

PA193 Secure coding principles and practices



Overview of the subject

Petr Švenda, Petr Ročkai, Marek Sýs, Kamil Dudka,
Mirek Jaroš, Martin Ukrop



PA193 Secure coding principles and practices

- Secure coding
 - How to write code in a more secure way
 - So that the program is harder to be attacked/exploited
 - Selected basic building blocks of security applications
- *2/2/2*
 - Lecture: 2 hours weekly
 - Seminar: 2 hours weekly (2 seminar groups)
 - Homework: about 6-8 hours/each (6+1 bonus)
 - Project: about 30-40 hours/person

People

- Main contact: Petr Švenda (CRoCS@FI MU)
 - Office hours: Monday 13:00-13:50, A406
 - svenda@fi.muni.cz, @rngsec
 - <https://crocs.fi.muni.cz/people/svenda>
- Petr Ročkai (FI MU)
 - Office hours: Tuesday 12:00-12:50, A406
 - xrockai@fi.muni.cz
- Other lectures and seminars
 - Marek Sýs (FI), Kamil Dudka (RedHat), Mirek Jaroš (RedHat), Martin Ukrop (FI)

Previous knowledge requirements

- Basic knowledge of (applied) cryptography and IT security
 - symmetric vs. asymmetric cryptography, PKI
 - block vs. stream ciphers and usage modes
 - hash functions
 - random vs. pseudorandom numbers
 - basic cryptographic algorithms (AES, DES, RSA, EC, DH)
 - risk analysis
- Basic knowledge in formal languages and compilers
- User-level experience with Windows and Linux OS
- **Practical experience with C/C++/Java language**

Organization

- Lectures + seminars + assignments + project + exam
- Assignments
 - 6 homework assignments (+ 1 bonus)
 - **Individual work of each student**
 - Lab A403 available to students (except teaching hours)
- Project
 - **Team work** (2-3 members)
 - Details at seminars, parser for a cryptocurrency blockchain
- Exam
 - Written exam, open questions, pencil-only

Grading

- Credits
 - 2+2+2 credits, plus 2 for the final exams
- Points [**Notice minimal number of points required!**]
 - Homework (30) – [minimum 15 required]
 - Project (20) – [minimum 10 required]
 - Written exam (50) – [no minimum limit]
 - Occasional bonuses 😊
- Grading
 - A \geq 90% of maximum number of points 100 (max)
 - B \geq 80% of maximum number of points 90
 - C \geq 70% of maximum number of points 80
 - D \geq 60% of maximum number of points 70
 - E \geq 50% of maximum number of points 60
 - F $<$ 50% of maximum number of points 50

Attendance

- Lectures
 - Attendance not obligatory, but highly recommended
- Seminars
 - Attendance **obligatory**
 - Absences must be excused at the department of study affairs
 - 2 absences are OK (even without excuse)
- Assignments and projects
 - Done during students free time (e.g. at the dormitory)
 - Access to network lab and CROCS lab is possible

Discussion forum in Information System

- Discussion forum in Information System (IS)
 - <https://is.muni.cz/auth/cd/1433/podzim2017/PA193/>
- Mainly for discussion among the students
 - Not observed by staff all the time!
 - Write us email if necessary
- What to ask?
 - OK to ask about ambiguities in assignment
 - NOT OK to ask for the solution
 - NOT OK to post your own code and ask what is wrong



Plagiarism

- Homeworks
 - Must be worked out independently by each student
- Projects
 - Must be worked out by a team of 3 students
 - Every team member must show his/her contribution
- Plagiarism, cut&paste, etc. is not tolerated
 - Plagiarism is use of somebody else words/programs or ideas without proper citation
 - Automatic tools used to recognize plagiarism
 - If plagiarism is detected student is assigned -7 points
 - More serious cases handled by the Disciplinary committee

Reuse of existing code

- Code reuse is generally great thing, but..
- NOT in homework or assignments!
- It is **NOTOK**:
 - Take any code from web when you should create code completely on your own (project - parser)
 - Share code of your solution with others (homework)

```

#include "LDSSecurityObject.h"
#include <dirent.h>
#include <openssl/sha.h>
int main(void)
{
    LDSSecurityObject_t *lds;
    lds = (LDSSecurityObject_t*)calloc(1, sizeof *lds);
    DIR *dir;
    FILE *fp;
    char dirname[100],dirname1[100];
    char filenames[100][100];
    char correctnames[100][100];
    int countfiles = 0;
    int count,j,flag=0;
    int foundindex;
    struct dirent *ent;
    if(!lds) exit(1);

    FILE *f=fopen("Sample-data/lds.bin","rb");
    if(!f) exit(1);
    unsigned char buffer[10000];
    int buflen,size;
    char *input;
    unsigned char *hashvalue;
    buflen=fread(buffer,1,10000,f);
    fclose(f);

    printf("Input the name of directory (example Sample-data)");
    scanf("%s",dirname);

    strcpy(dirname1,dirname);
    if ((dir = opendir (dirname)) != NULL)
    {
        while ((ent = readdir (dir)) != NULL)
        {
            strcpy(filenames[countfiles],ent->d_name);
            //printf ("%s\n", ent->d_name);
            //printf ("%s\n", filenames[countfiles]);
            countfiles++;
        }
        closedir (dir);
    }
    else
    {
        /* could not open directory */
        perror ("");
    }
}

```

```

#include "LDSSecurityObject.h"
#include <dirent.h>
#include <openssl/sha.h>
int main(void)
{
    LDSSecurityObject_t *lds;
    lds = (LDSSecurityObject_t*)calloc(1, sizeof *lds);
    DIR *dir;
    FILE *fp;
    char Directory[100],Directory1[100];
    char in_file_name[100][100];
    char corrcct_names[17][100];
    int no_of_files =0,i;
    int cnt,j,cmp,flag=0;

    struct dirent *ent;
    if(!lds) exit(1);

    FILE *f=fopen("Sample-data/lds.bin","rb");
    if(!f) exit(1);
    unsigned char buffer[10000];
    int buflen,size;
    char *input;
    unsigned char *hashvalue;
    buflen=fread(buffer,1,10000,f);
    fclose(f);

    printf("Enter the name of directory whose files to be veified :");
    scanf("%s",Directory);

    strcpy(Directory1,Directory);
    if ((dir = opendir (Directory)) != NULL)
    {
        while ((ent = readdir (dir)) != NULL)
        {
            strcpy(in_file_name[no_of_files],ent->d_name);
            no_of_files++;
        }
        closedir(dir);
    }
    else
    {
        /*Directory opening error*/
        perror ("");
    }
}

```

```
12/11/2015 11:27:15 4,135 bytes C, C++, C#, ObjC Source ANSI PC
int bitrates[] = {
    BITRATEFREE, BITRATEFREE, BITRATEFREE, BITRATEFREE, BITRATEFREE,
    32, 32, 32, 32, 8,
    64, 48, 40, 48, 16,
    96, 56, 48, 56, 24,
    128, 64, 56, 64, 32,
    160, 80, 64, 80, 40,
    192, 96, 80, 96, 48,
    224, 112, 96, 112, 56,
    256, 128, 112, 128, 64,
    288, 160, 128, 144, 80,
    320, 192, 160, 160, 96,
    352, 224, 192, 176, 112,
    384, 256, 224, 192, 128,
    416, 320, 256, 224, 144,
    448, 384, 320, 256, 160,
    BITRATEBAD, BITRATEBAD, BITRATEBAD, BITRATEBAD, BITRATEBAD,
    BITRATEBAD, BITRATEBAD, BITRATEBAD, BITRATEBAD, BITRATEBAD,
};

typedef struct{
```

```
18/11/2015 16:40:53 11,086 bytes C, C++, C#, ObjC Source ANSI UNIX
int bitrates[] = {
    BITRATEFREE, BITRATEFREE, BITRATEFREE, BITRATEFREE, BITRATEFREE,
    32, 32, 32, 32, 8,
    64, 48, 40, 48, 16,
    96, 56, 48, 56, 24,
    128, 64, 56, 64, 32,
    160, 80, 64, 80, 40,
    192, 96, 80, 96, 48,
    224, 112, 96, 112, 56,
    256, 128, 112, 128, 64,
    288, 160, 128, 144, 80,
    320, 192, 160, 160, 96,
    352, 224, 192, 176, 112,
    384, 256, 224, 192, 128,
    416, 320, 256, 224, 144,
    448, 384, 320, 256, 160,
    BITRATEBAD, BITRATEBAD, BITRATEBAD, BITRATEBAD, BITRATEBAD,
};

typedef struct{
    /// unsigned framesync :12; //Frame synchronizer
    /// unsigned int id3v2_start; //Start of id3v2 tag
```

Example of Plagiarism

```
12/11/2015 11:27:15 4,135 bytes C, C++, C#, ObjC Source ANSI PC
int readMP3header(FILE *f, MP3HEADER *h){
    MP3ID3TAG2 tag;

    //push file point to the beginning
    rewind(f);
    fread(&tag, 1, sizeof(MP3ID3TAG2), f);

    //tag id3v2 are located at the beginning of file, id3v1 at the end
    if(tag.tagid[0]=='I' && tag.tagid[1]=='D' && tag.tagid[2]=='3'){//is id3v2 present
        fseek(f, unpacktagsize(tag), SEEK_CUR);
    }else{//isn't tag id3v2 - go back
        rewind(f);
    }

    //I'm currently not interested in the final state of the file pointer
```

```
18/11/2015 16:40:53 11,086 bytes C, C++, C#, ObjC Source ANSI UNIX
int ReadMP3Header(FILE *f, MP3HEADER *h, unsigned int StartFlag, uint16_t framesync, unsigned int id3v2_start);
unsigned int unpacktaghead[4];
int id3v2_start;
MP3ID3TAG2 tag;
int lc = 0;

if ( StartFlag == 1 )
{
    rewind(f); //set file pointer to beginning of file
    fread(&tag, 1, sizeof(MP3ID3TAG2), f);

    // Check for the tag id3v2 is present at the beginning of file,
    if(tag.tagid[0]=='I' && tag.tagid[1]=='D' && tag.tagid[2]=='3')
    { //if tag id3v2 is present then jump to end of tag
        fseek(f, unpacktagsize(tag), SEEK_CUR);

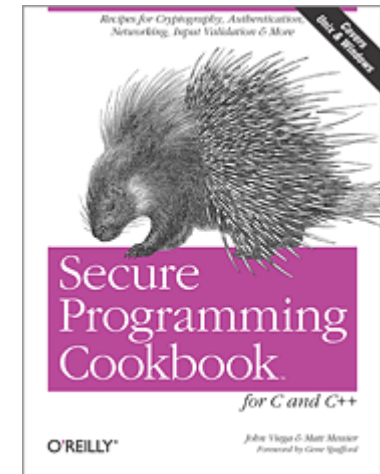
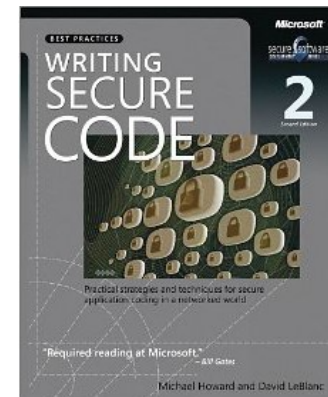
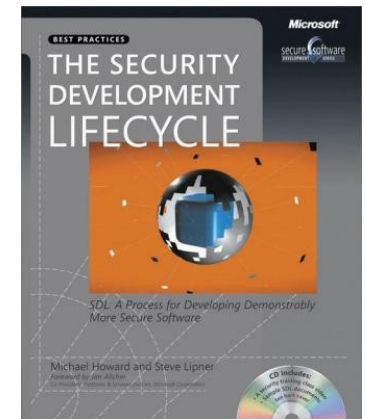
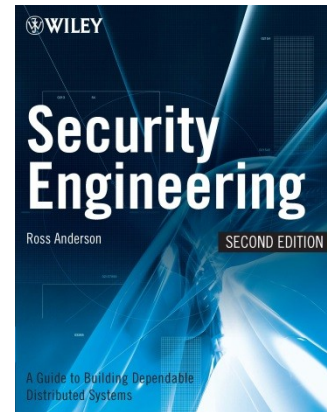
        printf("\nFile Has Id3Tag2 Present At Begining");
    }
    else{ //if tag idv3 isn't present then go back to beginning of file
        rewind(f);
    }
}
```

Course resources

- Lectures (PDF) available in IS
 - IS = Information System of the Masaryk University
 - <https://is.muni.cz/auth/el/1433/podzim2017/PA193/>
- Homeworks/assignments available in IS
 - Submissions also done via IS (Homework vaults)
- Additional tutorials/papers/materials from time to time will also be provided in IS
 - To better understand the issues discussed
- Recommended literatures
 - To learn more ...

Recommended literature

- Ross Anderson - Security engineering, Wiley
- Michael Howard, Steve Lipner - Secure Development Lifecycle, MS Press
- John Viega, Matt Messier - Secure programming cookbook, O'Reilly
- Michael Howard - Writing secure code, MS Press



Questions ?

