

Re: JDBC PreparedStatements, Java Data Objects/O-R mapping, and SQL Injection

Source: <http://www.derkeiler.com/Mailing-Lists/securityfocus/secprog/2003-01/0017.html>

From: Jeff Williams @ Aspect (@)

Date: 01/03/03

From: "Jeff Williams @ Aspect" <jeff.williams@aspectsecurity.com>

To: "Kevin Spett" <kspett@spidynamics.com>, <secprog@securityfocus.com>, <webappsec@securityfocus.com>

Date: Fri, 3 Jan 2003 14:01:43 -0500

The latest JDBC spec says:

"PreparedStatement objects represent SQL statements that can be prepared, or precompiled, for execution once and then executed multiple times. Parameter markers, represented by "?" in the SQL string, are used to specify input values to the statement that may vary at runtime."

But "precompiled" is never discussed or defined (it's not even technically required since the word "may" is used). But let's look at a real example, the MySQL JDBC driver, since it's open source and widely used. When a PreparedStatement is constructed, the query string is parsed into pieces separated by the "?" marker. Then the user uses the setXXX() methods to fill in the objects. Then when executeQuery() is called, the full query string is assembled, written to the database, and executed. The only "precompiling" is the simple parsing in the constructor. I see no protection against SQL injection gained by using PreparedStatements, at least with the MySQL driver. NOTE: I haven't confirmed this with a test program yet.

Now just to be clear, I think using PreparedStatements is a good idea. But since the spec doesn't provide any meaningful protection against SQL injection, I think anyone using JDBC ought to do their own protection. Then if they ever change drivers they will still be protected. There is really no way to know what a closed source database driver does behind the scenes.

--Jeff

Jeff Williams

Aspect Security, Inc.

<http://www.aspectsecurity.com>

----- Original Message -----

From: Kevin Spett

To: Jeff Williams @ Aspect ; secprog@securityfocus.com ;

Re: JDBC PreparedStatements, Java Data Objects/O-R mapping, and SQL Injection

webappsec@securityfocus.com

Sent: Friday, January 03, 2003 11:01 AM

Subject: Re: JDBC PreparedStatements, Java Data Objects/O-R mapping, and SQL Injection

As far as I can tell, the JDBC spec requires that a PreparedStatement be precompiled. This has the effect of separating the client-supplied values from the SQL statement, which prevents SQL injection. Every database server/JDBC API I have seen does this. Does anyone know of any exceptions?

Now of course, you can still shoot yourself in the foot programming with PreparedStatement if you build them by concatenating client-supplied data into them as opposed to using the '?' substitutions. But not only is that insecure, it also completely defeats the purpose of using PreparedStatement.

Kevin Spett

SPI Labs

<http://www.spidynamics.com/>

----- Original Message -----

From: "Jeff Williams @ Aspect" <jeff.williams@aspectsecurity.com>

To: "Kevin Spett" <kspett@spidynamics.com>; "Dave Aitel" <dave@immunitysec.com>; <webappsec@securityfocus.com>

Sent: Monday, December 30, 2002 10:37 PM

Subject: Re: JDBC PreparedStatement, Java Data Objects/O-R mapping, and SQL Injection

> *I think there's a very important point here about specifications and security guarantees.*

>

> *Not to rehash the whole discussion from the old PreparedStatement thread, but nothing in the JDBC spec precludes SQL injection from working with a PreparedStatement. That means that it depends on how the JDBC driver is implemented and what support is provided in the database for pre-compiling queries. I've checked the source for a few open-source JDBC drivers, and there is definitely room for security improvements. Who knows what's going on under the covers in a proprietary JDBC driver.*

>

> *Excellent question about OR mapping technologies and what I'll call "OQL injection." For those who don't know, OQL is a subset of SQL used to query objects from an object store that is generally backed by a relational database. I checked the Castor JDO implementation, and it uses a PreparedStatement under the hood, so it appears to be resistant to these attacks (depending on your JDBC driver and database). The translation from OQL to SQL is done with a *very* simple parser based on StringTokenizer. The JDO spec is silent on use of PreparedStatement and SQL injection, so there are no guarantees that your JDO implementation is resistant to OQL injection.*

>
> *In both of your questions, the specs don't detail the security*
> *guarantees -- meaning that if you want security you have to build it*
> *yourself. Even if you are currently not susceptible because your code*
> *is running with a strong driver/database, you have a latent flaw waiting*
> *to bite you.*
>
> *Bottom line -- if the spec doesn't guarantee it, you should protect your*
> *app against it. Using PreparedStatement *may* help, but that protection*
> *may disappear when you change platforms a few years out. In my opinion,*
> *the right approach here is to very carefully validate parameters*
> *yourself before they are used in any kind of JDBC query.*
>
> *--Jeff*
>
> *Jeff Williams*
> *Aspect Security, Inc.*
> *<http://www.aspectsecurity.com>*
>
>
> ----- Original Message -----
> *From: Kevin Spett*
> *To: Dave Aitel ; webappsec@securityfocus.com*
> *Sent: Monday, December 30, 2002 6:48 PM*
> *Subject: Re: JDBC PreparedStatement, Java Data Objects/O-R mapping, and*
> *SQL Injection*
>
>
> *Stored procedures by themselves do not provide protection, sorry if I*
> *worded*
> *that poorly. Prepared statements, *combined* with prepared statements*
> *do,*
> *which is how I meant that statement to be interpreted. Of course,*
> *"impossible" should be taken with a grain of salt.*
>
>
> *Kevin Spett*
> *SPI Labs*
> *<http://www.spidynamics.com/>*
>
> ----- Original Message -----
> *From: "Dave Aitel" <dave@immunitysec.com>*
> *To: <webappsec@securityfocus.com>*
> *Sent: Monday, December 30, 2002 6:14 PM*
> *Subject: Re: JDBC PreparedStatement, Java Data Objects/O-R mapping, and*
> *SQL*
> *Injection*
>
>
> > *I dunno about that. Impossible is such a big word, and I've seen SQL*
> > *Injection successfully done at least few times against a stored*

> > *procedure.*
> >
> > *You should put your sample apps on a web site somewhere so people can*
> > *knock it around a bit.*
> >
> > *Dave Aitel*
> > *Immunity, Inc.*
> > *<http://www.immunitysec.com/CANVAS/> (Remote SQL Server exploits make*
> > *SQL*
> > *Injection even more fun than usual!)*
> >
> >
> > *On Mon, 30 Dec 2002 17:32:13 -0500*
> > *"Kevin Spett" <kspett@spidynamics.com> wrote:*
> >
> > > *The use of prepared statements and stored procedures makes SQL*
> > > *injection impossible. A prepared statement is compiled before the*
> > > *user input is added to the SQL statement, effectively making it*
> > > *impossible to execute the client-supplied data because it is never*
> > > *compiled. There was a thread about this a couple of months back on*
> > > *this list, here's the first post:*
> > >
> > > *<http://archives.neohapsis.com/archives/sf/www-mobile/2002-q3/0105.html>*
> > >
> > > *Have a fun and securely programmed new year, everyone.*
> > >
> > > *Kevin Spett*
> > > *SPI Labs*
> > > *<http://www.spidynamics.com>*
> > >
> > >
> > > ----- Original Message -----
> > > *From: "Christopher Todd" <chris@christophertodd.com>*
> > > *To: <webappsec@securityfocus.com>*
> > > *Sent: Monday, December 30, 2002 3:29 PM*
> > > *Subject: JDBC PreparedStatements, Java Data Objects/O-R mapping, and*
> > > *SQL Injection*
> > >
> > >
> > > > *I am working on the Java language section of the OWASP Guide to*
> > > > *Securing*
> > > > *Web*
> > > > *Applications, and I have a question for the list. Have any of you*
> > > > *elite*
> > > > *SQL*
> > > > *Injectors ever been able to hack an application that was using*
> > > > *JDBC*
> > > > *PreparedStatements? Are any of you aware of a theoretical reason*
> > > > *this should be impossible? I have tried, and been unsuccessful,*
> > > > *to*
> > > > *perform SQL injection on an example app I coded up, but then*

> again,
>>> *I am not the*
>>> *world's*
>>> *most talented SQL Injector.*
>>>>
>>>> *On another note, have any of you ever successfully used SQL*
>>>> *Injection against a web app that was using Castor JDO, or other*
>>>> *similar Object–Relational mapping tools? Again, I have tried to*
>>>> *attack an example app I coded up and failed. Same question – is*
> *it*
>>>> *theoretically impossible to execute SQL injection against apps*
> *coded*
>>>> *using these techniques and tools?*
>>>>
>>>> *I ask these questions because I think these two techniques can be*
>>>> *used effectively to thwart (or at least make more difficult) SQL*
>>>> *injection attacks against Java–based web apps, but I want to*
>>>> *validate that belief to the best extent I can prior to putting*
> *such*
>>>> *statements into the Guide. Thanks in advance for any help you can*
>>>> *provide, as it will improve the quality and usefulness of the*
>>>> *Guide.*
>>>>
>>>> *Chris*
>>>>
>>>>
>>>
>>>
>>>
>>
>
>