Secure Programming Techniques — exam 01.06.2018

1. If the attacker of stack-based buffer overflow works around data execution protection (NX/DEP) with return-oriented programming, what can we do to stop the attack?

2. How would you break a sloppy OAuth+OIDC single sign-on implementation?

3. Find all potential vulnerabilities in this C function:

```
int fd;

int save(char *question) {
    char filename[100];
    struct stat *statbuf;

    printf(question);
    scanf("%s", filename);
    if (strlen(filename) > 100) {
        return 0;
    }
    /* exit if file exists */
    if (stat(filename, statbuf) < 0) {
        return 0;
    }
    fd = open(filename, O_WRONLY | O_CREAT, 0666);
    save_to_file(fd);
    close(fd);
    return 1;
}
```

4. Find all potential vulnerabilities in this Perl function:

```
sub store($globpattern, $thedata) {
 open($fnames, "ls -t ".$globpattern . " | head -n 1 |") ||
   die "Can not glob";
 $filename = <$fnames>;
 close($fnames);
 chomp($filename);
 if (-l $filename) {
   die "Not writing to symlink!"
 }
 open ($fh, ">", $filename) || die "Can not open file";
 print $fh $thedata;
 close $fh;
}
```

5. What possible vulnerabilities could the following URL exploit? Try to find as many as possible:

   ```
   http://example.com/index.php/functions.html?session=987654321&
   action=‘cat /etc/passwd‘&user=joe’%20union%20select%20’x&
   prompt=%3C%49%6DG%2B%53%72c%3D
   %22J%61%76a%53C%72i%70t%3Aa%6Ce%72t%28%27%21%27%29%3B%22%3E
   ```

Send response by e-mail to mroos@ut.ee no later than 14.00. You can answer either in English or in Estonian.