UNIVERSITY COMPUTER SCIENCE • EXERCISE SHEET

7.6 Protection and security

Name: _____ Class: _____ Date: _____ **Total: 24 marks**

KEY WORDS kernel mode • system call

Objective

Reason about the privilege boundary and its crossing costs.

1 Key knowledge and worked examples

■ The wall

User mode runs application code; kernel mode touches devices and page tables; the mode bit enforces the rings, and syscalls are the only doors ✓. File permissions: 9 bits (rwx for owner, group, other) ✓.

■ Worked example: the crossing

One syscall = two crossings of about 100 ns: 10^6 syscalls cost 0.2 s – the reason libraries buffer and batch ✓.

2 Practice

2.1 Name the instruction that crosses user to kernel, and the mechanism that picks the handler. [3]

2.2 Give the 9 permission bits as a triple of triples for rw-r——. [3]

2.3 Why is passing an open descriptor safer than passing a file name? [3]

3 Exam-style questions

3.1 A program issues 5000 syscalls; each crossing costs 120 ns.

Compute the total crossing overhead. Compute how many syscalls a buffering library must batch into one to cut the overhead by a factor of 100. State what the kernel must validate at every crossing regardless of batching, naming two checks. [8]

3.2 A printer daemon runs with privilege; a user asks it to print /salary.db.

Explain the confused-deputy problem in this scenario, naming whose authority opens the file. State the capability fix and why the daemon can then refuse safely. Give the general principle the scenario illustrates, and one way the deputy could still misbehave even with capabilities. [7]

Solutions

2.1

- The trap instruction; a fixed vector table picks the handler ✓.

2.2

- Owner rw-, group r–, other —: (110)(100)(000) ✓.

2.3

- The descriptor is authority the caller already earned – the daemon cannot widen it; a name asks the deputy to use its own greater authority ✓.

3.1

- Total: $5000 \times 2 \times 120$ ns $= 1.2$ ms ✓.
- Batch factor 100: one syscall per 100 calls – 50 syscalls at 240 ns each $= 12$ µs ✓.
- Validate always: the caller's identity and permissions, and every argument – pointers and lengths – at the boundary ✓.

3.2

- The daemon opens /salary.db with ITS OWN privilege, not the user's – the deputy is confused about whose authority to use ✓.
- Fix: the user passes an open descriptor (capability); the daemon prints only what it was handed and can refuse names outright ✓.
- Principle: least privilege – and validate every crossing ✓.
- Even with capabilities, a buggy deputy can print the content to the wrong place or leak it – authority is bounded, behaviour is not ✓.