




# WEB HACKING LAB



Gabriel Naples  
GJN5070@PSU.EDU

**Table of Contents**

Section I: Introduction

Page – 3

Section II

Pages 3-5

Section III

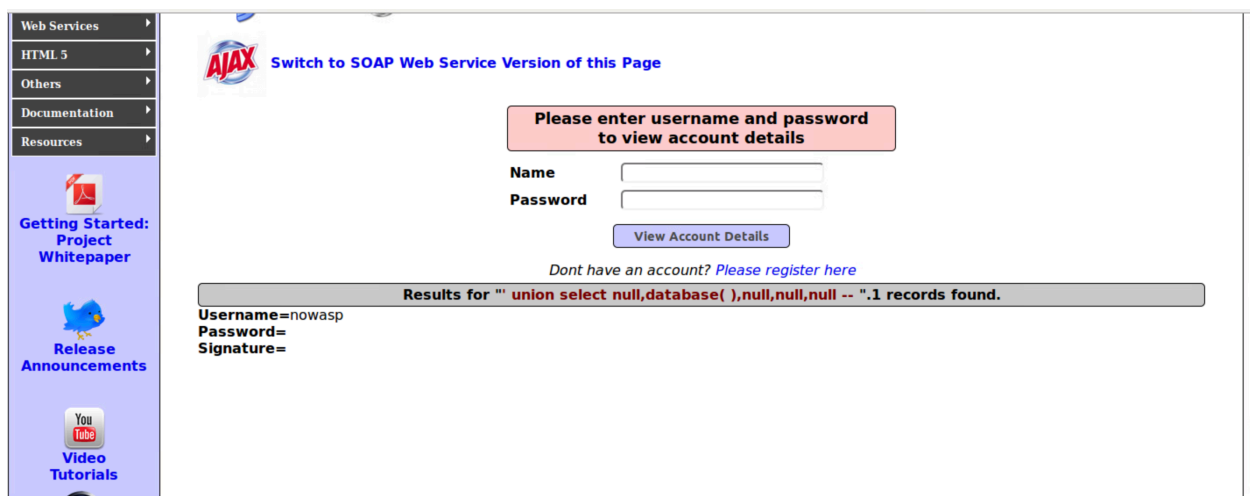
Pages 5-6

## Section I: Introduction

The goal and motivation of this project is to learn about SQL injections, the flaws and loopholes that can be found in weak code on websites, and learn a basic understanding of how to manipulate these flaws to gain access and steal information. During this lab, I expect to learn about common weaknesses in websites. I expect to gain an understanding of what exactly a SQL injection is and just how much information can be stolen using this type of attack. I also expect to learn the basics of how to carry out this attack and further my knowledge of how the Linux system and bash scripting works.

## Section II

- 1) The database that this system is running is called 'nowasp' and I found this information while testing code in the 'name' box. Also another example of the 'nowasp' table with more information can be found in the log file after the injection attack is run.



- 2) The version being used is 5.5.34, and this was found in a similar way that the database was found.

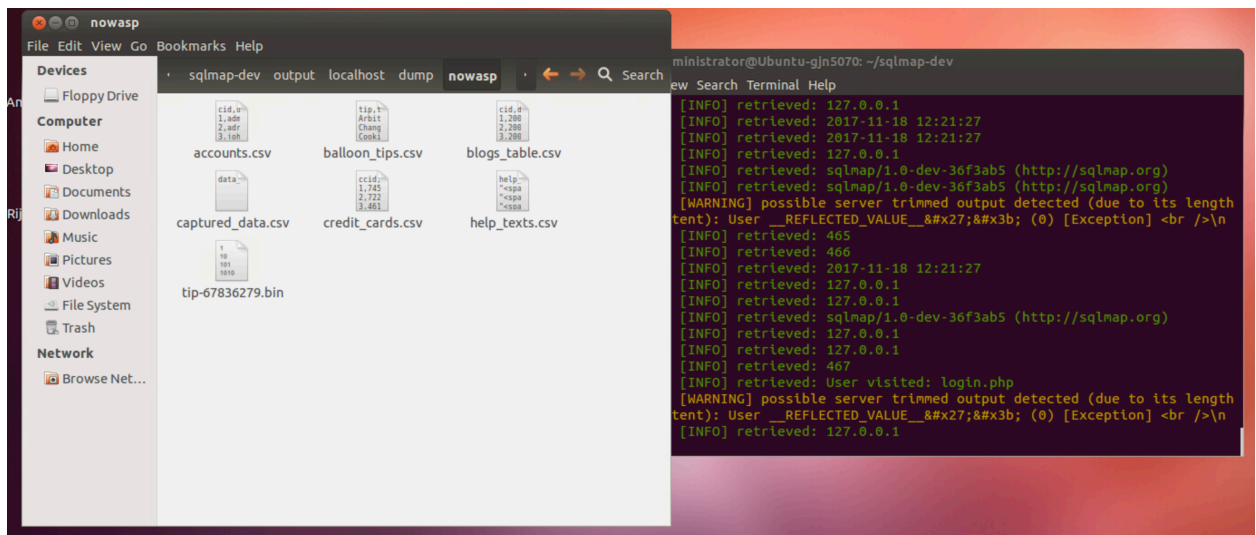


```
Database: nowasp
Table: accounts
[18 entries]
```

cid	username	is_admin	password	mysignature
1	admin	TRUE	adminpass	g0t r00t?
2	adrian	TRUE	somepassword	Zombie Films Rock!
3	john	FALSE	monkey	I like the smell of confunk
4	jeremy	FALSE	password	d1373 1337 speak
5	bryce	FALSE	password	I Love SANS
6	samurai	FALSE	samurai	Carving Fools
7	jim	FALSE	password	Jim Rome is Burning
8	bobby	FALSE	password	Hank is my dad
9	simba	FALSE	password	I am a super-cat
10	dreveil	FALSE	password	Preparation H
11	scotty	FALSE	password	Scotty Do
12	cal	FALSE	password	Go Wildcats
13	john	FALSE	password	Do the Duggie!
14	kevin	FALSE	42	Doug Adams rocks
15	dave	FALSE	set	Bet on S.E.T. FTW
16	patches	FALSE	tortoise	meow
17	rocky	FALSE	stripes	treats?
18	ed	FALSE	pentest	Commandline KungFu anyone?

- 4) The table, seen above, is called accounts and contains information pertaining to account login data of the users on the website.
- 5) Other sensitive data that was extracted from this site, along with the user data, signatures and passwords, was credit card numbers including the expiration date and CVC code.

A1	$f(x)$	$\Sigma$	=	ccid,ccv,ccnumber,expiration						
	A	B	C	D	E	F	G	H		
1	ccid,ccv,ccnumber,expiration									
2	1,745,4444111122223333,2012-03-01									
3	2,722,7746536337776330,2015-04-01									
4	3,461,8242325748474749,2016-03-01									
5	4,230,7725653200487633,2017-06-01									
6	5,627,1234567812345678,2018-11-01									
7										
8										
9										



### Section III

One method researched for securing data would be for the password data and other user data to be stored in configuration files instead of stored in the source code. This makes it harder to access and not open to everyone. Other efficient methods include

constraining input. This means you check for good and known data by validating the type, length, format, and range. Next you could use parameters with stored procedures. If you are using parameters then the input data is treated as a value instead of executable code, therefore can't run the SQL injection code. Furthermore, you can restrict permissions to specific accounts. This means that only authorized accounts have the ability to even make executable changes to the databases. Lastly, and importantly, you should make sure that the information shown within the error message doesn't give away important information about the database and the code. Users should not have access to detailed error messages.

(n.d.). Retrieved November 18, 2017, from <https://msdn.microsoft.com/en-us/library/ff648339.aspx>