running on an Oracle HTTP Server. This database and web application are the result of a default installation of the Oracle9i Release 2 with patchset 9.2.0.5 and the patch for Oracle Security Alert #68 (<http://www.appsecinc.com/resources/alerts/oracle/2004-0001/>).



In the screenshot we have entered one of the most common default username and password for Oracle – DBSNMP/DBSNMP. This account trumps the more traditional and better known SCOTT/TIGER combination because DBSNMP is a privileged account. Application Security's research indicates that somewhere between twenty and fifty percent of all Oracle databases out there are running with at least one default username and password. A list of default Oracle usernames and passwords can be found at <http://www.cirt.net/cgi-bin/passwd.pl?method=showven&ven=Oracle>.

For demonstration purposes, this password combination worked in our lab. Now that we've got an authenticated web user, we can start to execute some commands against the database. In this case, we will select the username and password hashes out of the DBA\_USERS table.



Oracle graciously responds, displaying the table of usernames and password hashes to the web browser:

WORK Screen

File or URL:  Browse... Load Script

Enter statements:

```
select * from dba_users
```

Execute Save Script Clear Screen Cancel

USERNAME	USER_ID	PASSWORD	ACCOUNT_STATUS	LOCK_DATE	EXPIRY_DA	DEFAULT_TABLES
SYS	0	9BDBFDA5BC24F760	OPEN			SYSTEM
SYSTEM	5	9BC141C650274F20	OPEN			SYSTEM
OUTLN	11	4A3BA55E08595C81	OPEN			SYSTEM
CTXSYS	33	24ABAB8B06281B4C	OPEN			DRSYS
DBSNMP	19	5066D314D54210CC	OPEN			SYSTEM

Connected as: DBSNMP@AARON Local intranet

### SQL INJECTION IN DEMO APPLICATIONS

Search engines can also be used to find sites with known security holes. This can be as simple as searching for a URL containing the name of the vulnerable web page or application. Oracle ships several sample web applications along with its databases. These applications are enabled by default, listening on port 7777, and known to be vulnerable to SQL Injection. This section focuses using search engines to attack Oracle databases by exploiting known vulnerabilities in Oracle's sample web applications.

We will use two different applications for this demonstration. These apps are typically accessed using the following two URLs:

1. /demo/sql/jdbc/JDBCQuery.jsp
2. /demo/sql/tag/sample2.jsp

The security holes in these applications allow a web user to exploit SQL Injection to submit arbitrary SQL statements to the database. We will get started by looking for instances of URL #1 above, the JDBC Query application. We used Google to search for “allinurl:JDBCQuery.jsp”. Google responded with a number of matches, essentially a list of sites that a hacker can easily attack.

Yahoo can also find sites that expose these sample pages. Just search on the string “Please enter a suitable JDBC connection string, before you try the above demo”. As we will demonstrate below, there are quite a few web sites out there that look like they are running Oracle HTTP Servers with the demo applications enabled.

The screenshot shows a Yahoo! search results page in Microsoft Internet Explorer. The search query is "Please enter a suitable JDBC connection string, before you try the above demo". The results list five links, each with a snippet of the page content:

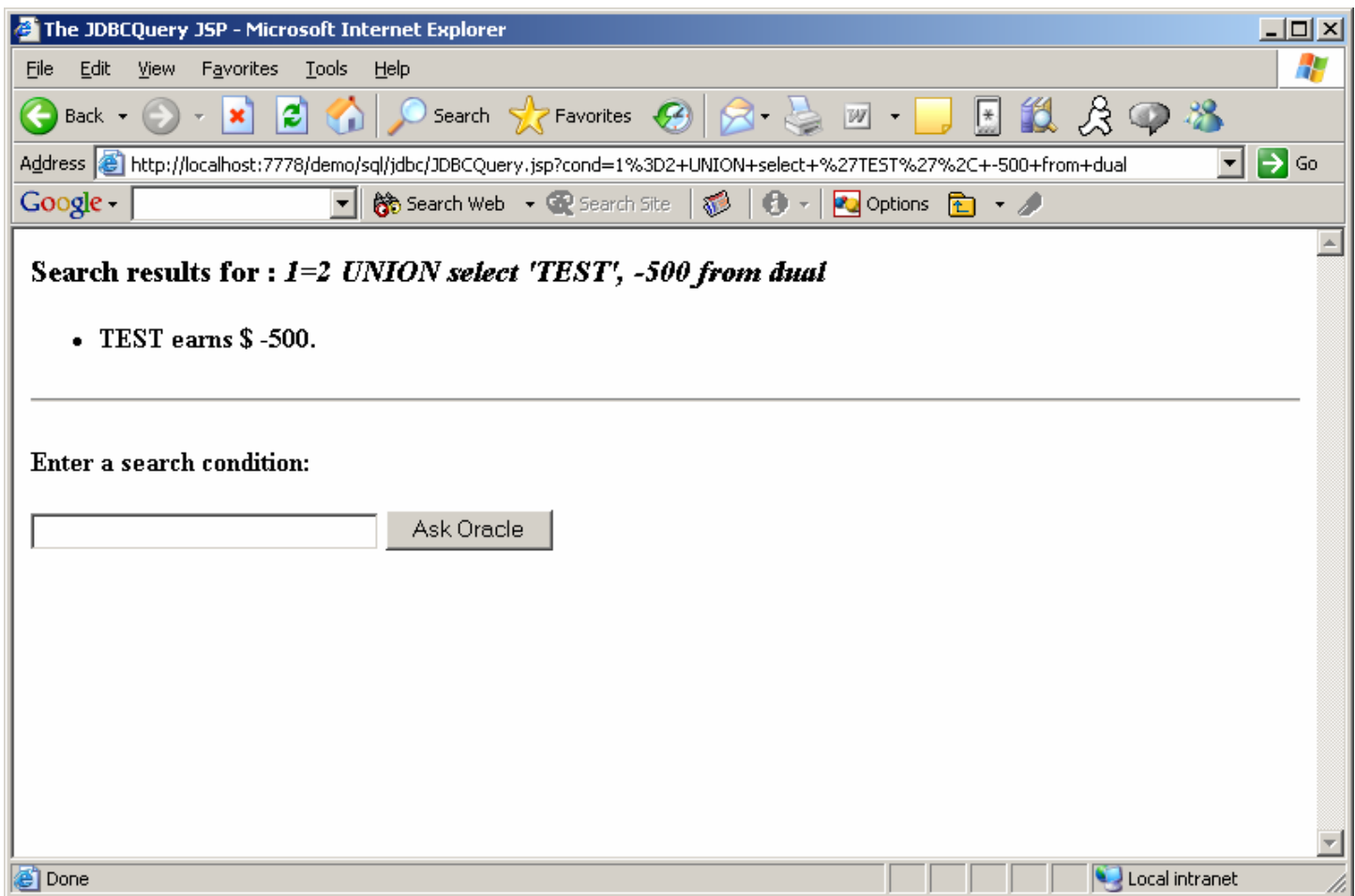
1. <http://coreapps2.evosource.net/demo/xml/xmlquery/XMLQuery.jsp>  
Please enter a suitable JDBC connection string, before you try the above demo  
coreapps2.evosource.net/demo/xml/ xmlquery/XMLQuery.jsp - 608 - Cached - More from this site
2. <http://infotrek.er.usgs.gov/demo/sql/sqlj/SQLJIterator.sqljsp>  
Please enter a suitable JDBC connection string, before you try the above demo. To use the thin driver insert your host, port and database id.  
infotrek.er.usgs.gov/demo/sql/sqlj/ SQLJIterator.sqljsp - 672 - Cached - More from this site
3. <http://ias.itec.suny.edu/demo/sql/tag/sample5.jsp>  
Please enter a suitable JDBC connection string, before you try the above demo  
ias.itec.suny.edu/demo/sql/tag/ sample5.jsp - 303 - Cached - More from this site
4. [OracleJSP](#)  
... Please enter a suitable JDBC connection string, before you try the above demo ...  
msdemo.msolutions.com/ojspdemos/ sql/index.jsp - 4k - Cached - More from this site
5. [XML and XSL Tag Support](#)  
... <font size=+0> <B>Please enter a suitable JDBC connection string, before you try the above demo</B> <pre> To use the ...  
deakin.edu.au/div\_its/isg/dba/docs/ 9iasrel2/web.902/a95883/xmlxsl.htm - 43k - Cached - More from this site

We will execute an actual attack in the AppSecInc test lab. When first connecting to the JDBC Query application, you are prompted to enter information about a database to which you want to connect (JDBC Connection Configuration). Picking or guessing this value correctly takes a little luck or intuition. We will return to this point a little later. Note: by default the sample applications connect to the database backend using the non-privileged account SCOTT.

From the web page <http://hostname/demo/sql/jdbc/JDBCQuery.jsp>, we can enter a simple SQL statement to demonstrate the vulnerability to SQL injection. For example, we enter:

```
1=2 UNION select 'TEST', -500 from dual
```

Below are the results of executing our SELECT from DUAL statement.



As we mentioned earlier, this web application connects to the database using the non-privileged account SCOTT. Does this mean that it can't be used to do anything dangerous? Of course not! Being a good security disciple, you should know that SQL injection is a problem even if you don't know how it can be exploited. The fact is, someone out there probably knows a way. Actually, they probably published the exploit for all to enjoy. Let's take a look at how some sample exploit code taken from a public source - the Full Disclosure news group - can be used to exploit SQL injection to gain control over the backend database.

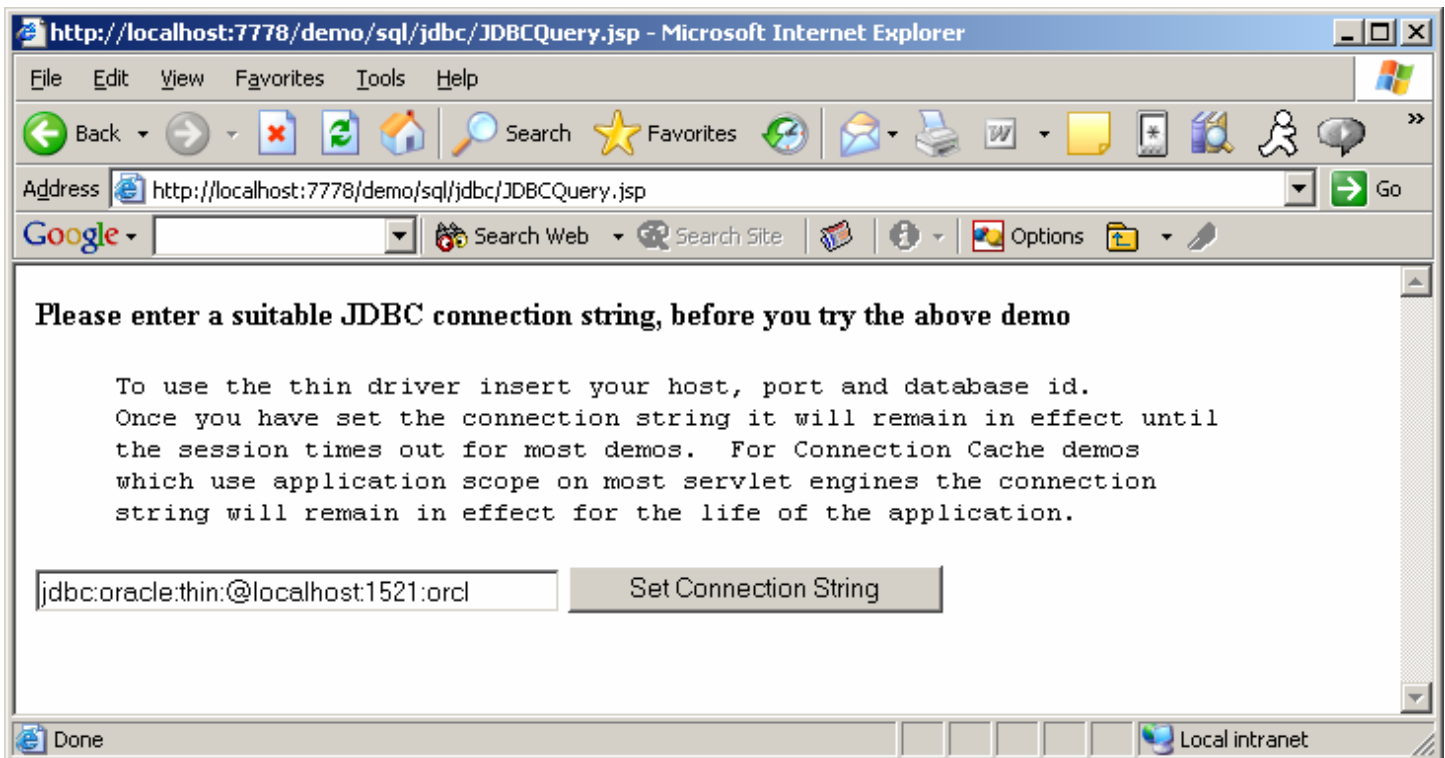
This attack takes advantage of a vulnerability in the Oracle built-in function called NUMTOYMINTERVAL (Oracle recently released a patch to remediate this issue). This function is available to any user (including SCOTT) and cannot be removed or restricted. The patch for this vulnerability is not included in the default Oracle installation.

Utilizing the exploit code from Full Disclosure, an attacker can use the sample JDBCQuery.jsp application to run arbitrary operating system commands under the OS privileges of the Oracle database (typically full administrative rights). In the example below, a hacker can select which commands to run by replacing the string “echo ARE YOU SURE?” with any operating system commands.

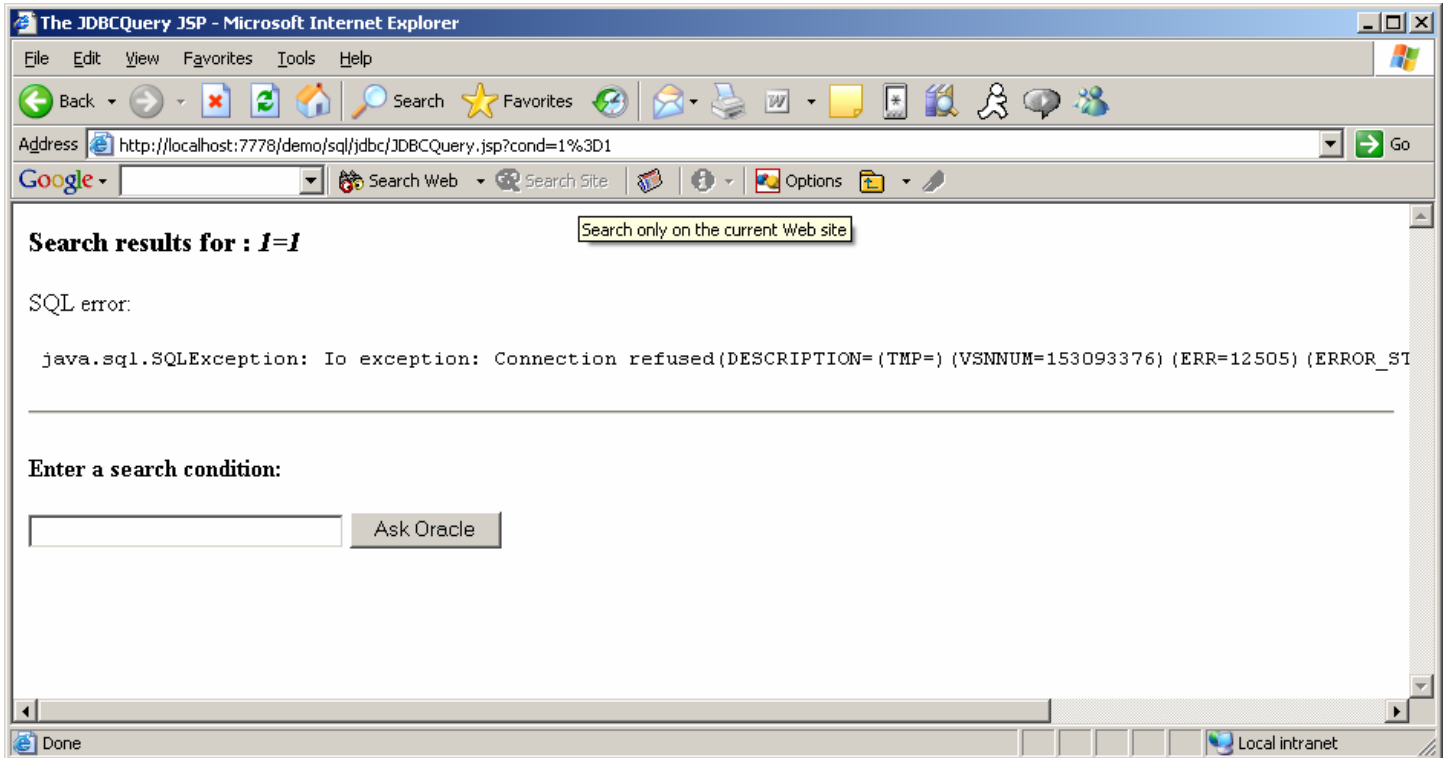
Example Attack String (note: several bytes have been changed from the original exploit to ensure this code will not work):

```
'1'='2' UNION SELECT
NUMTOYMINTERVAL(1,'AAAAAAAAAABBBBBBBBBBCCCCCCCCCABCDEFHGHIJKLMNOPQR' || chr(59) ||
chr(79) || chr(150) || chr(01) || chr(141) || chr(68) || chr(36) || chr(18) || chr(80) || chr(215) || chr(21) || chr(52) ||
chr(35) || chr(148) || chr(01) || chr(255) || chr(37) || chr(172) || chr(30) || chr(148) || chr(01) || chr(32) || 'echo ARE
YOU SURE? >c:\Unbreakable.txt'), 1 from dual
```

Taking a step back, we’ve yet to discuss getting past the JDBC Connection Configuration screen mentioned earlier. In order to execute the attack we’ve described above, we’ll need to enter a valid connection string to attach to the backend database. Below is an example of this screen:



By default the connection string is setup pointing to the localhost port 1521, using the SID of ORCL. Often this configuration is the correct choice, but not always. If it does not work, a guess will have to be made on the SID. The error below is returned when an invalid SID is selected.



This error message (Invalid SID) is very different from the error message that is returned when the port or IP address does not respond. If the port or IP address is not alive, the following error message is returned.

### SQL error:

```
java.sql.SQLException: Io exception: The Network Adapter could not establish the
connection
```

These error messages represent yet another security hole in the JDBC Query application. An attacker can use this information to probe the internal network for live Oracle databases, or simply for live IP addresses and ports. By entering different values for IP addresses, port and SID, then observing the responses, an anonymous user on the Internet can scan the entire internal network, regardless of perimeter security measures (firewalls)! An attacker would likely use this loophole to map any live Oracle databases running in the target's internal network. The attacker would then use this single Oracle HTTP Server running the sample applications to connect to each Oracle database individually, repeatedly running the attack as he goes. In this method, a single exposed application can lead to the exploitation of all internal database resources.

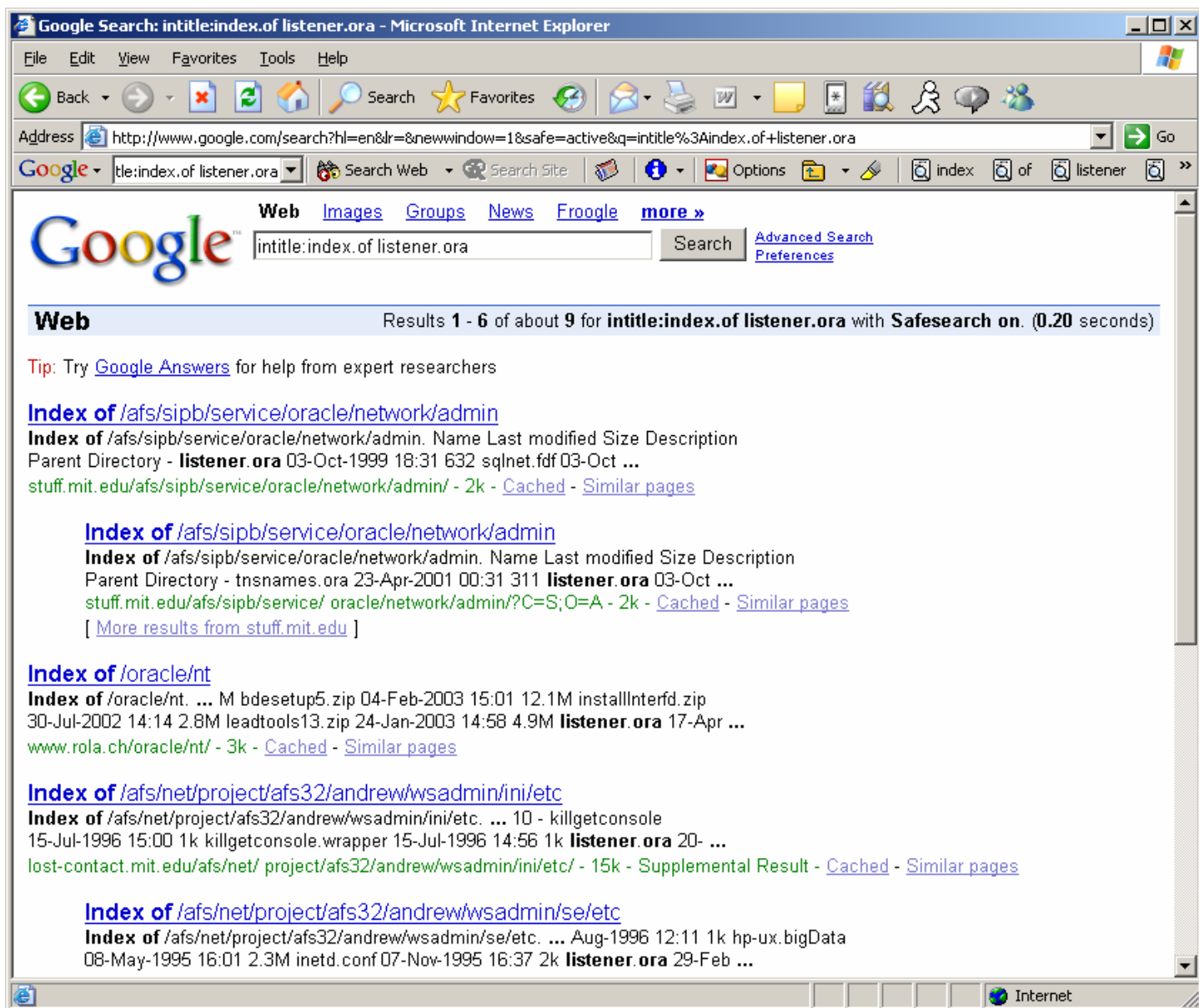
### *LOOKING FOR DIRECTORY INDEXING*

A common feature of many web servers is the ability to display a list of files in a directory when the client passes the name of the directory as the URL. This feature is commonly referred to as directory browsing or indexing. The results of directory browsing look very similar to viewing a directory on the local file system.

When a system has directory browsing enabled, it is simple for a client to enumerate and access any files retrievable by the web server; even those files that are not linked to a web application. It is possible that some directories hold files that contain sensitive information. If a directory contains any sensitive files, directory browsing exposes them to the public.

A side effect of enabling directory browsing relates back to search engines. If a search engine can find a file, it will be inventoried, even if it contains sensitive data. An attacker can therefore find web sites that have directory indexing enabled, with a search using the tag “intitle” looking for the value of “Index of”. The text “Index of” is found in the title of any page with directory indexing enabled.

One sensitive file that is often exposed by directory indexing is listener.ora. This Oracle configuration file lives on the node on which the listener service runs. It contains the parameters the listener uses including the SID names, IP addresses, port numbers, and even the listener password. Below is an example of a typical search for listener.ora files using Google.



By digging through these files, an attacker will find two useful pieces of information:

1. The location of any databases known to the listener(s).
2. The password(s) used by the listener(s). If a password is not in place the security situation is even worse. For more information, see [http://www.appsecinc.com/presentations/Protecting\\_Oracle\\_Databases\\_White\\_Paper.pdf](http://www.appsecinc.com/presentations/Protecting_Oracle_Databases_White_Paper.pdf).

There are actually many other Oracle files that contain sensitive data, many of these can be found using a similar technique to the one described above. Below is a sampling of these files:

1. tnsnames.ora: configuration file designed to resolve logical Oracle database names to specific IP address, port, and SIDs.
2. sqlnet.ora: configuration file designed to set the parameters used by the SQL\*Net protocol. Among these parameters is included the SQLNET.CRYPTO\_SEED used during encryption of network traffic.
3. sqlnet.log: output file generated when attempting to connect to a database. Contains information such as username, IP address, port, SID of the database, and database version.
4. pwdSID.ora: contains password hashes for the accounts that are granted SYSDBA privileges.

Once this information has been obtained, the work of breaking into the actual databases becomes much easier and less time consuming. Clearly these are critical holes that need to be sealed, regardless of the security at the network perimeter.

### **MITIGATING THE THREAT**

Which is the right approach to database security? It starts with proper resource allocation, and a defense-in-depth strategy. Most organizations have already taken steps to limit direct database access from the outside world. This is just the first layer of defense, it's important that we all realize that. At Application Security, we regularly talk with DBAs that believe security is not a real concern because the database is far behind a firewall. This outdated perspective has to be changed. It's now the time to look at placing additional layers of defense to lock down the data right where it lives – in the database.

Database security concepts are not much different than the concepts that have been implemented in the network security world. Start by eliminating as many security holes as possible using a vulnerability assessment and penetration testing tool. Next, implement a method of continuously monitoring for attacks and malicious behavior with an intrusion detection and security auditing solution. Finally, as a last line of defense, encrypt the most sensitive data and tightly control its use with a column-level encryption product. This suite of defense mechanisms offers the best chance to stop an attack before or while it happens, but also offers protection in the case that an attacker succeeds in stealing data from the database.

## CONCLUSION

This new search engine attack is particularly troublesome for several reasons. This is not a new vulnerability, but rather a more effective way of exploiting existing known-vulnerabilities. This makes a hacker's work much easier, exposed systems will be found and attacked. It is now more important than ever to properly secure and lock down your databases and applications.

Some folks even say that the answers to these issues lie in legislation; that governments need to step in and create software liability laws. This sounds good to some, but because of the ambiguity surrounding the issue, is unlikely to lead to meaningful controls. As we have seen, a hacker can use the content provided by a search engine to acquire targets for attack. The hacker may even be able to see credit card numbers, patient information, and other sensitive data directly via a search engine. This really clouds the potential legal issues surrounding the attack. If an attacker locates credit cards picked up by Google in a standard web crawl, who is in the wrong? The attacker that got the data from Google? Does Google bear any responsibility for crawling and displaying those credit cards? Or, does all the blame lie in the system administrator that exposed the credit card numbers in the first place? Does Oracle bear part of the blame for delivering software with a vulnerable default configuration? These questions will not be easy to answer, and any proposed legislation will likely be extremely unpopular with various powerful lobbying groups. Clearly, the legal route is not a viable short term solution.

Application Security offers a suite of products designed to provide enterprise-class database security and auditing. Our flagship application-focused vulnerability assessment product, AppDetective™ locates, details, and helps to fix security holes in applications within your network. Armed with a revolutionary security methodology, together with an extensive knowledgebase of application level vulnerabilities, AppDetective will find applications, scan them for vulnerabilities, then generate reports to help understand and fix your security holes and misconfigurations.

A demo version of AppDetective can be downloaded from <http://www.appsecinc.com/products/appdetective/>.

AppDetective currently supports the following platforms:

- Oracle
- Microsoft SQL Server
- Sybase
- IBM DB2 UDB
- IBM DB2 for Mainframe
- MySQL
- Oracle Application Server
- Lotus Domino
- BEA WebLogic
- IBM WebSphere
- Generic Web Applications

**BIBLIOGRAPHY**

The Google Hacker's Guide v1.0

<http://johnny.ihackstuff.com/modules.php?op=modload&name=Downloads&file=index&req=getit&lid=34>

Demystifying Google Hacks

<http://www.hackingspirits.com/eth-hac/papers/Demystifying%20Google%20Hacks.pdf>

Google: Net Hacker Tool du Jour

<http://www.wired.com/news/infostructure/0,1377,57897,00.html>

Oracle Security papers written by Pete Finnegan

<http://www.petefinnigan.com/orasec.htm>

CONFIGURING A SECURE ORACLE9I APPLICATION SERVER ENVIRONMENT - Oracle Corporation

Securing Oracle9iAS 1.0.2.x

Web Application Worms: Myth or Reality?

[http://www.imperva.com/application\\_defense\\_center/white\\_papers/application\\_worms.html](http://www.imperva.com/application_defense_center/white_papers/application_worms.html)

An Introduction to Database and Application Worms

[http://www.appsecinc.com/presentations/Database\\_Application\\_Worms.pdf](http://www.appsecinc.com/presentations/Database_Application_Worms.pdf)

Protecting Oracle Databases

[http://www.appsecinc.com/presentations/Protecting\\_Oracle\\_Databases\\_White\\_Paper.pdf](http://www.appsecinc.com/presentations/Protecting_Oracle_Databases_White_Paper.pdf)

**ABOUT APPLICATION SECURITY, INC.**

AppSecInc is the leading provider of database security solutions for the enterprise. AppSecInc products proactively secure enterprise applications at more than 250 organizations around the world by discovering, assessing, and protecting the database against rapidly changing security threats. By securing data at its source, we enable organizations to more confidently extend their business with customers, partners and suppliers. Our security experts, combined with our strong support team, deliver up-to-date application safeguards that minimize risk and eliminate its impact on business. Please contact us at 1-866-927-7732 to learn more, or visit us on the web at [www.appsecinc.com](http://www.appsecinc.com).