

Cognome: _____ Nome: _____ Matricola: _____ Calcolatore: _____

Sistemi Operativi — A.A. 2007-2008, prova pratica del 8 luglio 2008

Compito A

Vietato comunicare con chiunque. Vietato l'uso di rete, cellulari, floppy disk, pen drive e affini. Libri chiusi. Si può usare tutta la documentazione disponibile sul calcolatore. Non spegnere mai il calcolatore. Se hai problemi con il calcolatore rivolgiti subito al docente. Tempo a disposizione: 60 minuti.

Leggere attentamente prima di iniziare

- Scarica dall'url <http://192.168.161.70/compito.tar.gz> il pacchetto dei file che ti servono per il compito e scompattalo all'interno della tua home directory. (suggerimento: “wget <http://192.168.161.70/compito.tar.gz> ; tar xvzf compito.tar.gz ”)
- Scrivi nome, cognome, matricola e numero del calcolatore su questo foglio.
- **Scrivi gli stessi dati nel file `dati_studente.txt`.**
- Prepara un documento di identità a portata di mano.
- **Metti tutti i file che crei durante il compito nella directory `~/compito` dentro la home** (nel seguito la tilde “~” signifca “home dell'utente”).
- Gli esercizi sono etichettati con 1,2 o 3 asterischi:
 - * = esercizio semplice, risolvi questi prima degli altri
 - ** = esercizio di media difficoltà
 - *** = esercizio difficile, risolvi dopo aver risolto gli altri

Istruzioni per la consegna del compito

Non spegnere il calcolatore e recati dal docente con questo foglio compilato.

Esercizio 1

Il file di testo `alert.txt` contiene il log generato da un sistema di rilevamento di intrusioni. Nel file ciascun record è separato da una linea vuota. Ciascun record rappresenta un pacchetto riconosciuto come potenzialmente pericoloso. Per svolgere l'esercizio non è necessario conoscere il significato di tutti i campi. Suggerimenti: alcune volte conviene processare tale file con `awk` usando `RS=""` (stringa vuota) e `FS="\n"`, ricorda che, in `awk`, `gsub()` e' un efficace strumento di sostituzione.

1. * La terza riga di ciascun record contiene sempre una stringa *ipsorgente:porta* -> *ipdestinazione:porta* (la *porta* può mancare). Mostra un comando che selezioni tra tali righe quelle che hanno in *ipsorgente* il terzo numero pari a **127** (es. 193.16.**127**.0:123 -> 193.10.10.0:6503).
Scrivi nel file `~/compito/esercizio1/soluzione1.1.txt` il comando usato e il suo output (fai copia-e-incolla dal terminale).
2. ** La prima riga di ciascun record è del tipo “[...] [...] *nomeattacco* [...]”. Mostra un comando che dia una tabella, con colonne *quantità ipsorgente nomeattacco*, che mostri quante volte una certa sorgente ha inviato un attacco con tale nome. Dall'output vanno tolte le parti tra parentesi quadre “[...]”.
Scrivi nel file `~/compito/esercizio1/soluzione1.2.txt` il comando usato e il suo output (fai copia-e-incolla dal terminale).
3. *** [questo punto richiede più tempo degli altri, svolgilo per ultimo] Scrivere uno script che elabori i dati da `alert.txt` e produca su standard output il seguente report:
 - i record sono separati da una riga vuota
 - esiste un record per ciascun valore *ipdestinazione*, tale valore è indicato nella prima riga del record
 - le restanti righe del record contengono tutti i valori *nomeattacco* (eventualmente duplicati) che compaiono in records assieme a quell'*ipdestinazione*
 - i record devono essere ordinati in modo che *ipdestinazione* più frequenti nel file siano in cima.

Il nome del tuo script deve essere `~/compito/esercizio1/script.sh`

Esercizio 2

Il programma "fib" contenuto in prj1 è composto da più file con estensione ".c" e ".h". Tale programma inizializza una lista contenente una serie di numeri, derivati mediate una funzione, dai primi n numeri di fibonacci e ne stampa il contenuto (n è passato come parametro). La directory prj2 contiene una copia di prj1 che devi modificare per rispondere alle seguenti domande.

1. * Usa il comando "gcc -Wall *.c" per una prima compilazione (l'opzione -Wall attiva tutti i warning). Correggi gli errori/warning di pre-processamento, di compilazione e di link che incontri. Mostra tutti gli errori e spiegali man mano che li correggi. Mostra quindi l'output di compilazione ed esecuzione del programma corretto. Nota che per eseguire il programma devi fornire un parametro, usa, ad esempio, "5".

Per correggere gli errori modifica la copia in ~/compito/esercizio2/prj2 e lascia intatto ~/compito/esercizio2/prj1.

Scrivi in ~/compito/esercizio2/soluzione2.1.txt la soluzione di questo esercizio (fai copia-e-incolla dal terminale per gli errori e l'output).

2. ** Crea un Makefile con i seguenti target **badando a mettere le dipendenze opportune**
 - fib: crea l'eseguibile "fib" linkato dinamicamente con simboli di debug e compilazione con tutti i warning
 - fib_release: crea l'eseguibile "fib" linkato staticamente senza simboli di debug e compilazione ottimizzata (vedi man gcc)
 - clean: pulisce il progetto cancellando i file inutili (*.o, *.~)
 - delete: come clean ma cancella anche i target
 - main.asm: disassemblato del file main.o

Il Makefile si deve chiamare ~/compito/esercizio2/prj2/Makefile

3. * Crea una patch tra prj1 e prj2 e provala su una copia "prj1copy" che crei tu.

Metti nel file ~/compito/esercizio2/soluzione2.3.txt i comandi per creare e per applicare la patch (fai copia-e-incolla dal terminale). Metti la patch nel file ~/compito/esercizio2/soluzione2.3_patch.txt.

4. ** Considera una esecuzione di fib con parametro 10. Mostra lo stack subito dopo a 108-esima invocazione della fib().
Mostra il contenuto delle strutture *element* che rappresentano gli ultimi tre elementi della lista per l'esecuzione "fib 10".

Metti nel file ~/compito/esercizio2/soluzione2.4.txt i comandi per portare a termine l'esperimento (fai copia-e-incolla dal terminale) e la risposta alla domanda.

Compito B

Vietato comunicare con chiunque. Vietato l'uso di rete, cellulari, floppy disk, pen drive e affini. Libri chiusi. Si può usare tutta la documentazione disponibile sul calcolatore. Non spegnere mai il calcolatore. Se hai problemi con il calcolatore rivolgiti subito al docente. Tempo a disposizione: 60 minuti.

Leggere attentamente prima di iniziare

- Scarica dall'url <http://192.168.161.70/compito.tar.gz> il pacchetto dei file che ti servono per il compito e scompattalo all'interno della tua home directory. (suggerimento: “`wget http://192.168.161.70/compito.tar.gz ; tar xvfz compito.tar.gz`”)
- Scrivi nome, cognome, matricola e numero del calcolatore su questo foglio.
- **Scrivi gli stessi dati nel file `dati_studente.txt`.**
- Prepara un documento di identità a portata di mano.
- **Metti tutti i file che crei durante il compito nella directory `~/compito` dentro la home** (nel seguito la tilde “`~`” significa “home dell'utente”).
- Gli esercizi sono etichettati con 1, 2 o 3 asterischi:
 - * = esercizio semplice, risolvi questi prima degli altri
 - ** = esercizio di media difficoltà
 - *** = esercizio difficile, risolvi dopo aver risolto gli altri

Istruzioni per la consegna del compito

Non spegnere il calcolatore e recati dal docente con questo foglio compilato.

Esercizio 1

Il file di testo `alert.txt` contiene il log generato da un sistema di rilevamento di intrusioni. Nel file ciascun record è separato da una linea vuota. Ciascun record rappresenta un pacchetto riconosciuto come potenzialmente pericoloso. Per svolgere l'esercizio non è necessario conoscere il significato di tutti i campi. Suggerimenti: alcune volte conviene processare tale file con `awk` usando `RS=""` (stringa vuota) e `FS="\n"`, ricorda che, in `awk`, `gsub()` è un efficace strumento di sostituzione.

1. * La terza riga di ciascun record contiene sempre una stringa *ipsorgente:porta* -> *ipdestinazione:porta* (la *porta* può mancare). Mostra un comando che selezioni tra tali righe quelle che hanno in *ipsorgente* il secondo numero pari a **204** (es. `193.204.127.0:123` -> `193.10.10.0:6503`).
Scrivi nel file `~/compito/esercizio1/soluzione1.1.txt` il comando usato e il suo output (fai copia-e-incolla dal terminale).
2. ** La seconda riga di ciascun record è del tipo “[Classification: *nomeattacco*] [...]”. Mostra un comando che dia una tabella, con colonne *quantità ipsorgente nomeattacco*, che mostri quante volte una certa destinazione ha ricevuto un attacco con tale nome. Dall'output vanno tolte le parti “[Classification: ” e “[...]”.
Scrivi nel file `~/compito/esercizio1/soluzione1.2.txt` il comando usato e il suo output (fai copia-e-incolla dal terminale).
3. *** [questo punto richiede più tempo degli altri, svolgilo per ultimo] Scrivere uno script che elabori i dati da `alert.txt` e produca su standard output il seguente report:
 - i record sono separati da una riga vuota
 - esiste un record per ciascun valore *ipdestinazione*, tale valore è indicato nella prima riga del record
 - le restanti righe del record contengono tutti i valori *nomeattacco* (eventualmente duplicati) che compaiono in records assieme a quell'*ipdestinazione*
 - i record devono essere ordinati in modo che *ipdestinazione* più frequenti nel file siano in cima.

Il nome del tuo script deve essere `~/compito/esercizio1/script.sh`

Esercizio 2

Il programma "fib" contenuto in prj1 è composto da più file con estensione ".c" e ".h". Tale programma inizializza una lista contenente una serie di numeri, derivati mediante una funzione, dai primi n numeri di fibonacci e ne stampa il contenuto (n è passato come parametro). La directory prj2 contiene una copia di prj1 che devi modificare per rispondere alle seguenti domande.

1. * Usa il comando "gcc -Wall *.c" per una prima compilazione (l'opzione -Wall attiva tutti i warning). Correggi gli errori/warning di pre-processamento, di compilazione e di link che incontri. Mostra tutti gli errori e spiegali man mano che li correggi. Mostra quindi l'output di compilazione ed esecuzione del programma corretto. Nota che per eseguire il programma devi fornire un parametro, usa, ad esempio, "5".

Per correggere gli errori modifica la copia in ~/compito/esercizio2/prj2 e lascia intatto ~/compito/esercizio2/prj1.

Scrivi in ~/compito/esercizio2/soluzione2.1.txt la soluzione di questo esercizio (fai copia-e-incolla dal terminale per gli errori e l'output).

2. ** Crea un Makefile con i seguenti target **badando a mettere le dipendenze opportune**
 - fib: crea l'eseguibile "fib" linkato dinamicamente con simboli di debug e compilazione con tutti i warning
 - fib_release: crea l'eseguibile "fib" linkato staticamente senza simboli di debug e compilazione ottimizzata (vedi man gcc)
 - clean: pulisce il progetto cancellando i file inutili (*.o, *.~)
 - delete: come clean ma cancella anche i target
 - main.asm: disassemblato del file main.o

Il Makefile si deve chiamare ~/compito/esercizio2/prj2/Makefile

3. * Crea una patch tra prj1 e prj2 e prova la su una copia "prj1copy" che crei tu.

Metti nel file ~/compito/esercizio2/soluzione2.3.txt i comandi per creare e per applicare la patch (fai copia-e-incolla dal terminale). Metti la patch nel file ~/compito/esercizio2/soluzione2.3_patch.txt.

4. ** Considera una esecuzione di fib con parametro 10. Mostra lo stack subito dopo a 108-esima invocazione della fib().
Mostra il contenuto delle strutture *element* che rappresentano gli ultimi tre elementi della lista per l'esecuzione "fib 10".

Metti nel file ~/compito/esercizio2/soluzione2.4.txt i comandi per portare a termine l'esperimento (fai copia-e-incolla dal terminale) e la risposta alla domanda.