UNIVERSITÀ DEGLI STUDI DI PERUGIA

FACOLTÀ DI SCIENZE MM. FF. NN.

Laurea specialistica in Informatica

Corso di Sicurezza Informatica

SECURITY MODELS

STUDENTE
Nigro Antonio

DOCENTE
Bistarelli Stefano

A.A. 2008/2009

ARGOMENTI

- The Harrison-Ruzzo-Ullman (HRU) model
- Information Flow models
  - Riconoscere i flussi d'informazione
- Noninterference models
- Execution monitor

IL MODELLO HRU

Per descrivere il modello abbiamo bisogno:

- un set di soggetti S;
- un set di oggetti O;
- un set di diritti di accesso R;
- una matrice degli accessi $M=(M_{so})$ $s \epsilon S$, $o \epsilon O$; ogni casella di M contiene un sottoinsieme di R specificando i diritti del soggetto s sull'oggetto o.

IL MODELLO HRU

	o_1	o_2	o_3	...	o_n
s_1					
s_2		**r**	**wr**		
s_3					
...					
s_n			**r**		

IL MODELLO HRU

Esistono sei operazioni per manipolare l'insieme dei soggetti S, degli oggetti O e la matrice M:

- enter r into M
- delete r from M
- create subject s
- delete subject s
- create object o
- delete object o

IL MODELLO HRU

command c(x_1,...,x_k)
    if r1 in $M_{s1,o1}$ and
    if r2 in $M_{s2,o2}$ and
    .
    .
    if rm in $M_{sm,om}$ then
        op1
        op2
        .
        .
        opn
end

Esempio:

command
create_file(s,f)
    create f
    enter o into Ms,f
    enter r into Ms,f
    enter w into Ms,f
end

IL MODELLO HRU

POLICY: controllare che non esistano vie che permettano di concedere diritti di accesso indesiderati.

IL MODELLO HRU

- Una matrice M è detta **LEAK** rispetto ad un permesso r se esiste un comando c che aggiunge il permesso r in una posizione della matrice che non lo conteneva.
- Una matrice M è detta **SAFE** rispetto ad un permesso r se nessuna sequenza di comandi permette di portare lo stato della matrice nello stato LEAK.

LIMITI DEL MODELLO HRU

Teorema. Data una matrice degli accessi M e un diritto r, verificare la sicurezza di M rispetto a r è un problema non decidibile.

Dim. Ridurre il problema di verificare la sicurezza di M al problema dell'arresto della macchina di Turing che sappiamo essere non decidibile.

LIMITI DEL MODELLO HRU

Teorema. Dato un sistema composto di comandi mono-operazionali, una matrice degli accessi M e un diritto r, verificare la sicurezza del sistema rispetto al diritto r è decidibile.

Dim. Il numero minimo di comandi per conferire un diritto contiene almeno m=|r|*(|s|+1)*(|o|+1)+1 comandi. La verifica consiste nel controllare tutte le sequenze di comandi di lunghezza m, se il diritto è stato conferito si troverà in una di queste squenze.

LIMITI DEL MODELLO HRU

Lipton e Snyder nel 1978:

Teorema. Dato un sistema composto da un numero arbitrario di comandi, se il numero di soggetti è finito allora il sistema è decidibile.

In conclusione si può affermare che anche se non esiste un algoritmo capace di verificare se un sistema complesso è sicuro o meno, questo modello però non esclude la possibilità di effettuare verifiche su particolari sistemi di protezione.

INFORMATION FLOW MODELS

I componenti di un modello Information Flow sono:

- un reticolo “lattice” (L,≤) di livelli di sicurezza;
- un insieme di oggetti etichettati;
- una politica di sicurezza: il flusso d'informazione da un oggetto con etichetta c1 ad un oggetto con etichetta c2 è permesso se e solo se c1≤c2; ogni flusso d'informazione che viola tale regola non è ammesso.

INFORMATION FLOW MODELS

Flusso di Informazione:

- fluire di informazione da un oggetto x ad un oggetto y;
- possiamo capire meglio x osservando y.

Se si sa già tutto su x, non può fluire informazione da x.

INFORMATION FLOW MODELS

Tipi di flusso:

- flusso esplicito: osservando y dopo l'assegnamento `y:=x`, si conosce il valore di x.

- flusso implicito: osservando y dopo la condizione `IF x=0 THEN y:=1`, è possibile dire qualcosa su x anche se l'assegnamento non è stato eseguito. Per esempio if y=2, sappiamo che x è diverso da 0.

INFORMATION FLOW MODELS

Entropia Condizionata (Equivocation):

- misura il flusso di informazione.

$$H_y(x) = -\sum_{j=1}^{m} q(y_j) \sum_{i=1}^{n} p(x_i|y_j) \log_2 p(x_i|y_j)$$

- probabilità $q(y_j)$ con $y_j \in \{y_1,\ldots,y_m\}$
- probabilità condizionata $p(x_i|y_j)$

RICONOSCERE I FLUSSI DI INFORMAZIONE

Analisi delle politiche di sicurezza:

- statica
- dinamica

Meccanismi basati sull'anailsi del codice di un programma:

- Compiler-based
- Execution-based

RICONOSCERE I FLUSSI DI INFORMAZIONE

Distinguiamo i costrutti del codice:

- costrutti di assegnamento
- costrutti condizionali e iterativi

Definizione. *Un insieme di costrutti è certificato rispetto ad una politica di information flow se il flusso d'informazione che passa attraverso l'insieme di costrutti non viola la politica.*

RICONOSCERE I FLUSSI DI INFORMAZIONE

La notazione $\underline{x} \leqslant \underline{y}$ indica che l'informazione può fluire da un elemento di classe x ad un elemento di classe y.

Se osserviamo le classi come un Lattice e l'informazione può fluire dalle classi A e B nella variabile x, allora $lub\{A, B\} \leqslant \underline{x}$.

RICONOSCERE I FLUSSI DI INFORMAZIONE

Compiler-based:

- determina se i flussi di informazione in un programma possano violare la politica di information flow
- meccanismo restrittivo

Esempio. Consideriamo il seguente costrutto di assegnamento,

meccanismo Compiler-based:

x:=y+z;

il requisito per far sì che il flusso di informazione sia sicuro è

$lub\{\underline{y}, \underline{z}\} \leqslant \underline{x}$.

RICONOSCERE I FLUSSI DI INFORMAZIONE

Nel caso in cui il costrutto è di tipo iterativo

```
while f(x1,...,xn) do
    S;
```

i requisiti per rendere il costrutto sicuro sono:

- l'iterazione deve terminare;
- S sia sicuro;
- $lub\{\underline{x1},\ldots,\underline{xn}\} \leqslant glb\{\underline{y} | y\ è\ l'\ obbiettivo\ di\ un\ assegnamento\ in\ S\}.$

RICONOSCERE I FLUSSI DI INFORMAZIONE

Execution-based:

- previene i flussi di informazione che violano la politica. Esso raggiunge tale obbiettivo per costrutti che contengono flussi di tipo esplicito
- difficoltà flussi impliciti

Soluzione Fenton's Data Mark Machine:

- tag di sicurezza ad ogni variabile
- tag al Program Counter

NONINTERFERENCE MODELS

- Divisione degli utenti su gruppi: Low e High
- Gli utenti del gruppo High non interferiscono con il gruppo Low se i comandi che essi eseguono non modificano le cose che il gruppo Low può vedere.

- Questi modelli sono oggigiorno argomento di ricerca.

EXECUTION MONITOR

Il termine Execution Monitoring (EM) è stato introdotto da Schneider (2000) per meccanismi che monitorano i passi d'esecuzione di un sistema e terminano l'esecuzione se si verifica una violazione della politica di sicurezza.

Esempi: firewalls, sistemi operativi, web services...

EXECUTION MONITOR

Consideriamo tre classi di politiche di sicurezza (Schneider, 2000):

- Access Control: definiscono delle restrizioni sulle operazioni principali che possono essere eseguite su un oggetto;
- Information Flow: limitano le operazioni principali che possono dedurre informazione su di un oggetto osservando il comportamento del sistema;
- Availability: limitano le operazioni principali negando l'utilizzo di una risorsa ad alcuni soggetti.

LIMITS of EXECUTION MONITOR

- Non possono predire i risultati dell'esecuzione che stanno osservando in quanto non hanno un modello del sistema target (obbiettivo).

  Es. compilatori ma non rientrano nella categoria EM
- Non possono modificare un target prima di eseguirlo.

  Es. In-line reference monitors e la programmazione Reflection-oriented ma non rientrano nella categoria EM.

PROPERTIES OF EXECUTION

- L'esecuzione di un sistema target = sequenza di passi
- Indichiamo con Ψ l'insieme di tutte le esecuzioni finite e infinite
- Con $\sigma[..i]$ indichiamo i primi i passi di σ, $\sigma \epsilon \Psi$
- Un insieme Γ di esecuzioni è detto proprietà se i membri di un elemento sono determinati dal singolo elemento e non da altri elementi dell'insieme.

PROPERTIES OF EXECUTION

- Una politica di sicurezza deve essere una proprietà per essere analizzata da un meccanismo EM.

- Non tutte le politiche sono una proprietà.

  Es.le politiche Information Flow: il flusso d'informazione da una variabile x in una variabile y ottenuto da una esecuzione dipende, parzialmente, dal valore che y assume in altre possibili esecuzioni.

PROPERTIES OF EXECUTION

- Non tutte le proprietà sono analizzabili da EM.
- I meccanismi EM non possono prevedere i passi successivi quando viene presa una decisione su un'esecuzione.

  Es. Consideriamo una esecuzione sicura σ che ha un prefisso σ' non sicuro. L'esecuzione probabilmente terminerà passando attraverso un stato non sicuro.

- Un EM deve vietare un prefisso non sicuro e fermare l'esecuzione. Approcio conservativo.

SAFETY & LIVENESS

Tra le proprietà dell'esecuzioni citiamo:

- *safety*: nulla di sbagliato può accadere.
- *liveness*: eventualmente qualcosa di buono accadrà.

Esiste una relazione tra la proprità safety e il tipo di politica da analizzare.

Definizione. Una prorpietà Γ è detta safety se per ogni esecuzione finita o infinita σ

$$\sigma \notin \Gamma \Rightarrow \exists i \left(\forall \tau \in \Psi : \sigma \left[.. i \right] \tau \notin \Gamma \right).$$

SAFETY & EM

Definizione. Un meccanismo EM analizza politiche di sicurezza che sono proprietà safety.

- Se un insieme di esecuzioni per una politica di sicurezza non è proprietà safety allora la politica non ha un meccanismo di analisi EM.

SAFETY & POLITICHE DI SICUREZZA

- Access Control: definiscono proprietà safety; esecuzioni parziali che terminano con la prova di un'operazione inaccettabile saranno vietate.
- Information Flow: non definiscono insiemi di esecuzioni che sono proprietà; di conseguenza Information Flow non può essere una proprietà safety e non può essere analizzato da EM.
- Availability: non definiscono proprietà safety; qualche esecuzione parziale potrebbe essere estesa così che permetta un accesso alla risorsa.

Le politiche Availability che si riferiscono al Maximum Waiting Time (MWT) sono proprietà safety. Non appena un'esecuzione resta in attesa superando il valore corrispondente a MWT qualsiasi estensione sarà considerata una violazione della politica.