18734 Homework 1
Privacy Audits with REDUCE

Due:12 noon, Sep 30

Workflow for REDUCE

Basics

Each student should have a login for the server `possibility.cylab.cmu.edu`. You can ssh into the machine and change your password. You can also use scp/sftp to put or get files from the server if you are not comfortable editing with linux editors. If you are using Windows, The Tectia client [1] allows you to ssh into the server, and also copy files to and from the server to your computer. Please be careful about the end of line characters that differ on Windows and Linux. Notepad++ is a good Windows editor that allows converting Windows EOL to Linux EOL.

The binary file for the tool named "test-hipaa" has been provided in the home directory. The file containing the policy is stored in $HOME/hipaa-case-study/hipaa.f and the file containing various configurations necessary for test-hipaa is stored in $HOME/hipaa-case-study/hipaa.d. The database file that stores the log must be in the $HOME directory and must be named hipaa.db.

Policy Specification

The provided hipaa.f specifies a policy in first order logic. The file uses prefix notation, i.e., "(a) and (b)" is written as "and (a) (b)". Following the restrictions on forall and exists (as explained in class) $\forall x.\ c(x) \supset B(x)$ is written as "all [x] ($c(x)$) ($B(x)$)" with the implication being implicit. (Note the brackets carefully). The table below lists the syntax mapping

infix	prefix(ish)
(a) and (b)	and (a) (b)
(a) or (b)	or (a) (b)
(a) imp (b)	imp (a) (b)
(a) plus (b)	plus (a) (b)
$\forall x, y.\ c(x, y) \supset B(x, y)$	all [x][y] ($c(x, y)$) ($B(x, y)$)
$\exists x, y.\ c(x, y) \wedge b(x, y)$	ex [x][y] ($c(x, y)$) ($b(x, y)$)
predicate-name(arg1, ...)	(predicate-name arg1 ...)

An important point to remember (in problem 2) is that $\exists x, y.\ c(x, y) \wedge b(x, y) \wedge a(x, y)$ must be written as "ex [x][y] ($c(x, y)$) (and $b(x, y)$ $a(x, y)$)", as ex only takes two arguments. Further

[1] `http://www.cmu.edu/computing/software/all/tectia/index.html`

remember that c is a guard and hence c must have finite substitutions. When checking the audit log the reduce algorithm checks the guard first.

How to run REDUCE

1. For each Problem, you have to add one (or more) clauses into the policy file (hipaa.f). You will find an existing hipaa.f file (along with hipaa.d) in the folder "hipaa-case-study". You will have to convert a given clause into a logic statement, and add it to the policy file (hipaa.f).

2. In order to test your policy specification, you would require audit logs, which are essentially database (hipaa.db) files. In your parent directory, you will find several .sql files which can generate appropriate audit logs. You can generate audit logs by running
   "sqlite3 hipaa.db < ⟨filename⟩.sql".
   Remember, the database file MUST be named hipaa.db, and must be located in the parent directory for the tool to work. Also, in order to test every new log, you MUST remove the old hipaa.db file, otherwise the sql script will over-write the existing file, and you wont get expected results.

3. The tool runs from the command line ("./test-hipaa") and uses the policy file (hipaa.f), the database file (hipaa.db) and the specifications file (hipaa.d) to produce an output. You do not need to edit the hipaa.d file for this assignment. When the tool is run output is either audit passed (true) or audit failed (false). The first line of output is the time taken, followed by (true or false) or a subjective predicate in some cases. E.g.,

   ```
   %%%%%%%%% time used: 0.001%%%%%%%%%%%%
   (<POL/DISCLOSE> false)
   ```

   ```
   %%%%%%%%% time used: 0.001%%%%%%%%%%%%
   (is-valid-authorization (str2msg ”_fake”) (str2prin ”_hospital0”) (str2prin ”_rv1”) (str2prin
   ”_p1”) 1971:04:01:20:30:00)
   ```

Variable Modes and Use of Guards

To understand which predicates can be used in the guard, it is important to understand variable modes in predicates. You can find variable modes in addition to other information in the file `hipaa.d`. Some rules for designing the guard and information about where to find mode information follows:

1. The guard cannot contain a subjective predicate If a predicate is subjective, you can see it in the `hipaa.d` file. For example,

   ```
   is-valid-authorization:  message(+) -> prin(+) -> prin(+) -> prin(+) -> date(+)
   -> pred SUBJ.
   ```

   is subjective predicate as indicated by 'pred SUBJ' at the end of the definition.

2. The guard can contain a predicate which has all variables in the output mode You can find the mode of the variables in a predicate beside the variables in the predicate definition in the hipaa.d file. For example,

   ```
   send:  prin(-) -> prin(-) -> message(-) -> date(-) -> pred DB.
   ```

   can be used in a guard because all the variables (prin, prin, message, date) are in output mode. `pred DB` indicates that REDUCE finds finite substitutions by searching through a database.

3. If a predicate has a variable in input mode, it must be preceded by another predicate which has the same variable in output mode. For example, if look at the send and `eq_ msg` predicates

   ```
   send:  prin(-) -> prin(-) -> message(-) -> date(-) -> pred DB
   eq_ msg:  message(+) -> message(-) -> pred EVAL.
   ```

   $$
   \begin{array}{l}
   all[p1][p2][m][i][u][pp] \\
   \qquad (and \\
   \qquad\quad (send\ p1\ p2\ m\ u) \\
   \qquad\quad (eq\_msg\ m (msg\ i\ pp)) \\
   \qquad) \\
   \qquad (true)
   \end{array}
   $$

   is a valid policy statement, because m is in input mode in *eq_msg*, but in output mode in *send*.

An interesting example is

```
inrelation:  prin(+) -> prin(+) -> relation(-) -> date(-) -> pred DB.
```

In the provided hipaa.f file, you would find that inrelation is used in the guard despite having two variables in the input mode:

$$
\begin{array}{l}
ex[u1] \\
\qquad (inrelation\ p\ p2\ treatment-relation\ u1)
\end{array}
$$

This is because the guard of $ex[u1]$ requires finite substitutions of u and not variables not quantified here. The two `prin` variables p and $p2$ are grounded in the outermost guard:

$$
\begin{array}{l}
(and \\
\qquad (send\ p1\ p2\ m\ u) \\
\qquad (eq\_msg\ m (msg\ i\ pp)) \\
\qquad (hasattrof\ \ i\ p\ t)
\end{array}
$$

Problem 1 [10]

a) Understanding notation

1. Translate the given hipaa.f file into a logic formula and write it out. The predicates are send, hasattrof, inrelation, time_in, refers. This is a policy with two positive norms.

2. Next, with the following interpretations, explain the policy in English.

   (send p1 p2 m u) means that “p1” is sender of message “m” to “p2” at time “u”.

   (eq_msg m (msg i pp)) means that message “m” is the pair (i, pp), where “i” is info, “pp” is purpose.

   (hasattrof i p t) means info “i” contains attributes “t” about individual “p”.

   (inrelation p p2 treatment-relation u1) means “p” has treatment-relation with “p2” at time “u1”.

   (time_in u v w) means $u < v < w$, where time is counted in days.

   (refers p1 p2 p referral u1) mean “p1” referred individual “p” to “p2” at time “u1”.

Produce a log (hipaa.db) by executing “sqlite3 hipaa.db <prob1.sql”. Upon running the tool by executing “./test-hipaa”, the audit should fail, i.e. output false.

b) Adding a positive norm

Using the predicate:

(prescribes p2 q prescription w) meaning “p2” prescribes a prescription for patient “q” at time “w”

add a positive norm that says that “p2” prescribes a prescription for patient “q” within 30 days of the day “u” when the message “m” was sent.

To test if your written policy is correct run the tool again with the modified hipaa.f that contains the added positive norm. The log contains a message that is allowed by the added positive norm. So the audit succeeds, i.e. returns true.

Problem 2 [20]

Using the following predicates:

(attr_in t psychotherapy-notes) meaning attributes “t” contains the attribute “psychotherapy-notes”

(lessthan u t) meaning $u < t$

(is-valid-authorization m p1 p2 p u) meaning “m” is a valid authorization by “p” for “p1” to enable him sending message with “psychotherapy-notes” about “p” to ‘p2” at time “u”

add a negative norm stating that in cases when the message “m” sent by “p1” to “p2” contained “psychotherapy-notes” about “p”, then it must be the case that “p1” had, in the past, received a message from “p” that was a valid authorization by “p”.

There are two sql files provided for this problem: prob2fake.sql and prob2.sql. The log generated by prob2fake.sql contains a fake authorization sent by “p” to “p1” and the log in prob2.sql contains no sending of any such authorization. Further, it is specified in hipaa.d that the predicate is-valid-authorization is a subjective predicate.

To test if your written policy is correct, run the tool on the log generated by prob2fake.sql and prob2.sql (one at a time) with your modified hipaa.f that contains the negative norm. Then, if the negative norm has been written properly, with prob2.sql the audit will just fail and with prob2fake.sql the audit will output the subjective predicate is-valid-authorization. You should delete the existing hipaa.db file (using the command “rm hipaa.db”) before creating new ones, otherwise entries keep getting appended to the existing db file and you may get unexpected results.

Problem 3 [20]

For the given clause, do the following:

1. Identify the type of clause (positive or negative)
2. Write the clause in logic (predicates are provided below)
3. Put the clause in the hipaa.f file. An sql file is provided to test the correctness of your implementation.

A covered entity that participates in an organized health care arrangement may disclose protected health information about an individual to another covered entity that participates in the organized health care arrangement for any health care operations activities of the organized health care arrangement.

The predicates (functions) to you'll need are described below:

(inrole p role u) meaning p is in role "role" at time u.

(belongrole p role u) p is in role "role" at time u

(The above is a subjective predicate as opposed to inrole, which is known from a db)
"covered-entity", "organized-healthcare-arrangement" are roles that you will need to use. "covered-entity" is a recognized role, whereas "organized-healthcare-arrangement" is not a recognized role.

(purp_in pp purpose) meaning pp contains the purpose "purpose".

(is-participant-of-organized-health p q u) p is a participant in the organized health-care arrangement provided by q at time u

(healthcare-op p) is a function returning a purpose—the returned purpose is the health care operations of p.

"phi" is an known attribute meaning "protected health information".

The log created by prob3.sql should output the subjective predicate for the first clause (belongrole). That means that this log passed all checks of the first clause, except for the subjective predicate that the tool does not know how to evaluate.

Problem 4 [30]

For the given clause, do the following:

1. Identify the type of clause (positive or negative)
2. Write the clause in logic (predicates are provided below)
3. Put the clause in the hipaa.f file. An sql file is provided to test the correctness of your implementation.

A covered entity may disclose protected health information to a public health authority that is authorized by law to collect or receive such information for the purpose of preventing or controlling disease; or, at the direction of such an authorized public health authority, to a foreign government agency.

Two concrete roles to be used are “public-health-authority” and “foreign-gov-agency”. One concrete purpose is “disease-prevention-control”.

(is-authorized-by-law p1 pp u) meaning p1 is authorized by the law to collect or receive protected health information. (This is a subjective predicate)

(directed-disclosure p0 p1 p2 t pp u) meaning p0 has directed p1 to disclose to p2 information that contains the attribute t with purpose pp at time u.

The log created by prob4.sql should output one subjective predicate (is-authorized-by-law). That means that the log created by prob4.sql passed all checks, except for the subjective predicate that the tool does not know how to evaluate.

Problem 5 [20]

In this problem we extend the policy by adding exceptions to clauses (as described in class). The deliverables of this part is

1. Identify the type of clause (positive or negative)

2. Write the clause in logic (predicates are provided below)

3. Put the clause in the hipaa.f file. An sql file is provided to test the correctness of your implementation.

Notwithstanding any provision of this subpart, a covered entity must obtain an authorization for any use or disclosure of psychotherapy notes, except to carry out the following treatment, payment, or health care operations:
(A) Use by the originator of the psychotherapy notes for treatment;
(B) Use or disclosure by the covered entity for its own training programs in which students, trainees, or practitioners in mental health learn under supervision to practice or improve their skills in group, joint, family, or individual counseling;
or
(C) Use or disclosure by the covered entity to defend itself in a legal action or other proceeding brought by the individual

The predicates (functions) to be used in each clause are described below

(is-valid-authorization m p1 p2 p u) “m” s a valid authorization by “p” for “p1” to enable him sending message with “psychotherapy-notes” about “p” to ‘p2” at time “u”

(counseling-training-programs p) is a function returning a purpose—the returned purpose is counseling training programs of p.

(defense-in-legal-proceeding p q) is a function returning a purpose—the returned purpose is for defense of p1 in a legal case initiated by q.

One concrete attribute is “psychotherapy-notes”.

User the provided prob4.sql to create a log file. This log contains an authorization by patient followed by sending of message with psychotherapy notes by the hospital. The sends are so formulated such that on running the tool, the output will be two “is-valid-authorization” predicates.

Next, uncomment line 55 (remove leading −−) in prob4.sql and generate the log again. This line adds the purpose counseling and training. Now, the audit will succeed, and the tool will output 'true'.

Files to be submitted

Make a pdf file for the written parts. Put the hipaa.f and the pdf files in one zip file, name the file <your_ andrew_ id>_HW1.zip and submit on blackboard. The submission deadline is 12 noon, Sep 30. (it is 12 noon, not midnight).