Keeping Secrets in Incomplete Databases

Joachim Biskup and Torben Weibert

Information Systems and Security
Department of Computer Science
University of Dortmund, Germany

FCS'05 – June 30, 2005

Outline

1. Introduction
2. Constructing Safe Censors
   - Uniform Lying
   - Combined Lying and Refusal
   - Uniform Refusal
3. Formalization
4. Conclusion

Introduction

Controlled query evaluation

- preserves confidentiality in information systems
- checks confidentiality dynamically at runtime
- considers the *inference problem*
- provides a fundamental framework

Introduction

Controlled query evaluation

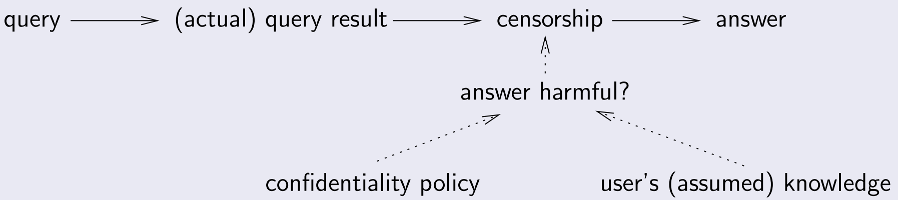
- preserves confidentiality in information systems
- checks confidentiality dynamically at runtime
- considers the *inference problem*
- provides a fundamental framework

Basic ideas

- administrator defines confidentiality policy – information to hide
- user issues a sequence of queries against database
- *censor* checks each query result for possible confidentiality violations
- if confidentiality is threatened, query result is *distorted*
  - lying (modified answer is returned)
  - refusal (no "useful" answer is returned)

Preserving Confidentiality: Preliminaries

The systems's perspective



Preserving Confidentiality: Preliminaries

The systems's perspective

query ⟶ (actual) query result ⟶ censorship ⟶ answer

answer harmful? ⇢ censorship

confidentiality policy ⇢ answer harmful?

user's (assumed) knowledge ⇢ answer harmful?

The user's perspective

query ⟶ answer ⟶ interpretation of answer ⟶ gain of information

awareness of the system ⇢ interpretation of answer

Our Framework

Possibly incomplete logic databases

- database schema *DS*: set of propositions
- instance *db*: (consistent) set of sentences with propositions from *DS*

Our Framework

Possibly incomplete logic databases

- database schema *DS*: set of propositions
- instance *db*: (consistent) set of sentences with propositions from *DS*

(Closed) query Φ: propositional sentence

- either *true*, *false* or *undef* in *db*
- query evaluation based on logical implication $\models_{PL}$

$$eval(\Phi)(db) := \texttt{case} \quad \begin{array}{lll} db \models_{PL} \Phi & : \mathit{true} & | \\ db \models_{PL} \neg\Phi & : \mathit{false} & | \\ \texttt{else} & : \mathit{undef} & \end{array} \quad \texttt{end}$$

Preserving Confidentiality

Security policy $pot\_sec = \{\Psi_1, \ldots, \Psi_m\}$ (set of propositional sentences)

If potential secret Ψ_i is *true* in *db*, user may *not* learn this fact
(if Ψ_i is *false* or *undef*, this fact may be disclosed)

Preserving Confidentiality

Security policy $pot\_sec = \{\Psi_1, \ldots, \Psi_m\}$ (set of propositional sentences)

If potential secret Ψ_i is *true* in *db*, user may *not* learn this fact (if Ψ_i is *false* or *undef*, this fact may be disclosed)

Definition of confidentiality (informal)

- For any instance db_1, any security policy *pot_sec*,
  any potential secret $\Psi \in pot\_sec$,
  and any query sequence $Q = \langle \Phi_1, \ldots, \Phi_n \rangle$,
  there is an instance db_2
  (i) in which Ψ is **not** *true*,
  (ii) and under which the same answers are returned.
- To the user, db_1 and db_2 are indistinguishable

Preserving Confidentiality

Security policy *pot_sec* $= \{\Psi_1, \ldots, \Psi_m\}$ (set of propositional sentences)

If potential secret Ψ_i is *true* in *db*, user may *not* learn this fact
(if Ψ_i is *false* or *undef*, this fact may be disclosed)

Definition of confidentiality (informal)

- For any instance db_1, any security policy *pot_sec*,
  any potential secret $\Psi \in pot\_sec$,
  and any query sequence $Q = \langle \Phi_1, \ldots, \Phi_n \rangle$,
  there is an instance db_2
  (i) in which Ψ is **not** *true*,
  (ii) and under which the same answers are returned.
- To the user, db_1 and db_2 are indistinguishable

Definition does not state which techniques to use.

Answers and Inferences

Inference set V: Set of values the user regards as possible wrt. the actual query value (after having received an answer *ans*)

Examples:

$\{true\}$	"actual value is *true*"
$\{true, false\}$	"actual value is either *true* or *false* (but not *undef*)"

Answers and Inferences

Inference set V: Set of values the user regards as possible wrt. the actual query value (after having received an answer *ans*)

Examples:

$\{true\}$	"actual value is *true*"
$\{true, false\}$	"actual value is either *true* or *false* (but not *undef*)"

Inference set V from answer *ans*: two different approaches

1. "answer inferences": $V = \{ans\}$ (one element)
2. "meta inferences" based on user awareness of the censor: set of values $v \in \{true, false, undef\}$ that lead to the answer *ans* in the given situation (one, two or three elements)

Security Configurations

Censor considers which inferences are harmful.

Security Configurations

Censor considers which inferences are harmful.

Security configuration $C = \{ V \mid V \text{ is a "harmful" inference set} \}$

Example: $C = \{\{true\}, \{false\}, \{true, false\}\}$:

- user may not infer that the actual value is *true*
- user may not infer that the actual value is *false*
- user may not infer that the actual value is either *true* or *false*

Security Configurations

Censor considers which inferences are harmful.

Security configuration $C = \{ V \mid V \text{ is a "harmful" inference set} \}$

Example: $C = \{\{true\}, \{false\}, \{true, false\}\}$:

- user may not infer that the actual value is *true*
- user may not infer that the actual value is *false*
- user may not infer that the actual value is either *true* or *false*

Censor's decision based on ...

1. security configuration C
2. actual query value $v = eval(\Phi)(db)$

Answer generation: $ans := censor(C, v) \in \{true, false, undef, refuse\}$

Constructing Safe Censors

Uniform Lying

Preliminaries for uniform lying

- Method: only lying is allowed as a distortion method
- “answer inferences” (only consider unary inference sets)

Constructing Safe Censors

Uniform Lying

Preliminaries for uniform lying

- Method: only lying is allowed as a distortion method
- "answer inferences" (only consider unary inference sets)

Basic decision table (without distortion)

Security Configuration C	$eval(\Phi)(db) = \ldots$ *true*	*false*	*undef*
$\{\{true\},\{false\},\{undef\}\}$	*true*	*false*	*undef*
$\{\{true\},\{false\}\}$	*true*	*false*	*undef*
$\{\{true\},\{undef\}\}$	*true*	*false*	*undef*
$\{\{false\},\{undef\}\}$	*true*	*false*	*undef*
$\{\{true\}\}$	*true*	*false*	*undef*
$\{\{false\}\}$	*true*	*false*	*undef*
$\{\{undef\}\}$	*true*	*false*	*undef*
$\emptyset$	*true*	*false*	*undef*

Constructing Safe Censors

Situation 1: One "harmful" inference

Security Configuration C	$eval(\Phi)(db) = \ldots$ *true*	*false*	*undef*
$\{\{true\}\}$	*true*	*false*	*undef*

Constructing Safe Censors

Situation 1: One "harmful" inference

Security Configuration C	$eval(\Phi)(db) = \ldots$ *true*	*false*	*undef*
$\{\{true\}\}$	*undef*	*false*	*undef*

Change answer to any of the remaining two "safe" answers.

Constructing Safe Censors

Situation 1: One “harmful” inference

Security Configuration C	$eval(\Phi)(db) = \ldots$ *true*	*false*	*undef*
$\{\{true\}\}$	*undef*	*false*	*undef*

Change answer to any of the remaining two “safe” answers.

Situation 2: Two “harmful” inferences

$\{\{true\}, \{false\}\}$	*true*	*false*	*undef*

Constructing Safe Censors

Situation 1: One "harmful" inference

Security Configuration C	$eval(\Phi)(db) = \ldots$ *true*	*false*	*undef*
$\{\{true\}\}$	*undef*	*false*	*undef*

Change answer to any of the remaining two "safe" answers.

Situation 2: Two "harmful" inferences

$\{\{true\}, \{false\}\}$	*undef*	*undef*	*undef*

Change answer to the remaining "safe" one.

Constructing Safe Censors
Uniform Lying

Situation 3: Three "harmful" inferences

$\{\{true\},\{false\},\{undef\}\}$	*true*	*false*	*undef*

Constructing Safe Censors
Uniform Lying

Situation 3: Three "harmful" inferences

$\{\{true\}, \{false\}, \{undef\}\}$	*true*	*false*	*undef*

Problem: A "hopeless situation" for uniform lying

- Any answer leads to a "harmful" inference.
- No suitable lie may be safely given.

Constructing Safe Censors
Uniform Lying

Situation 3: Three "harmful" inferences

$\{\{true\},\{false\},\{undef\}\}$	*true*	*false*	*undef*

Problem: A "hopeless situation" for uniform lying

- Any answer leads to a "harmful" inference.
- No suitable lie may be safely given.

Two possible solutions

1. Avoid "hopeless situation" by a stronger definition of "harmfulness"
2. Allow refusal (leading to a *combined lying and refusal censor*):

$\{\{true\},\{false\},\{undef\}\}$	*refuse*	*refuse*	*refuse*

Constructing Safe Censors
Uniform Lying

A uniform lying censor

Security Configuration C	$eval(\Phi)(db) = \ldots$ true	false	undef
$\{\{true\}, \{false\}\}$	undef	undef	undef
$\{\{true\}, \{undef\}\}$	false	false	false
$\{\{false\}, \{undef\}\}$	true	true	true
$\{\{true\}\}$	undef	false	undef
$\{\{false\}\}$	true	undef	undef
$\{\{undef\}\}$	true	false	false
$\emptyset$	true	false	undef

Constructing Safe Censors
Uniform Lying

A uniform lying censor

Security Configuration C	$eval(\Phi)(db) = \ldots$ true	false	undef
$\{\{true\}, \{false\}\}$	*undef*	*undef*	*undef*
$\{\{true\}, \{undef\}\}$	*false*	*false*	*false*
$\{\{false\}, \{undef\}\}$	*true*	*true*	*true*
$\{\{true\}\}$	*undef*	*false*	*undef*
$\{\{false\}\}$	*true*	*undef*	*undef*
$\{\{undef\}\}$	*true*	*false*	*false*
$\emptyset$	*true*	*false*	*undef*

Censor is *locally safe* for a security configuration C if

for each $v \in \{true, false, undef\}$:

(a) [safe answers] $\{censor(C, v)\} \notin C$

(b) [only lie if necessary] if $\{v\} \notin C$ then $censor(C, v) = v$

Constructing Safe Censors

A combined lying and refusal censor

C	*true*	*false*	*undef*
$\{\{true\},\{false\},\{undef\}\}$	*refuse*	*refuse*	*refuse*
$\{\{true\},\{false\}\}$	*undef*	*undef*	*undef*
$\{\{true\},\{undef\}\}$	*false*	*false*	*false*
$\{\{false\},\{undef\}\}$	*true*	*true*	*true*
$\{\{true\}\}$	*undef*	*false*	*undef*
$\{\{false\}\}$	*true*	*undef*	*undef*
$\{\{undef\}\}$	*true*	*false*	*false*
$\emptyset$	*true*	*false*	*undef*

Constructing Safe Censors

A combined lying and refusal censor

C	*true*	*false*	*undef*
$\{\{true\}, \{false\}, \{undef\}\}$	*refuse*	*refuse*	*refuse*
$\{\{true\}, \{false\}\}$	*undef*	*undef*	*undef*
$\{\{true\}, \{undef\}\}$	*false*	*false*	*false*
$\{\{false\}, \{undef\}\}$	*true*	*true*	*true*
$\{\{true\}\}$	*undef*	*false*	*undef*
$\{\{false\}\}$	*true*	*undef*	*undef*
$\{\{undef\}\}$	*true*	*false*	*false*
$\emptyset$	*true*	*false*	*undef*

Censor is *locally safe* for a security configuration C if

(a) for each $v \in \{true, false, undef\}$: $\{censor(C, v)\} \notin C$

(b) for each $v \in \{true, false, undef\}$: if $\{v\} \notin C$ then $censor(C, v) = v$

(c) if $censor(C, v) = refuse$ for any v then $censor(C, v) = refuse$ for all v

Constructing Safe Censors
Uniform Refusal

Preliminaries for uniform refusal

- Policy: only refusal is allowed as a distortion method

Constructing Safe Censors
Uniform Refusal

Preliminaries for uniform refusal

- Policy: only refusal is allowed as a distortion method

Situation 1: One “harmful” inference

$\{\{true\}\}$	*true*	*false*	*undef*

Constructing Safe Censors
Uniform Refusal

Preliminaries for uniform refusal

- Policy: only refusal is allowed as a distortion method

Situation 1: One "harmful" inference

$\{\{true\}\}$	*refuse*	*false*	*undef*

Constructing Safe Censors
Uniform Refusal

Preliminaries for uniform refusal

- Policy: only refusal is allowed as a distortion method

Situation 1: One "harmful" inference

$\{\{true\}\}$	*refuse*	*false*	*undef*

Problem: meta inferences

- Two basic awareness assumptions
  1. user knows the security policy, thereby the security configuration
  2. user knows the algorithm of the censor
- user can infer: under the security configuration $\{\{true\}\}$, the answer *refuse* is only produced by the value *true*
- $\Rightarrow$ user can infer: $eval(\Phi)(db) = true$

Constructing Safe Censors

Uniform Refusal

Solution: additional *refuse*-conditions

Security Configuration C	$eval(\Phi)(db) = \ldots$ *true*	*false*	*undef*
$\{\{true\}\}$	*refuse*	*refuse*	*undef*

Meta inference now: "value is either *true* or *false*"
(which is not harmful – otherwise, $\{false\}$ would be as well!)

Constructing Safe Censors
Uniform Refusal

Situation 2: Two "harmful" inferences

$\{\{true\}, \{false\}\}$	*true*	*false*	*undef*

Constructing Safe Censors
Uniform Refusal

Situation 2: Two "harmful" inferences

$\{\{true\}, \{false\}\}$	*refuse*	*refuse*	*undef*

Constructing Safe Censors
Uniform Refusal

Situation 2: Two "harmful" inferences

$\{\{true\}, \{false\}\}$	*refuse*	*refuse*	*undef*

Problem: (partial) meta inferences

- User can infer: "value is either *true* or *false*" $\Rightarrow V = \{true, false\}$
- This partial inference may be harmful (but it may be not)

Constructing Safe Censors
Uniform Refusal

Solution: consider partial inferences, case differentiation

- check whether partial inference is harmful
- if so, add an additional *refuse*

Security Configuration C	$eval(\Phi)(db) = \ldots$ *true*	*false*	*undef*
$\{\{true\}, \{false\}\}$	*refuse*	*refuse*	*undef*
$\{\{true\}, \{false\}, \{true, false\}\}$	*refuse*	*refuse*	*refuse*

Constructing Safe Censors

A uniform refusal censor

C	*true*	*false*	*undef*
$\{\{true\}, \{false\}, \{undef\}, \ldots\}$	*refuse*	*refuse*	*refuse*
$\{\{true\}, \{false\}, \{true, false\}\}$	*refuse*	*refuse*	*refuse*
$\{\{true\}, \{false\}\}$	*refuse*	*refuse*	*undef*
$\{\{true\}, \{undef\}, \{true, undef\}\}$	*refuse*	*refuse*	*refuse*
$\{\{true\}, \{undef\}\}$	*refuse*	*false*	*refuse*
$\{\{true\}\}$	*refuse*	*false*	*refuse*
$\{\{false\}, \{undef\}, \{false, undef\}\}$	*refuse*	*refuse*	*refuse*
$\{\{false\}, \{undef\}\}$	*true*	*refuse*	*refuse*
$\{\{false\}\}$	*true*	*refuse*	*refuse*
$\{\{undef\}\}$	*true*	*refuse*	*refuse*
$\emptyset$	*true*	*false*	*undef*

Constructing Safe Censors

A uniform refusal censor

C	*true*	*false*	*undef*
$\{\{true\}, \{false\}, \{undef\}, \ldots\}$	*refuse*	*refuse*	*refuse*
$\{\{true\}, \{false\}, \{true, false\}\}$	*refuse*	*refuse*	*refuse*
$\{\{true\}, \{false\}\}$	*refuse*	*refuse*	*undef*
$\{\{true\}, \{undef\}, \{true, undef\}\}$	*refuse*	*refuse*	*refuse*
$\{\{true\}, \{undef\}\}$	*refuse*	*false*	*refuse*
$\{\{true\}\}$	*refuse*	*false*	*refuse*
$\{\{false\}, \{undef\}, \{false, undef\}\}$	*refuse*	*refuse*	*refuse*
$\{\{false\}, \{undef\}\}$	*true*	*refuse*	*refuse*
$\{\{false\}\}$	*true*	*refuse*	*refuse*
$\{\{undef\}\}$	*true*	*refuse*	*refuse*
$\emptyset$	*true*	*false*	*undef*

Censor is *locally safe* for a security configuration C if

$\forall ans \in \{true, false, undef, refuse\}$: meta inference from *ans* is not in C

Formalization

So far, we have ...

- heuristically constructed censors based on "security configurations"
- declarative semantics for security configurations

Formalization

So far, we have ...

- heuristically constructed censors based on “security configurations”
- declarative semantics for security configurations

But ...

- How to determine the set of “harmful” inferences?
- How to decide whether an inference is “harmful”?
- How to account for information gained from previous queries?

Formalization

So far, we have ...

- heuristically constructed censors based on "security configurations"
- declarative semantics for security configurations

But ...

- How to determine the set of "harmful" inferences?
- How to decide whether an inference is "harmful"?
- How to account for information gained from previous queries?

Solution: Epistemic logic

- representation of inferences as modal sentences
- store inferences in a *user log*
- note: epistemic logic only used as an auxiliary means

The User Log

Convert pair of query Φ and value v to epistemic sentence

- $\Delta(\Phi, \mathit{true}) = K\Phi$
- $\Delta(\Phi, \mathit{false}) = K\neg\Phi$
- $\Delta(\Phi, \mathit{undef}) = \neg K\Phi \wedge \neg K\neg\Phi$

The User Log

Convert pair of query Φ and value v to epistemic sentence

- $\Delta(\Phi, \mathit{true}) \quad = K\Phi$
- $\Delta(\Phi, \mathit{false}) \quad = K\neg\Phi$
- $\Delta(\Phi, \mathit{undef}) = \neg K\Phi \wedge \neg K\neg\Phi$

Convert pair of query Φ and inference set V to epistemic sentence

- $\Delta^*(\Phi, V) \quad = \bigvee_{v \in V} \Delta(\Phi, v)$

The User Log

Convert pair of query Φ and value v to epistemic sentence

- $\Delta(\Phi, \mathit{true}) = K\Phi$
- $\Delta(\Phi, \mathit{false}) = K\neg\Phi$
- $\Delta(\Phi, \mathit{undef}) = \neg K\Phi \wedge \neg K\neg\Phi$

Convert pair of query Φ and inference set V to epistemic sentence

- $\Delta^*(\Phi, V) = \bigvee_{v \in V} \Delta(\Phi, v)$

User log log_i: Representation of user's assumptions at time i

- log_0: a priori assumptions
- log_i: generated by adding inference set V_i from ans_i:
  $log_i = log_{i-1} \cup \{\Delta^*(\Phi_i, V_i)\}$ with $V_i = \mathit{inference}(\mathit{censor}, C_i, ans_i)$

The User Log (II)

Calculating the inference from an answer

Uniform lying and combined lying/refusal method: answer inferences

Take answer as inference, discard refusals:

$$\mathit{inference}^{ans}(\mathit{censor}, C, \mathit{ans}) \;:=\; \texttt{if } \mathit{ans} = \mathit{refuse} \texttt{ then } \emptyset \texttt{ else } \{\mathit{ans}\}$$

The User Log (II)

Calculating the inference from an answer

Uniform lying and combined lying/refusal method: answer inferences

Take answer as inference, discard refusals:

$$\mathit{inference}^{\mathit{ans}}(\mathit{censor}, C, \mathit{ans}) \;:=\; \texttt{if } \mathit{ans} = \mathit{refuse} \texttt{ then } \emptyset \texttt{ else } \{\mathit{ans}\}$$

Uniform refusal method: meta inferences

Calculate set of values that produce this answer under the given security configuration:

$$\mathit{inference}^{\mathit{meta}}(\mathit{censor}, C, \mathit{ans}) \;:=\; \{\ v \mid v \in \{\mathit{true}, \mathit{false}, \mathit{undef}\} \text{ and } \mathit{censor}(C, v) = \mathit{ans}\ \}$$

Security Violations

Uniform refusal and combined lying/refusal method

- user log is "violating" if it implies any of the potential secrets
- $\mathit{violates}^{\mathit{single}}(\mathit{pot\_sec}, \mathit{log}) \; := \; (\exists \Psi \in \mathit{pot\_sec})[\mathit{log} \models_{S5} \Psi]$

Security Violations

Uniform refusal and combined lying/refusal method

- user log is "violating" if it implies any of the potential secrets
- $\mathit{violates}^{\mathit{single}}(\mathit{pot\_sec}, \mathit{log}) \;:=\; (\exists \Psi \in \mathit{pot\_sec})[\mathit{log} \models_{S5} \Psi]$

Uniform lying method

- ... if it implies the disjunction of all potential secrets
- $\mathit{violates}^{\mathit{disj}}(\mathit{pot\_sec}, \mathit{log}) \;:=\; \mathit{log} \models_{S5} \bigvee_{\Psi \in \mathit{pot\_sec}} \Psi$
- stronger condition, but avoids $C^{\text{“hopeless”}} = \{\{\mathit{true}\}, \{\mathit{false}\}, \{\mathit{undef}\}\}$

Security Violations

Uniform refusal and combined lying/refusal method

- user log is "violating" if it implies any of the potential secrets
- $violates^{single}(pot\_sec, log) \;:=\; (\exists \Psi \in pot\_sec)[log \models_{S5} \Psi]$

Uniform lying method

- ... if it implies the disjunction of all potential secrets
- $violates^{disj}(pot\_sec, log) \;:=\; log \models_{S5} \bigvee_{\Psi \in pot\_sec} \Psi$
- stronger condition, but avoids $C^{\text{“hopeless”}} = \{\{true\}, \{false\}, \{undef\}\}$

Calculating the security configuration

$$C_i := sec\_conf(pot\_sec, log, \Phi) = \{\; V \mid V \in \mathcal{I} \text{ and } violates(pot\_sec, log \cup \{\Delta^*(\Phi, V)\}) \;\}$$

$\mathcal{I}$: range of the *inference*-function used by the respective method

Formalization and Security

Formalization of a method for controlled query evaluation

$control\_eval(\langle \Phi_1, .., \Phi_n \rangle, log_0, db, pot\_sec) := \langle (ans_1, log_1), .., (ans_n, log_n) \rangle$

Formalization and Security

Formalization of a method for controlled query evaluation

$$control\_eval(\langle \Phi_1, .., \Phi_n \rangle, log_0, db, pot\_sec) := \langle (ans_1, log_1), .., (ans_n, log_n) \rangle$$

In each step i:

1. $C_i := sec\_conf(pot\_sec, log_{i-1}, \Phi_i)$
2. $ans_i := censor(C_i, eval(\Phi_i)(db))$
3. $log_i := log_{i-1} \cup \{\Delta^*(\Phi_i, inference(censor, C_i, ans_i))\}$

Formalization and Security

Formalization of a method for controlled query evaluation

$control\_eval(\langle \Phi_1, .., \Phi_n \rangle, log_0, db, pot\_sec) := \langle (ans_1, log_1), .., (ans_n, log_n) \rangle$

In each step i:

1. $C_i \quad := sec\_conf(pot\_sec, log_{i-1}, \Phi_i)$
2. $ans_i := censor(C_i, eval(\Phi_i)(db))$
3. $log_i := log_{i-1} \cup \{\Delta^*(\Phi_i, inference(censor, C_i, ans_i))\}$

Theorem: *control_eval* preserves confidentiality if ...

1. $\neg violates(pot\_sec, log_0)$
2. censor function is *locally safe* for each security configuration C

Formalization and Security

Formalization of a method for controlled query evaluation

$$control\_eval(\langle \Phi_1, .., \Phi_n \rangle, log_0, db, pot\_sec) := \langle (ans_1, log_1), .., (ans_n, log_n) \rangle$$

In each step i:

1. $C_i \quad := sec\_conf(pot\_sec, log_{i-1}, \Phi_i)$
2. $ans_i := censor(C_i, eval(\Phi_i)(db))$
3. $log_i := log_{i-1} \cup \{\Delta^*(\Phi_i, inference(censor, C_i, ans_i))\}$

Theorem: *control_eval* preserves confidentiality if ...

1. $\neg violates(pot\_sec, log_0)$
2. censor function is *locally safe* for each security configuration C

Proof idea: $log_n \not\models_{S5} \Psi \;\Rightarrow\;$ ex. M, s : $(M, s) \models log_n$, $(M, s) \not\models \Psi$
$db_2 := \{\phi \mid \phi$ is propositional sentence and $(M, s) \models K\phi\}$

Methods – Comparison

Uniform lying

- always lie, never refuse
- pro: no need to consider meta inferences
- con: protects disjunction of secrets

Methods – Comparison

Uniform lying

- always lie, never refuse
- pro: no need to consider meta inferences
- con: protects disjunction of secrets

Uniform refusal

- always refuse, never lie
- pro: protects each single secret
- con: needs to consider meta inferences

Methods – Comparison

Uniform lying

- always lie, never refuse
- pro: no need to consider meta inferences
- con: protects disjunction of secrets

Uniform refusal

- always refuse, never lie
- pro: protects each single secret
- con: needs to consider meta inferences

Combined lying and refusal

- lie as long as possible, refuse in case of “hopeless situation”
- pro: protects each single secret
- pro: no need to consider meta inferences

Future Work

Consider . . .

- higher logics (FOL, . . .)
- different types of security policies
- open queries (paper in preparation)
- special cases (unknown security policies, . . .)

Find applications

- trust negotiation
- . . . ?

Implementation

- existing prototype for special case of complete information systems
- find suitable engine for modal logic