Homework 3

1. What should the architecture of a network application under Unix be that processes potentially hostile data?
2. How can you exploit the fact that every night root has a cron job that deletes the files in `/tmp`? (Hint: cron-attack)
3. How does a buffer-overflow attack work? (Hint: What happens on the stack.)
4. Why is it crucial for a buffer overflow attack that the stack grows from higher addresses to lower ones?
5. If the attacker uses a buffer overflow attack in order to inject code, why can this code not contain any zero bytes?
6. How does a stack canary help with preventing a buffer-overflow attack?
7. Why does randomising the address where programs are run help defending against buffer overflow attacks?
8. Assume format string attacks allow you to read out the stack. What can you do with this information? (Hint: Consider what is stored in the stack.)
9. Assume you can crash a program remotely. Why is this a problem?
10. How can the choice of a programming language help with buffer overflow attacks? (Hint: Why are C-programs prone to such attacks, but not Java programs.)