Solutions

Exercises 2nd Semester

Exercise 3.1 (Authentication Bypass)

#	Username	Password	Created SQL Query	Query Result
1	`horst`	`n0Rd4kAD3m!E`	`SELECT id FROM users WHERE name = 'horst' AND password = 'n0Rd4kAD3m!E'`	`42`
2	`'`	`qwertz`	`SELECT id FROM users WHERE name = ''' AND password = 'qwertz'`	`Error`
3	`'--`	`abc123`	`SELECT id FROM users WHERE name = ''-- AND password = 'abc123'`	`null`

#	Username	Password	Created SQL Query	Query Result
4	`horst'--`	`qwertz`	`SELECT id FROM users WHERE name = 'horst'-- AND password = 'abc123'`	`42`
5	`admin'--`	`<anything>`	`SELECT id FROM users WHERE name = 'admin'`	`1`
6	`' OR 1=1--`	`<anything>`	`SELECT id FROM users`	`1`, `2`, …

Exercise 4.2 (Session ID Generator)

The IDs are short (15 chars), have low entropy (a-z, 0-9) and contain **predictable patterns** indicating at least partial non-randomness.

#	Session ID	#	Session ID
1	h5kek4z**9ha1**rtrf	7	po953ld**7hg2**awi9
2	gj75l3k**7hb1**5rtr	8	t6zhj2n**5hh2**7bn0
3	l8l65k4**5hc1**rw7i	9	iu345r5**3hi2**aw34
4	p05jrj5**3hd1**i039	10	o0z4341**1hj2**njkl
5	5urltda**1he1**bn46	11	9por42o**9hk3**dfrz
6	j5le97h**9hf2**yq3h	...	...

Exercise 8.2 (ArrayList Deserialization)

```
/**
 * The maximum size of array to allocate.
 * Some VMs reserve some header words in an array.
 * Attempts to allocate larger arrays may result in
 * OutOfMemoryError: Requested array size exceeds VM limit
 */
private static final int MAX_ARRAY_SIZE = Integer.MAX_VALUE - 8;
```

💥 *Whenever an* `OutOfMemoryError` *occurs, the affected JVM crashes.*

Exercise 7.3 (HashSet Deserialization)

```
i=0, root=[[], [foo]]
i=1, root=[[[], [foo]], [[], foo, [foo]]]
i=2, root=[[[[], [foo]], [[], foo, [foo]]], [[[], [foo]], foo, [[], foo, [foo]
i=3, root=[[[[[], [foo]], [[], foo, [foo]]], [[[], [foo]], foo, [[], foo, [foo
i=4, root=[[[[[[], [foo]], [[], foo, [foo]]], [[[], [foo]], foo, [[], foo, [fo
i=5, root=[[[[[[[], [foo]], [[], foo, [foo]]], [[[], [foo]], foo, [[], foo, [f
i=6, root=[[[[[[[[], [foo]], [[], foo, [foo]]], [[[], [foo]], foo, [[], foo, [
i=7, root=[[[[[[[[[], [foo]], [[], foo, [foo]]], [[[], [foo]], foo, [[], foo,
i=8, root=[[[[[[[[[[], [foo]], [[], foo, [foo]]], [[[], [foo]], foo, [[], foo,
```

💥 *With its members recursively linked to each other, when deserializing* `root`*, the JVM will begin creating a recursive object graph. It will never complete, and consume CPU indefinitely.*

Exercise 9.1 (Protection Req. Calc.)

Aspect / Application	Website	VCS	Webshop	B2B API
Business criticality	2	1	5	2
Information classification	0	2	2	2
Compliance requirements	0	0	2	1
Exposure to threats	5	1	5	5
Authentication mechanism	0	-2	-1	-1
Total Score	7	2	13	9
Rating	Medium	Low	High	Medium