BMP_HIDE

Arttu Kilpinen

4. marraskuuta 2013

1 Yleistä

Tieteellinen laskenta II -kurssilla sai kotikokeen valita useasta aiheesta. Työ, jota tämä dokumentti käsittelee oli "viestin piilotus bmp kuvaan". Tehtävänanto oli seuraava:

Ohjelmoi ohjelma, jolle annetaan komentoriviparametrinä bmp (bitmap) muodossa olevan kuvan nimi. Ohjelma kysyy käyttäjältä halutaanko kuvaan piilotta jokin viesti vai puretaan kuvaan jo piilotettu viesti. Jos käyttäjä haluaa piilottaa kuvaan viestin kysytään tämä viesti seuraavaksi käyttäjältä, jonka jälkeen kyseinen viesti tietenkin piilotetaan kuvaan. Jos taas käyttäjä haluaa purkaa kuvassa olevan viestin se puretaan ja kirjoitetaan ruudulle.

2 Tekniset tiedot

2.1 Yleistä

Ohjelma bmp_hide toteuttaa edellämainitun tehtävänannon. Toteutus on tehty niin, että viesti jonka käyttäjä haluaa piilottaa kirjoitetaan kuvatiedostossa olevien pikselien vähiten merkitseviin bitteihin (least significant bits). Kuvatiedoston header osaan ei siis tehdä muutoksia. Koska yksi tavu kirjoitetaan kahdeksan tavun vähiten merkitseviin bitteihin, tarvitsee kuvatiedoston pikselien datan olla 8 kertaa suurempi kuin sinne salattavan viestin. Viestin lopuksi kuvaan tallennetaan vielä nollatavu. Tämä ilmaisee viestin loppumista, jolloin viestin pituutta ei tarvitse tallentaa kuvaan. Viestiä siis luetaan niin kauan, kunnes luettu merkki on nolla. Ohjelmassa on käytetty vain c-kielen standardikirjastoon kuuluvia funktioita, sekä yhtä funktiota (getop/unistd.h) c posix kirjastosta

2.2 Tiedostorakenne

Ohjelma koostuu kuudesta c-kielisestä tiedostosta: main.c, options.c, bitwise.c, files.c, functions.c ja bmp_hide.h Ohjelmakoodi on jäsennelty tiedostoihin, niin että tietyssä tiedostossa on tiettyyn aihealueeseen liittyvät funktiot tai määritelmät.

2.2.1 main.c

Tiedosto on pyritty pitämään mahdollisimman lyhyenä ja yksinkertaisena. Se sisältää vain funktion main, joka vastaa ohjelman suorituksesta, kutsumalla muissa tiedostoissa olevia funktioita. Main-funktio määrittelee asetusten lukemisen interaktiivisessa tilassa kutsumalla funktiota interactive_options(), jos ohjelmaa käynnistettäessä ei asetettu yhtään komentoriviparametria. Jos parametreja asetettiin, lähettää main- funktio ne edelleen funktiolle check_options(), joka palauttaa, kuten interactive_options:kin, mainiin osoittimen bmp_hide.h tiedostossa määriteltyyn asetus-tietueeseen. Tämän jälkeen tehdään asetuksille muutamia virheentarkistuksia. Jos virheitä ei ilmene, kutsutaan funktiota read_message() tai write_message(), joissa ohjelman suoritus etenee loppuun. Onnistuneen suorituksen jälkeen main() palauttaa arvon 0. Muuten palautetaan virhekoodi 1-5. (Näistä lisää kohdassa paluuarvojen analysointi.)

2.2.2 options.c

Tiedosto sisältää tietueeseen OPTIONS (määritelty tiedostossa bmp_hide.h) liittyvät funktiot: check_options(), check_options_integrity(), free_options(), reset_options() ja interactive_options(). Funktiot interactive_options() ja check_options() määrittelevät asetustietueen arvot. Interactive_options() määrittelee asetukset interaktiivisesti käyttäjän kanssa ja check_options() sille annettujen parametrien mukaan. Onnistuneen suorituksen jälkeen, molemmat palauttavat osoitteen asetustietueeseen (muistinvaraus tapahtuu näissä funktioissa). Virhetilanteessa palautetaan NULL. Vain toista edellä mainituista funktioista kutsutaan ohjelman yhden suorituskerran aikana. Funktio check_options_integrity() tutkii ovatko sille annetut asetukset käyttökelpoiset. Jos asetuksissa ei ole ongelmia, palautetaan 0, muutoin virhekoodi 1-4. Funktio free_options() sulkee asetustietueessa määritellyt tiedostot sekä vapauttaa dynaamisesti varatun muistin. Reset_options() puolestaan asettaa alkuarvot sille annettuun asetustietueeseen.

2.2.3 bitwise.c

Tiedostossa bitwise.c on kaikki bittioperaatioita käyttävät funktiot. Lukufunktioita on 4 kappaletta: read_lsb(), read_byte(), read_char() ja read_word(). Funktio read_lsb() palauttaa parametrina saamansa tavun viimeisen eli vähiten merkitsevän bitin tavuna, jonka arvo on 0 tai 1. Read_byte() palauttaa parametrinaan saamasta tiedosto-osoittimesta seuraavan tavun. Read_char() kokoaa sille välitetyn tiedosto-osoittimen kahdeksasta seuraavasta tavusta yhden tavun ja palauttaa sen char tyyppisenä. Tavu kootaan kutsumalla read_lsb() -funktiota. Funktio read_word() lukee 32 bittisen little-endian-koodatun sanan parametrinaan saamasta tiedosto-osoittimesta ja palauttaa sen. Kirjoitusfunktioita on kaksi: write_char() ja write_bit(). Ensimmäinen kirjoittaa tiedosto-osoittimen määrittämään kohtaan write_bit() -funktiota apuna käyttäen yhden tavun. Tavu koodataan kahdeksaan tavuun, niin että jokaisen vähiten merkitsevää bittiä muutetaan. Yhden tavun kirjoitukseen tarvitaan siis 8 tavua.

Write_bit() vaihtaa tiedosto-osoitimen määräävän tavun viimeisen bitin parametrinaan saama bitin (tavu jonka arvo on 0 tai 1).

2.2.4 files.c

Tiedosto files.c sisältää tiedostojen lukemiseen liittyvät funktiot poisluettuna bittioperaatioita käyttävät funktiot. Funktio file_size() palauttaa parametrinaan saaman .bmp tiedoston pikselien datan tavuina. Koko saadaan laskemalla koko tiedoston pituus ja vähentämällä siitä pikselien datan alkukohta. Read_user_message() lukee salattavan viestin joko stdinistä tai erillisestä tiedostosta ja tallentaa sen asetustietueeseen yhtenä merkkijonona. Read_pixel_offset() palauttaa 32 bittisen luvun, joka on tavujen määrä tiedoston alusta, pixelien datan alkuun.

2.2.5 functions.c

Tähän tiedostoon on koottu funktioita jotka eivät kuulu muihin kokonaisuuksiin. Funktiot read_message() ja write_message() ovat ohjelman kannalta erittäin keskeisiä. suurin osa suoritusajasta vietetään näiden funktioiden kirjainten luku/kirjoitus silmukoissa. Molemmat edellä mainituista funktioista siirtävät tiedosto-osoittimen pikseleiden datan alkuun. Tämän jälkeen write_message() kirjoittaa kirjain kerrallaan viestin kuvaan ja read_message() lukee vastaavasti. Lisäksi write_message() tarkistaa että kirjoitettava viesti mahtuu kuvaan. Write_message() palauttaa nollan jos kirjoittaminen onnistui ja vitosen jos viesti oli liian pitkä. Muita virheentarkistuksia ei tehdä, tai ne on tehty aikaisemmin. Functions.c tiedostossa on myös funktio print_help(), joka suoritetaan ku ohjelma ajetaan komentoriviparametrilla -h.

2.2.6 bmp_hide.h

Tiedosto bmp_hide.h sisältää esittelyt kaikille ohjelmassa käytettäville funktioille (mainia lukuunottamatta), sekä muutaman vakion ja asetustietueen määritelmän. Vakioita READ, WRITE ja UNDEFINED käytetään asetustietueessa kuvaamaan prosessin suuntaa (eli kirjoitetaanko vai luetaanko). UNDEFINED on suunnan alkuarvo jota käytetään hyväksi komentoriviparametreistä luetuissa asetuksissa: Jos asetusten jälkeen suunnan arvo on UNDEFINED, aiheutuu virhe ja ohjelmaa ei suoriteta loppuun. Tekstiä luettaessa stdinistä tai tiedostosta, se luetaan ensin puskuriin ja puskuri katenoidaan yksittäiseksi merkkijonoksi. Vakio BUFFERSIZE määrittelee tämän puskurin koon. Arvo on asetettu yhteen kibiin. Tämä rajoittaa käyttäjän syötteen/luettavan tiedoston rivin pituudeksi 1024-1=1023 merkkiä. Jos käyttäjä haluaa syöttää ohjelmalle pidempiä rivejä, tulee BUFFERSIZE:n arvoa muuttaa ja ohjelma kääntää uudestaan. PIXEL_OFFSET on vakio jota käytetään lukiessa bmp-tiedoston header osasta pikselien sijainti tiedostossa. Sen arvo on 0xA eli 10, mikä on bmp tiedostossa se paikka josta kyseinen tieto löytyy.

2.3 Kääntäminen ja linkitys

Ohjelmakoodin ohessa on tiedosto nimeltä makefile. Makefile sisältää ohjeet kääntämisestä ja linkittämisestä. Makefilen suorittamiseksi käytettävässä järjestelmässä on oltava asennettuna GNU make tai jokin vastaava ohjelma, joka ymmärtää GNU makelle tehtyjä makefilejä. Ohjelman voi kuitenkin kääntää ja linkittää gcc:llä seuraavasti:

```
gcc -o bmp_hide main.c options.c bitwise.c files.c functions.c
```

Yllä oleva komento kääntää ja linkittää c -kieliset tiedostot binääritiedostoksi bmp_hide, joka on suorituskelpoinen ohjelma.

2.4 Paluuarvojen analysointi

Onnistuneen suorituksen jälkeen ohjelma palauttaa sitä kutsuneeseen ympäristöön arvon 0. Jos virheitä on tapahtunut, ei ohjelman suoritus etene loppuun asti, vaan se keskeytyy ja palauttaa arvon väliltä 1-5. Arvot tarkoittavat seuraavia virheitä:
1 - Asetuksia ei luettu onnistuneesti: Määriteltyjä tiedostoja ei saatu auki tai kirjoitus ja lukeminen koitetaan tehdä yhdellä kertaa.
2 - Kuvatiedostoa ei ole määritelty
3 - Lukemista tai kirjoittamista ei ole määritelty
4 - Vipua -m käytetään lukemisen yhteydessä (sallittu vain kirjoitukselle)
5 - kuvatiedoston koko on liian pieni kirjoitettavalle viestille

3 Ohjelman suorittaminen

Ohjelman voi suorittaa kahdella eri tavalla; interaktiivisesti tai komentoriviparametreilla. Ohjelma käynnistyy interaktiiviseen tilaan, mikäli se ei saa yhtäkään komentoriviparametria. Jos parametreja syötetään yksikin kappale, niin asetukset luetaan parametreista.

3.1 Interaktiivisessa tilassa

käyttäjältä kysytään ensin kirjoitetaanko vai luetaanko tiedostoon. Tämän jälkeen syötettä luetaan niin kauan kunnes käyttäjä kirjoittaa r (read) tai w (write). Seuraavaksi kysytään tiedostonimi. Mikäli ensimmäisessä kohdassa valittiin lukeminen, ohjelma lukee viestin tiedostosta tai ilmoittaa ettei tiedostoa voida lukea. Jos ensimmäisessä kohdassa valittiin kirjoittaminen, kysyy ohjelma vielä tämän jälkeen viestin käyttäjältä. Viesti voi olla rajattoman pitkä (ei kuitenkaan pidempi kuin tiedoston koko/8). Viestiä luetaan stdio:sta kunnes end-of-file saavutetaan. Useimmissa unix- järjestelmissä ctrl+d ilmaisee syötteen loppumista. Viestin kirjoituksen voi siis lopettaa painamalla ctrl+d uuden rivin alussa.

3.2 Komentoriviparametreilla

Sallittuja komentoriviparametrejä ovat -r -w -h -f ja -m
-r ilmaisee lukemista (read)
-w ilmaisee kirjoittamista (write)
-h tulostaa lyhyen help messagen
-f [file] määrittelee kuvatiedoston
-m [file] määrittelee tekstitiedoston joka kirjoitetaan kuvaan
Valitsimet r ja w ovat vaihtoehtoisia. Vain toista voidaan käyttää yhdellä kertaa. Kuitenkin jompaa kumpaa on pakko käyttää. Myös kuvatiedosto on välttämätön määritellä. Kirjoitettaessa viestiä, voidaan määritellä viesti luettavaksi erillisestä tiedostosta vivulla -m. Jos parametria -m ei ole annettu, kysyy ohjelma viestin interaktiivisesti. Valitsinta -m ei voi käyttää -r kanssa samanaikaisesti. Komentoriviparametrit voidaan antaa vapaavalintaisessa järjestyksessä. Komennot

```
./bmp_hide -f kuva.bmp -r
./bmp_hide -r -f kuva.bmp
```

ajavat siis saman asian.

3.3 esimerkkiajoja

Kirjoitetaan viesti tiedostosta message.txt kuvaan kuva.bmp

```
./bmp_hide -w -m message.txt -f kuva.bmp
```

saman voi tehdä myös putkea käyttäen:

```
cat message.txt | ./bmp_hide -w -f kuva.bmp
```

Yllä olevat komennot ovat siis ekvivalentit.
Kirjoitetaan interaktiivisesti määritelty viesti (ei käytetä valitsinta -m) kuvaan kuva.bmp

```
[apa@jla tila]$ ./bmp_hide -f kuva.bmp -w
Type the message you want to hide. (Ctrl+d on new line ends input.)
tämä on viesti
[apa@jla tila]$
```

Luetaan viesti kuvasta kuva.bmp

```
[apa@jla tila]$ ./bmp_hide -r -f kuva.bmp

tämä on viesti

[apa@jla tila]$
```

Viestin tulostukselle erilliseen tiedostoon ei tässä ohjelmassa ole toteutusta. Koska tulos kirjoitetaan stdoutiin voi sen putkittaa halutessaan erilliseen tiedostoon:

```
./bmp_hide -r -f kuva.bmp | viesti.txt
```