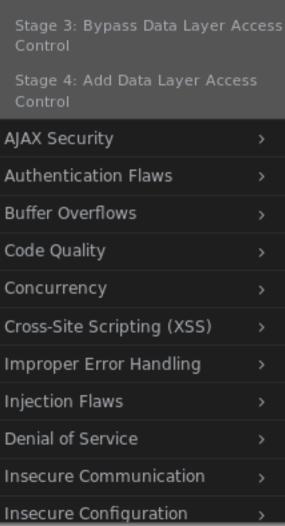


General

Control

Control



Control

AJAX Security

Authentication Flaws

Buffer Overflows

Code Quality

Concurrency

Injection Flaws

Denial of Service

stage 3, and verify that access to other employee's profiles is properly denied. * You have completed Stage 3: Bypass Data Layer Access Control. * Welcome to Stage 4: Add Data Layer Access Control **Goat Hills Financial Human Resources** Welcome Back Tom - View Profile Page First Name: Larry Last Name: Stooge 9175 Guilford Rd City/State: New York, NY Street: Phone: 443-689-0192 Start Date: 1012000 SSN: 386-09-5451 Salary: 55000 Credit Card: 2578546969853547Credit Card Limit: 5000 Comments: Does not work well with others Disciplinary Explanation: Disc. Dates: 10106 Constantly harassing coworkers Manager: 102

Access Control Flaws AJAX Security Authentication Flaws Password Strength \mathbb{C} Forgot Password Multi Level Login 1 Multi Level Login 2 Buffer Overflows Code Quality Concurrency Cross-Site Scripting (XSS) Improper Error Handling Injection Flaws Denial of Service Insecure Communication Insecure Configuration

Web applications frequently provide their users the ability to retrieve a forgotten password. Unfortunately, many web applications fail to implement the mechanism properly. The information required to verify the identity of the user is often overly simplistic. General Goal(s):

Users can retrieve their password if they can answer the secret question properly. There is no lock-out mechanism on this 'Forgot Password' page. Your username is 'webgoat' and your favorite color is 'red'. The goal is to retrieve the password of another user.

* Congratulations. You have successfully completed this lesson.

Webgoat Password Recovery

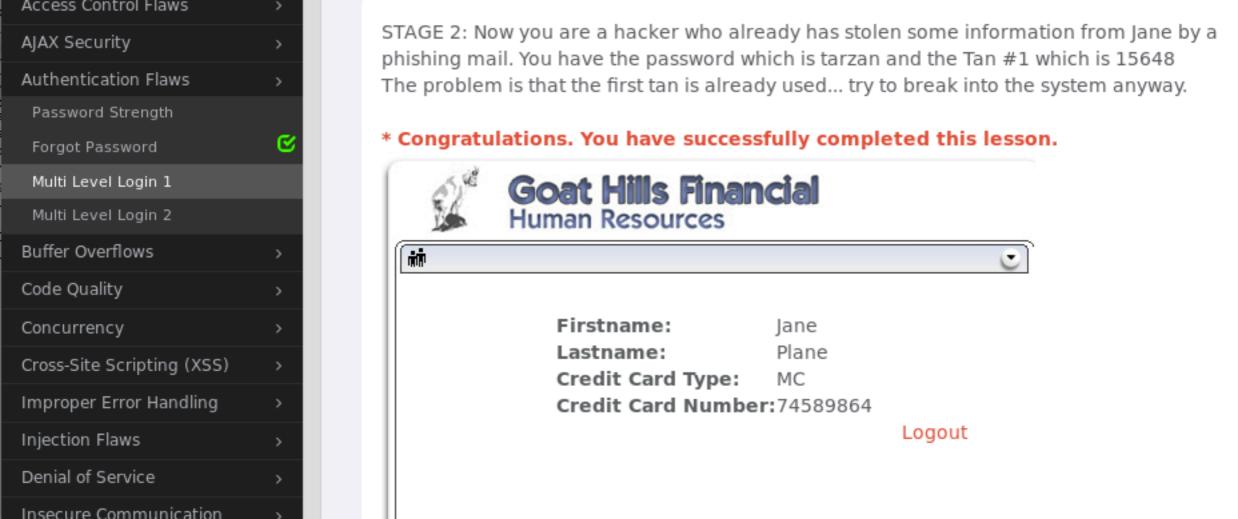
For security reasons, please change your password immediately.

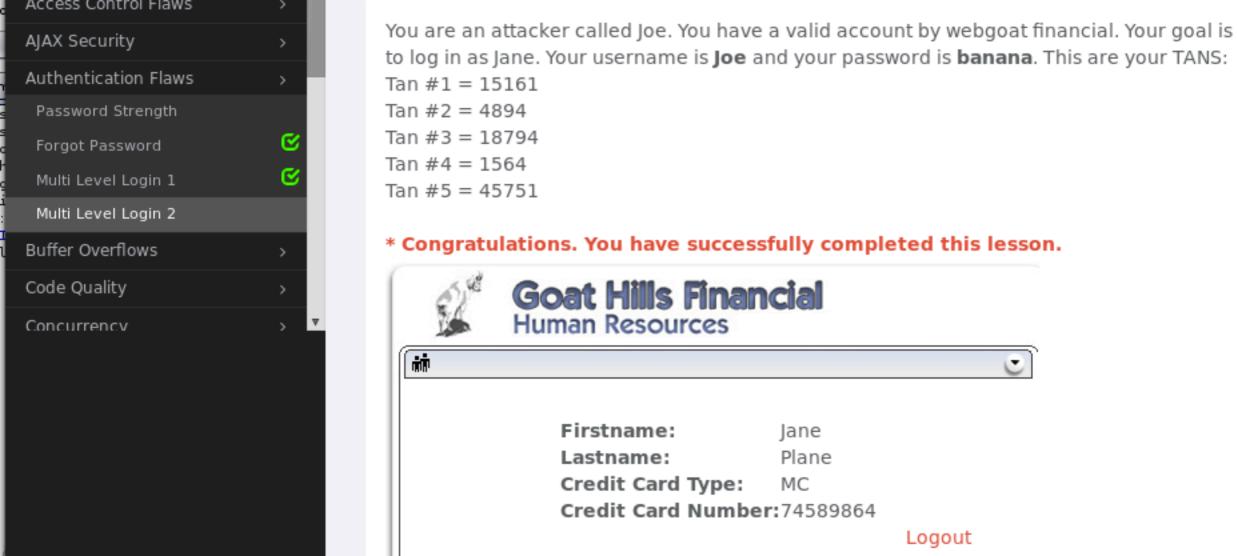
Results:

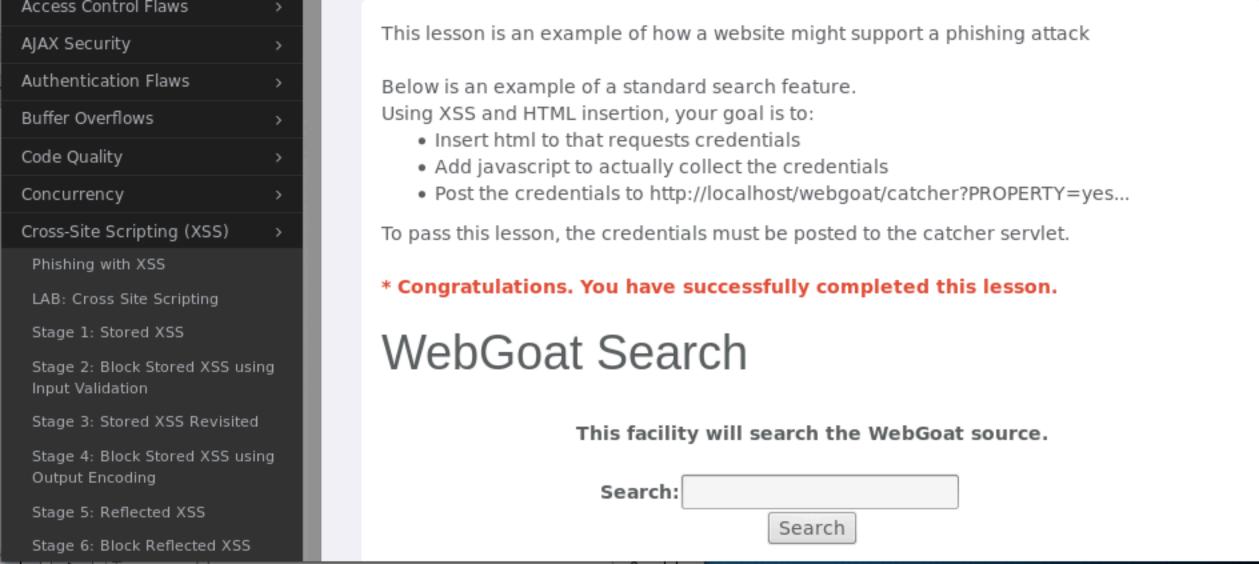
Username: admin

Color: green

Password: 2275\$starBo0rn3







Cross-site stripting (ASS) Phishing with XSS LAB: Cross Site Scripting Stage 1: Stored XSS Input Validation Stage 3: Stored XSS Revisited Stage 4: Block Stored XSS using Output Encoding Stage 6: Block Reflected XSS Stored XSS Attacks Reflected XSS Attacks Cross Site Request Forgery (CSRF) CSRF Prompt By-Pass CSRF Token By-Pass HTTPOnly Test

Cross Site Tracing (XST) Attacks

- * You have completed Stage 1: Stored XSS.
- * Welcome to Stage 2: Block Stored XSS using Input Validation

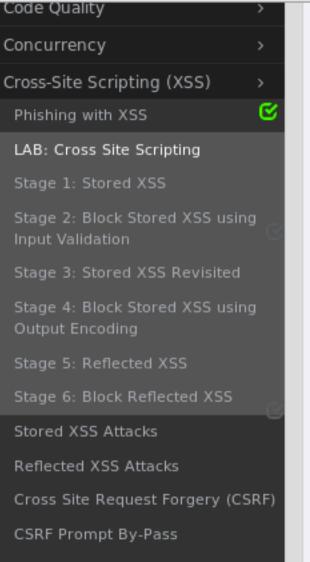


Goat Hills Financial Human Resources



ListStaff EditProfile DeleteProfile

Logout



Code Quality

Concurrency

Phishing with XSS

Stage 1: Stored XSS

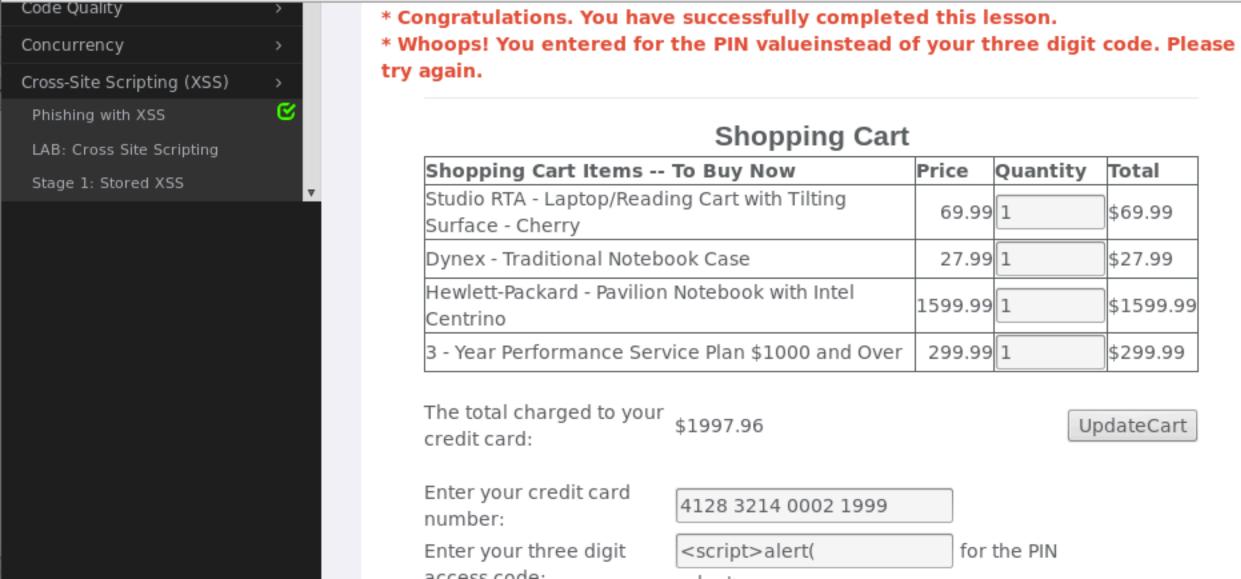
Output Encoding

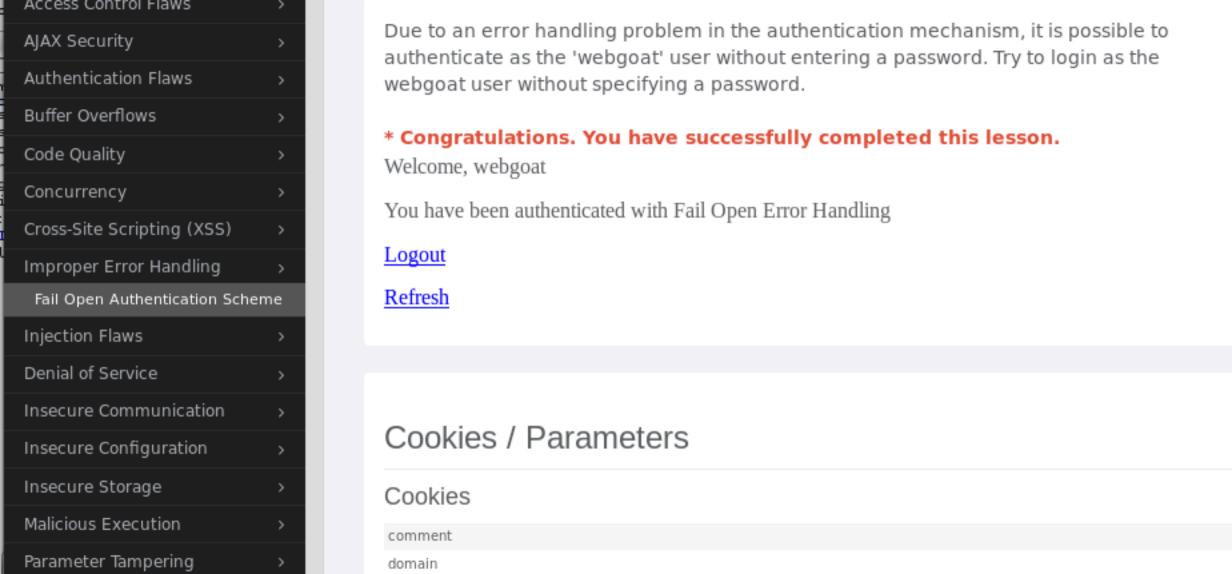
Stored XSS Attacks

Implement a fix to block this reflected XSS attack, Repeat step 5. Verify that the atta is no longer effective. * You have completed Stage 5: Reflected XSS.

- * Welcome to Stage 6: Block Reflected XSS







Burp Suite Free Edition v1.7.17 - Temporary Project Burp Intruder Repeater Window Help Decoder Extender Project options User options Alerts Sequencer Comparer Target Proxy Spider Scanner Intruder Repeater Intercept HTTP history WebSockets history Options Request to http://localhost:8080 [127.0.0.1] Forward Intercept is on Action Drop Headers Raw Params. Hex POST /WebGoat/attack?Screen=114&menu=1100 HTTP/1.1 Host: localhost:8080 User-Agent: Mozilla/5.0 (X11; Linux x86 64; rv:45.0) Gecko/20100101 Firefox/45.0 Accept: */* Accept-Language: en-US, en; q=0.5 Injection Flaws: Command Content-Type: application/x-www-form-urlencoded; charset=UTF-8 Injection: I tried few times. X-Requested-With: XMLHttpRequest but still failed to complete Referer: http://localhost:8080/WebGoat/start.mvc Content-Length: 45 this task, sorry for that. Here Cookie: JSESSIONID=D6DD7C64CFCCC1B1231CE582C468BCA6 was what I did. Connection: close HelpFile=AccessControlMatrix.help" & netstat -ant & ifconfig&SUBMIT=View

Access Control Flaws	>	
AJAX Security	>	Command injection attacks represent a serious threat to any parameter-driven site. methods behind an attack are easy to learn and the damage caused can range from
Authentication Flaws	>	considerable to complete system compromise. Despite these risks an incredible nur
Buffer Overflows	>	systems on the internet are susceptible to this form of attack.
Code Quality	>	Not only is it a threat easily instigated, it is also a threat that, with a little common-se and forethought, can be almost totally prevented. This lesson will show the student s
Concurrency	>	examples of parameter injection.
Cross-Site Scripting (XSS)	>	It is always good practice to sanitize all input data, especially data that will used in Os
Improper Error Handling	>	command, scripts, and database queries. Try to inject a command to the operating system.
Injection Flaws	>	You are currently viewing: AccessControlMatrix.help"
Command Injection	_	Select the lesson plan to view: AccessControlMatrix.help ▼ View
Numeric SQL Injection		
Log Spoofing		ExecResults for '[/bin/sh, -c, cat "/root/Downloads/.extract/webapps/WebGoat/lesspn
XPATH Injection		/en/AccessControlMatrix.html""]' Injection Flaws: Command
LAB: SQL Injection		Returncode: 2 Injection: I tried few times, but still failed to complete
Stage 1: String SQL Injection		Bad return code (expected 0) this task, sorry for that. Here
Stage 2: Parameterized Quer	y #1	was what I got.

	Log Spoofing		
	XPATH Injection		
	LAB: SQL Injection		
	Stage 1: String SQL Injection		
	Stage 2: Parameterized Query	#1	
	Stage 3: Numeric SQL Injection		
	Stage 4: Parameterized Query	#2	
	String SQL Injection		
	Modify Data with SQL Injection		
	Add Data with SQL Injection		
	Database Backdoors		
	Blind Numeric SQL Injection		
	Blind String SQL Injection		
[Denial of Service	>	
ı	nsecure Communication	>	
I	nsecure Configuration	>	

in all the weather data being displayed.

* Congratulations. You have successfully completed this lesson.

* Bet you can't do it again! This lesson has detected your successful attack and has now switched to a defensive mode. Try again to attack a parameterized query.

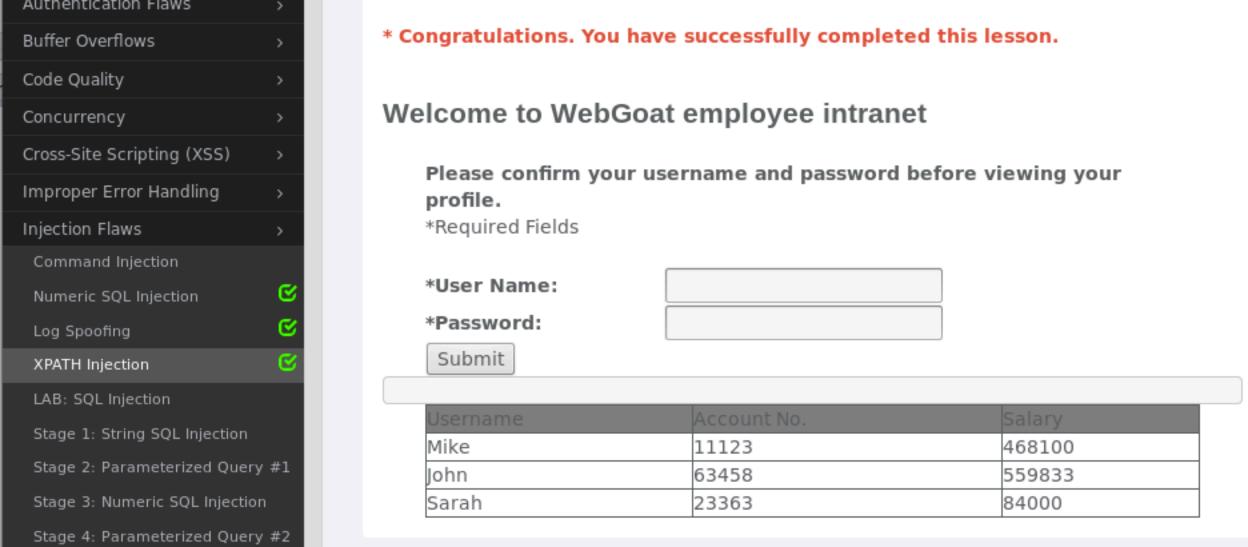
Select your local weather station: Columbia ▼

Go!

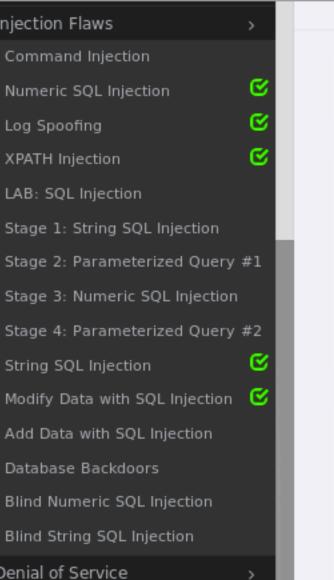
SELECT * FROM weather data WHERE station = 101 or 1=1

CTATION	NAME	CTATE	MAINI TEMAR	MAN TEMP
STATION	NAME	SIAIE	MIN_IEMP	MAX_TEMP
101	Columbia	MD	-10	102
102	Seattle	WA	-15	90
103	New York	NY	-10	110
104	Houston	TX	20	120
10001	Camp David	MD	-10	100
11001	Ice Station Zebra	NA	-60	30

AJAX Security	>	* The grey area below represents what is going to be logged in the web server's log fil * Your goal is to make it like a username "admin" has succeeded into logging in.
Authentication Flaws	>	* Elevate your attack by adding a script to the log file.
Buffer Overflows	>	
Code Quality	>	* Congratulations. You have successfully completed this lesson. User Name:
Concurrency	>	Password :
Cross-Site Scripting (XSS)	>	Login
Improper Error Handling	>	Logiii
Injection Flaws	>	
Command Injection		Login failed for username: Smith
Numeric SQL Injection	€	Login Succeeded for username: admin
Log Spoofing	€	
XPATH Injection		
LAB: SQL Injection		Cooking / Devemetors
Stage 1: String SQL Injection		Cookies / Parameters
Stage 2: Parameterized Que	ry #1	Cookies
Stage 3. Numeric SOL Injection	0.0	



miproper error ridiidiiiig	COIIIIIa	па, ѕспріѕ, апа	ruatabase querie	s, even ii the threat or	SQL IIIJed	.uon nas	been	
Injection Flaws	preven	ted in some oth	er manner.					
Command Injection	Genera	l Goal(s):						
Numeric SQL Injection	∑ The for	The form below allows a user to view their credit card numbers. Try to inject an SQL string						
Log Spoofing	that res	that results in all the credit card numbers being displayed. Try the user name of 'Smith'.						
XPATH Injection	* Cong	ratulations V	ou have success	fully completed thi	is lesson			
LAB: SQL Injection				erformed an SQL inj			ame typ	e
Stage 1: String SQL Injection	of atta	ck on a param	neterized query	. Restart the lesson	if you w	ish to re	eturn to	1
Stage 2: Parameterized Query #	1	ectable query						
Stage 3: Numeric SQL Injection	Enter yo	ur last name: Er	win' OR '1'='1	Go!				
Stage 4: Parameterized Query #	2 SELEC	T * FROM user	data WHERE last	_name = 'Erwin' OR	'1'='1'			
String SQL Injection	31110							
Modify Data with SQL Injection	USERII	FIRST_NAME	LAST_NAME	CC_NUMBER	CC_TYPE	COOKIE	LOGIN_C	COU
Add Data with SQL Injection	101	Joe	Snow	987654321	VISA		0	
	101	Joe	Snow	2234200065411	MC		0	
Oatabase Backdoors	102	John	Smith	2435600002222	MC		0	
Blind Numeric SQL Injection	102	John	Smith	4352209902222	AMEX		0	
Blind String SQL Injection	103	Jane	Plane	123456789	MC		0	
	103	lane	Plane	333498703333	AMEX		n	



Injection Flaws

Log Spoofing

XPATH Injection

LAB: SQL Injection

String SQL Injection

Database Backdoors

Denial of Service

Command Injection

[ava [Source] | Solution | Lesson Plan | Hints | Restart Lesson The form below allows a user to view salaries associated with a userid (from the table

* Congratulations. You have successfully completed this lesson.

Enter your userid: whatever'; UPDATE salaries ! Go! No results matched. Try Again.

named **salaries**). This form is vulnerable to String SQL Injection. In order to pass this

Cookies / Parameters

Cookies

comment

lesson, use SQL Injection to modify the salary for userid **ismith**.



Add Data with SQL Injection









Introduction

General Access Control Flaws

AJAX Security

Authentication Flaws

Buffer Overflows

Code Quality

Concurrency

Cross-Site Scripting (XSS) Improper Error Handling

Injection Flaws

Command Injection

[ava [Source] | Solution | Lesson Plan | Hints Restart Lesson

The form below allows a user to view salaries associated with a userid (from the table named **salaries**). This form is vulnerable to String SQL Injection. In order to pass this lesson, use SQL Injection to add a record to the table.

* Congratulations. You have successfully completed this lesson.

Enter your userid: whatever'; INSERT INTO salai Go! No results matched. Try Again.

Cookies / Parameters

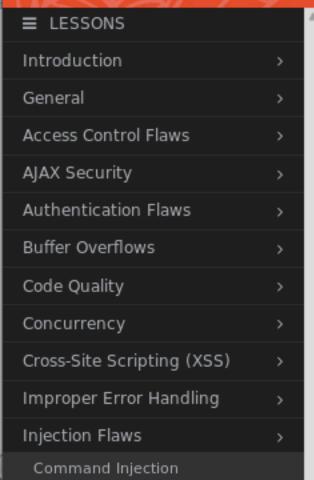


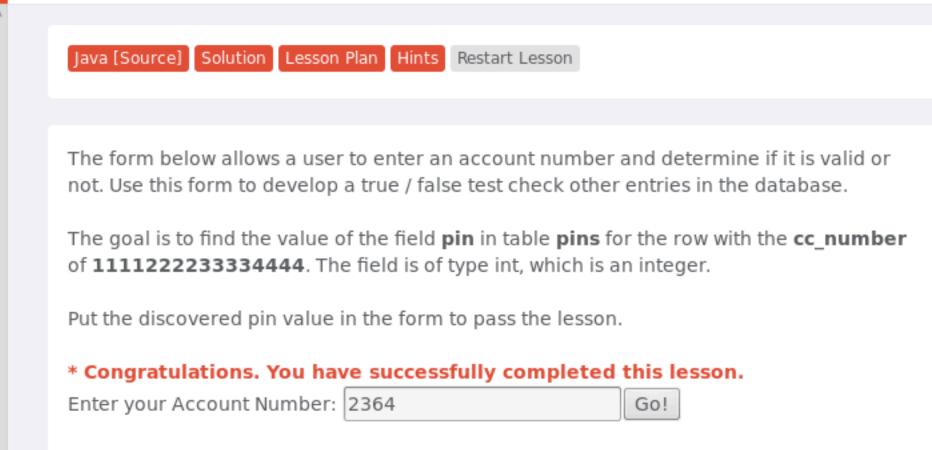
Blind Numeric SQL Injection













Blind String SQL Injection







```
■ LESSONS
Introduction
General
Access Control Flaws
AJAX Security
Authentication Flaws
Buffer Overflows
Code Quality
Concurrency
Cross-Site Scripting (XSS)
Improper Error Handling
Injection Flaws
```

Command Injection

```
Java [Source] Solution Lesson Plan Hints Restart Lesson
```

The form below allows a user to enter an account number and determine if it is valid or not. Use this form to develop a true / false test check other entries in the database.

Reference Ascii Values: 'A' = 65 'Z' = 90 'a' = 97 'z' = 122

The goal is to find the value of the field **name** in table **pins** for the row with the **cc_number** of **4321432143214321**. The field is of type varchar, which is a string.

Put the discovered name in the form to pass the lesson. Only the discovered name should be put into the form field, paying close attention to the spelling and capitalization.

* Congratulations. You have successfully completed this lesson.