TAKR - zadání projekt II

Průběh projektu II:

- 5 týden – Zadání projektu
- 9 týden – Kontrola studie
- 12 týden – Odevzdání projetu (do 26.4 2017)
- 13 týden – Obhajoba projektu

Bodové hodnocení projektu II:

- Studie problematiky – 4b
- Realizace programu – 7b
- Obhajoba projektu – 4b

Do 9. týdne semestru je povinnost odevzdat studii problematiky, která bude obsahovat:

- úvod a popis projektu,
- cíle projektu, kterých by mělo být dosaženo,
- teoretickou část popisující vybrané téma,
- aktuální stav řešení (v případě nefunkčnosti okomentovat možné příčiny a další kroky; v případě funkčnosti uvést možné rozšíření)
- představit autory a uvést jejich pracovní přínos do projektu.

Dokument musí obsahovat náležitosti technické zprávy (titulní strana, obsah, úvod, závěr, přílohy), rozsah je stanoven na 4 – 5 stran (dle složitosti problematiky). Inspirovat se můžete některým z odborných článků dostupných na internetu. Doporučená je konzultace s vyučujícím, aby se váš projekt nebral špatným směrem.

Níže jsou uvedeny jednotlivé projekty, které budou zadány skupinám dle registrace v online dokumentu.
https://docs.google.com/spreadsheets/d/1eDVPhqQnOLKJOT09_PGxTEOqoxCmn5xQ5gU95bPJO9A/edit#gid=0

Odevzdání projektu bude probíhat v elearningu v archivovaném souboru ve formátu zip s číslem skupiny. Odevzdání provede jen jeden člen ze skupiny.

Lámání Caesarovy šifry

1. Vytvořte program, který bude schopen prolomit Caesarovu šifru. První část programu bude sloužit k šifrování textové zprávy. Další část programu bude prolamovat zašifrovaný text bez znalosti původního textu a posunu.
    - Vstupem programu bude text z konzole nebo textového souboru a počet míst posunu.
    - Každá dílčí činnost programu bude vypsána do konzole. (načtený text, zašifrovaný text, výstupní dešifrovaný text)
2. Navrhněte metodu pro zlepšení Caesarovy šifry
    - Navrhněte metodu, která zvýší zabezpečení Caesarovy šifry.
    - Aplikujte navrženou metodu na programu a demonstrujte na procesu lámání.

V tomto zadání projektu není možné použít knihovny zaměřené na šifrování a lámání Caesarovy šifry!!

[1] Caesarova šifra (https://cs.wikipedia.org/wiki/Caesarova_%C5%A1ifra)
[2] Caesarova šifra (http://www.matematika.cz/caesarova-sifra)

Lámání Vigenérovy šifry

1. Vytvořte program, který bude schopen prolomit Vigenérovu šifru. První část programu bude sloužit k šifrování textové zprávy. Další část programu bude prolamovat zašifrovaný text bez znalosti původního textu a hesla.
    - Vstupem programu bude text z konzole nebo textového souboru a zvolené heslo.
    - Každá dílčí činnost programu bude vypsána do konzole. (načtený text, zašifrovaný text, výstupní dešifrovaný text)
2. Zjistěte, jak dlouhé heslo je možné prolomit do max. 1 hodiny.

V tomto zadání projektu není možné použít knihovny zaměřené na šifrování a lámání Vigenérovy šifry!!

[1] Vigenèrova šifra (https://cs.wikipedia.org/wiki/Vigen%C3%A8rova_%C5%A1ifra)
[2] Vigenèrova šifra (http://www.matematika.cz/vigenerova-sifra)

Lámání transpoziční šifry

1. Vytvořte program, který bude schopen prolomit transpoziční šifru. První část programu bude sloužit k šifrování textové zprávy. Další část programu bude prolamovat zašifrovaný text bez znalosti původního textu a metody transpozice. **Využijte minimálně tři metody transpozice textu.**
    - Vstupem programu bude text z konzole nebo textového souboru a metoda transpozice.

- Každá dílčí činnost programu bude vypsána do konzole. (načtený text, zašifrovaný text, výstupní dešifrovaný text)

2. Porovnejte zvolené metody transpozice a zjistěte nejbezpečnější.

V tomto zadání projektu není možné použít knihovny zaměřené na šifrování a lámání Vigenérovy šifry!!

[1] Transpoziční šifra (https://cs.wikipedia.org/wiki/Transpozi%C4%8Dn%C3%AD_%C5%A1ifra)

Šifrovaná komunikace mezi dvěma uživateli

Vytvořte program, který bude schopen provádět šifrovanou komunikaci mezi dvěma uživateli. Z právy budou zašifrované pomocí symetrické kryptografie a pro výměnu klíče u symetrické kryptografii využijte asymetrickou kryptografii. Veřejné klíče uživatelů budou opatřeny certifikáty, které se budou ověřovat u certifikační autority. Každá dílčí činnost programu bude vypsána do konzole.

Průběh komunikace:

1. Vytvoření klíčů asymetrické kryptografie pro certifikační autoritu a obě strany.
2. Podepsání veřejných klíčů.
3. Ověření klíčů u certifikační autority.
4. Stanovení klíče pro symetrickou komunikaci a přenos šifrovaného textu.
5. Dešifrování a zobrazení souboru.

V tomto zadání je možné použít knihovny zaměřené na symetrickou kryptografii, asymetrickou kryptografii a certifikáty!!!

[1] RSA (https://cs.wikipedia.org/wiki/RSA)

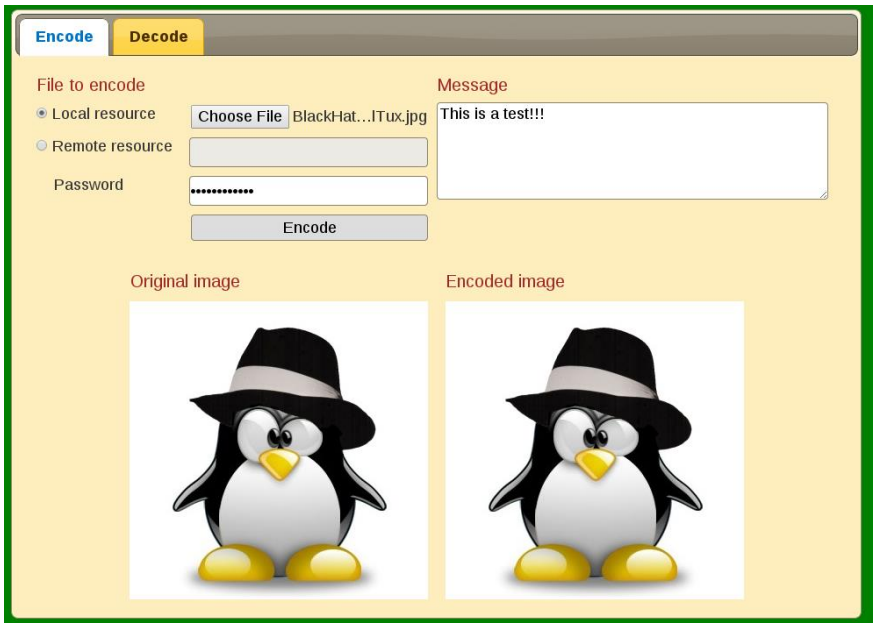
[2] AES (http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf)

[3] Digitální certifikát (https://cs.wikipedia.org/wiki/Digit%C3%A1ln%C3%AD_certifik%C3%A1t)

Steganografie

Pomocí steganografie ukryjte tajná data do obrázku. Vytvořte program, který bude schopen pomocí dané metody ukrýt vložený text do libovolného obrázku. Zpočátku je třeba nastudovat problematiku týkající se steganografie. Následně se zaměřit na práci s obrázky v jazyce python a vhodným algoritmem zakomponovat do daných bitů snímku tajný text.

Pro lepší pochopení je zde uveden ilustrační obrázek:



Je třeba postupovat dle pravidel:

- definovat maximální velikost textu, který jste schopni skrýt,
- nastavit si limity, kdy je upravený obrázek na první pohled shodný s originálem (je vhodné demonstrovat zlom, kdy dojde k degradaci snímku),
- zda vytvoříte GUI rozhraní nebo čistě konzolovou aplikaci.

V tomto zadání je možné použít knihovny zaměřené na práci s obrázky, je však zakázáno využití knihoven určených přímo pro steganografii.

Lámání heše SHA1

Nejde o samotné prolomení heše, ale o zjištění hesla hrubou silou v porovnání s vytvořeným hešem.

Jako první úkol bude vytvoření heše pomocí hešovací funkce SHA1. Tento heš bude sloužit k ověření funkčnosti.

Dále vytvořte program na lámání hesla hrubou silou. Dle složitosti začínejte na krátkých vstupech, postupně složitost zvyšujte. Program bude generovat znaky a skládat z nich slova. Ty následně bude hešovat a získaný heš bude porovnávat s testovaným hešem vloženým na vstupu (vytvoření v prvním kroku). Jestliže se heš bude shodovat, heslo bylo prolomeno a zobrazí se v aplikaci.

Heslo bude libovolně dlouhé dle složitosti výpočtu a doby výpočtu (alespoň 6 znaků), a bude se skládat z:

- malé anglické abecedy,
- velké anglické abecedy,
- číslic,
- speciálních znaků.

Doporučená je práce ve vláknech (paralelní zpracování výpočtu). Nesmí se používat slovníkový útok.

V tomto zadání je možné použít knihovny zaměřené na tvorbu heše. Nesmí se však používat další knihovny a balíčky spojené s lámáním hesla. Návrh algoritmu je samostatná práce. Optimalizace výpočtu je doporučena! Práce ve více vláknech je přednostně hodnocena.

[1] SHA-1 (https://en.wikipedia.org/wiki/SHA-1)
[2] SHA-1 (http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf)

Lámání heše MD5

Nejde o samotné prolomení heše, ale o zjištění hesla hrubou silou v porovnání s vytvořeným hešem.

Jako první úkol bude vytvoření heše pomocí hešovací funkce MD5. Tento heš bude sloužit k ověření funkčnosti.

Dále vytvořte program na lámání hesla hrubou silou. Dle složitosti začínejte na krátkých vstupech, postupně složitost zvyšujte. Program bude generovat znaky a skládat z nich slova. Ty následně bude hešovat a získaný heš bude porovnávat s testovaným hešem vloženým na vstupu (vytvoření v prvním kroku). Jestliže se heš bude shodovat, heslo bylo prolomeno a zobrazí se v aplikaci.

Heslo bude libovolně dlouhé dle složitosti výpočtu a doby výpočtu (alespoň 6 znaků), a bude se skládat z:

- malé anglické abecedy,
- velké anglické abecedy,
- číslic,
- speciálních znaků.

Doporučená je práce ve vláknech (paralelní zpracování výpočtu). Nesmí se používat slovníkový útok.

V tomto zadání je možné použít knihovny zaměřené na tvorbu heše. Nesmí se však používat další knihovny a balíčky spojené s lámáním hesla. Návrh algoritmu je samostatná práce. Optimalizace výpočtu je doporučena! Práce ve více vláknech je přednostně hodnocena.

[1] MD5 (https://cs.wikipedia.org/wiki/Message-Digest_algorithm)

Tvorba šifrátoru DES

Úkolem je vytvořit program, který bude šifrovat a dešifrovat vstupní data pomocí algoritmu DES. V první řadě je potřeba nastudovat problematiku šifrování a pochopit funkci algoritmu. Dále je nutné v jazyce python naprogramovat jednotlivé bloky a transformace potřebné pro správnou funkci algoritmu. Vstupem programu bude text nebo textový soubor, který bude zašifrovaný algoritmem DES po zadání klíče uživatelem. Šifrový text bude uložen do paměti a následně po volbě uživatele jej bude možné dešifrovat do původního tvaru. V další části projektu se zaměřte na modifikovanou variantu 3DES.

Je zakázáno využít existující knihovny pro podporu DES! Studenti budou pracovat s jednotlivými vstupními bity.

[1] DES (http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf)

Tvorba šifrátoru AES

Úkolem je vytvořit program, který bude šifrovat a dešifrovat vstupní data pomocí algoritmu AES. V první řadě je potřeba nastudovat problematiku šifrování a pochopit funkci algoritmu. Dále je nutné v jazyce python naprogramovat jednotlivé bloky a transformace potřebné pro správnou funkci algoritmu. Vstupem programu bude text nebo textový soubor, který bude zašifrovaný algoritmem AES po zadání klíče uživatelem. Šifrový text bude uložen do paměti a následně po volbě uživatele jej bude možné dešifrovat do původního tvaru v případě shody se správným klíčem.

Je zakázáno využít existující knihovny pro podporu AES! Studenti budou pracovat s jednotlivými vstupními bity, nad kterými budou provádět potřebné transformace. Výjimku může tvořit využití existujícího návrhu transformace SubBytes.

[1] AES (http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf)
[2] AES (https://cs.wikipedia.org/wiki/Advanced_Encryption_Standard)