Consent & Access Control for Patient Information

Atif Khan
Ian McKillop

PhD Seminar

David R. Cheriton School of Computer Science
University of Waterloo

Jun 10th, 2013
Motivation

Patients are empowered to play an active role in the management of their medical information

- increasing awareness of individual privacy
- legislative-based measures
  - HIPAA – U.S. Health Insurance Portability and Accountability, 96
  - PIPEDA – Personal Information Protection and Electronic Documents Act, 00
  - PHIPA – Personal Health Information Protection Act, 04
  - EU-DPD – European Union Data Protection Directive, 95

- punish not prevent
Motivation

Patient information is

- dispersed over heterogeneous health information systems under the administration of different security domains
- managed by multiple owners
- dynamic circle-of-care membership with high churn
Motivation

Considerations

- privacy & security policies
- heterogeneous security models across different administrative domains
- multiple owners
- trusted links required for information exchange
Outline

Background
- challenges of consent application
- knowledge engineering

Proposed Model
- control primitives
- POC ontological model
- example scenarios
- handshake protocol

Conclusion
Outline

Background
- challenges of consent application
- knowledge engineering

Proposed Model
- control primitives
- POC ontological model
- example scenarios
- handshake protocol

Conclusion

House Keeping
- breakdown roughly 40/20
- questions
Outline

Background
- challenges of consent application
- knowledge engineering

Proposed Model
- control primitives
- POC ontological model
- example scenarios
- handshake protocol

Conclusion
Integrating Consent is Challenging

Precision vs. Expressivity

![Bar chart comparing natural languages and formal languages in terms of precision and expressivity.](image)
Integrating Consent is Challenging

Precision vs. Expressivity

- **high degree of precision** (of interpretation) of **consent** preferences is a **must**

- **precision** can be **enhanced** by use of **formal languages**

'**cost of precision**' exponentially grows with an increase in required expressiveness
Integrating Consent is Challenging

Precision vs. Expressivity

– *high degree* of *precision* (of interpretation) of *consent* preferences is a *must*

– *precision* can be *enhanced* by use of *formal languages*

*'cost of precision'* exponentially grows with an increase in required expressiveness

consent expression and its application requires maximizing both *expression* and *precision*
Integrating Consent is Challenging

Global Interpretation – Universal Semantics

– given the *variances* in data and *security primitives*, it is important to *establish* and *preserve* the *meaning* of *consent* rules and *policies*

*e.g., considering RBAC, different security domains can describe the same role “physician” with different levels of access privileges*
Integrating Consent is Challenging

Coverage

- given a finite expression space of a language, it is difficult to express consent for all possible scenarios

instead, it is preferable to have consent primitives offer transference properties
Integrating Consent is Challenging

Coverage

– transferring existing consent

\[ \text{applies}(c, s_1) \]
Integrating Consent is Challenging

Coverage

- transferring existing consent

\[
\text{applies}(c, s_1) \land f(s_1, s_2) > t
\]

threshold
similarity function
Integrating Consent is Challenging

Coverage

- transferring existing consent

\[
\text{applies}(c, s_1) \land f(s_1, s_2) > t \rightarrow \text{applies}(c, s_2)
\]
Integrating Consent is Challenging

Universal Enforcement

\[ R = \{ r_1, r_2, r_3, r_4, r_5, \ldots, r_k \} \land \text{belongsTo}(R, P) \]
Integrating Consent is Challenging

Universal Enforcement

\[ R = \{r_1, r_2, r_3, r_4, r_5, \ldots, r_k\} \land \text{belongsTo}(R, P) \]
Integrating Consent is Challenging

Universal Enforcement

\[ R = \{ r_1, r_2, r_3, r_4, r_5, \ldots, r_k \} \land belongsTo(R, P) \]

enforce \( C \) across \( D \), where \( D = \{ d_1, d_2, \ldots \} \)
Integrating Consent is Challenging

Enforcement Guarantees

- validation
  were the consent preferences applied properly?

- audit
  can it be confirmed for correctness?
Traditional Access Control Models

Limitations

- **enforcement across security domains**
  - requires pre-established trust relationship
  - predefined mapping of security models

- **multiple owners**
  - traditional models are *system-centric* → single owner where a *user-centric* approach is required
Traditional Access Control Models

Limitations

- traditional models result in implementations with static configurations
  - access control parameters/policies are predefined
  - leads to *breaking-of-the-glass* scenarios
Outline

Background
- challenges of consent application
- knowledge engineering

Proposed Model
- control primitives
- POC ontological model
- example scenarios
- handshake protocol

Conclusion
Ontology

an ontology is a formal, explicit specification of a shared conceptualization (R. Struder 98)

- a 'domain-of-discourse' is described using ontological concepts and the relationships between these concepts

- each ontological concept and relationship is unique (precision of interpretation = very high)

- the expressivity of an ontology ≡ construction (concepts and relationships linking the concepts)
Semantic Knowledge Representation

Ontology

- AccessControl
- Resource
- MedicalRecord
- Patient
Semantic Knowledge Representation

Ontology

- **AccessControl**
- **Resource**
- **MedicalRecord**
- **Patient**

Relationships:
- MedicalRecord → hasPatient → Patient
- MedicalRecord → primaryOwner → Patient
- Resource → isa → MedicalRecord
- AccessControl → requires → Resource
Ontology

Let $\mathcal{V}$ be a set of structured vocabulary, and $\mathcal{A}$ axioms about $\mathcal{V}$, which are formulated in a formal language $\mathcal{L}$. Then an ontology $\mathcal{O}$ is a sign-system:

$$\mathcal{O} = \langle \mathcal{L}, \mathcal{V}, \mathcal{A} \rangle$$

(Hussain 09)
Knowledge Inference

Inference

- Axioms are defined to infer *implicit knowledge* from *explicitly* stated *facts*.

- Axiom classes *(not discussed)*.

- Entailment rules *(in N3)*.
Knowledge Inference

Inference

- axioms are defined to infer *implicit knowledge* from *explicitly* stated facts

- entailment rules

\[
\{ f_1, f_2, f_3, \cdots, f_n \} \rightarrow \{ a_1, a_2, \cdots \}
\]

\[
\downarrow
\]

\[
\{ f_1 \wedge f_2 \wedge f_3 \wedge \cdots \wedge f_n \} \quad \{ a_1 \wedge a_2 \cdots \}
\]

monotonic process
Knowledge Inference

Inference

- axioms are defined to infer *implicit knowledge* from *explicitly* stated *facts*

- entailment rules *(in N3)*

```plaintext
triple representation: subject verb object.

```{?R :isa :MedicalRecord; :hasPolicy ?POL.
    ?POL :hasScope :OptIn; :hasOverride :None.

} => { ?DR :hasAccess ?R}
```
Knowledge Inference

Inference

- axioms are defined to infer \textit{implicit knowledge} from \textit{explicitly} stated \textit{facts}

- entailment rules \textit{(in N3)}

\[
\{ \textcolor{green}{?R : isa :MedicalRecord; :hasPolicy} \textcolor{red}{?POL}. \\
\textcolor{red}{?POL : hasScope :OptIn; :hasOverride :None.} \\
\textcolor{green}{?DR : isa :Doctor.} \} \implies \{ \textcolor{red}{?DR : hasAccess} ?R\}
\]
Knowledge Inference

Inference

- axioms are defined to infer *implicit knowledge* from *explicitly* stated *facts*

- entailment rules (*in N3*)

\[
\{ ?R :isa :MedicalRecord; :hasPolicy ?POL. \\
?POL :hasScope :OptIn; :hasOverride :None. \\
?DR :isa :Doctor. \\
\} \Rightarrow \{ ?DR :hasAccess ?R \}
\]

form an ontology

generic rules written using variables
Knowledge Reasoning

Reasoning Process

- discover new facts by applying inference rules to available *triples* to answer the query
- based on *first order logic*
- requires a *semantic reasoner*
Knowledge Reasoning

- triplestore
- reasoner
- query
- inference rules
- Result
- Proof
Knowledge Reasoning

- query
- triplestore
- reasoner
- inference rules

Query answer
Result
Proof

* logic-based,
* can be verified
Outline

Background
- challenges of consent application
- knowledge engineering

Proposed Model
- control primitives
- $POC$ ontological model
- example scenarios
- handshake protocol

Conclusion
Control Primitives

*define the building blocks of the proposed access control model*
Control Primitives

Resource

resource represents any \textit{entity} for which access control is required

\[ \forall t : \text{requires}(t, \text{AccessControl}) \rightarrow \text{resource}(t) \]

e.g.
medical records
diagnostic images
medical procedures
Control Primitives

Resource

resource represents any "entity" for which access control is required

- a resource can be identified by
  - a specific resource identifier (such as record number)
  - a logical grouping of things

\[ R_M = \{ r \mid r \text{ is a mental health record} \} \]
Control Primitives

Resource

resource represents any “entity” for which access control is required

- a resource can be identified by
  - a specific resource identifier (such as record number)
  - a logical grouping of things

\[
R_M = \{ r \mid r \text{ is a mental health record} \}
\]

this can be statically defined or can be inferred dynamically

composite resource

concept hierarchy along isa relationship

rule-based inference
Control Primitives

Resource

resource represents any “entity” for which access control is required

resource provenance

\[ \text{created}(r, d) \implies \text{originatesFrom}(r, d) \]

a resource may be transferred over

\[ \text{created}(r, d_l) \land \text{transferedTo}(r, d_l, d_h) \implies \text{owner}(d_h, r) \land \text{originatesFrom}(r, d_l) \]
Owner

an *entity* that is *allowed* to define rules for a resource

\[ \text{canDefine}(e, r, p) \rightarrow \text{owner}(e, r) \]

where:
- \(e\) is an entity
- \(r\) is a resource
- \(p\) is a policy
Control Primitives

Owner

an *entity* that is *allowed* to define rules for a resource

\[
\text{canDefine}(e, r, p) \rightarrow \text{owner}(e, r)
\]

– multiple *(resource)* owners
  
  • patients are primary owners of their information
  
  • all other owners are considered secondary owners
Control Primitives

Domain

an *administrative abstraction* for an entity governing a set of resources

- example,
  - a hospital, or a family physician’s office

- a *domain* is an *implied owner* for all resources that originate within its administrative control
Policy

a set of rules that must be fulfilled to grant access to a resource

- coarse-grained policy expressions
  - all treating physicians have access to my medical records
  - no one has access unless there is a life threatening emergency

- fine-grained policy expressions
  - Dr. Smith cannot access my mental health records
Policy

a set of rules that must be fulfilled to grant access to a resource

- linking policies to resources

\[\text{hasPolicy}(r, p)\] where:
- \(r\) is a resource
- \(p\) is a policy

\[\text{hasPolicy}(R, P)\] where:
- \(R\) is a set of resources
- \(P\) is a set of policies
Policy

a set of rules that must be fulfilled to grant access to a resource

– linking policies to resources

\[
\text{hasPolicy}(r, p) \quad \text{hasPolicy}(R, P)
\]

\[
P = \{c_{\text{pat}}, p_{\text{institutional}}, p_{\text{provincial}}\}
\]

\[
R = \{r \mid r \text{ is a mental health record}\}
\]
Control Primitives

Policy

- a set of rules that must be fulfilled to grant access to a resource

- policy conflict resolution
  - resources are protected by policies defined by multiple owners

\[
\mathbf{f}_{\text{resolve}} : p_i \circ p_{i+1} \circ \cdots \circ p_{i+k} \rightarrow p_{\text{effective}}
\]
Policy

- a set of rules that must be fulfilled to grant access to a resource

- policy conflict resolution
  - resources are protected by policies defined by multiple owners

\[
f_{\text{resolve}}: p_i \circ p_{i+1} \circ \cdots \circ p_{i+k} \rightarrow p_{\text{effective}}
\]
Policy

a *set of rules* that *must be fulfilled* to grant access to a resource

- policy *conflict resolution*
  - *resources* are protected by *policies* defined by multiple *owners*

\[
f_{\text{resolve}} : C_{\text{patient}} \lor \left( P_{\text{hospital}} \land P_{\text{province}} \right) \Rightarrow P_{\text{effective}}\]
Outline

Background
– challenges of consent application
– knowledge engineering

Proposed Model
– control primitives
– POC ontological model
– example scenarios
– handshake protocol

Conclusion
A POC Access Control Ontology

\[ O = \langle \mathcal{L}, \mathcal{V}, \mathcal{A} \rangle \]

- choosing \( \mathcal{L} \)
  - high degree of expressivity and precision
  - computational completeness
    
    all decisions are guaranteed to be computable
  - decidability
    
    all computations will finish in finite time
A POC Access Control Ontology

\[ O = \langle \mathcal{L}, \mathcal{V}, \mathcal{A} \rangle \]

- choosing \( \mathcal{L} = \text{OWL-DL} \)

web ontology language (OWL)

- formal semantics
- machine processable serialization
- based on description logic (DL)
A POC Access Control Ontology

\[ O = \langle \mathcal{L}, \mathcal{V}, \mathcal{A} \rangle \]

choosing \( \mathcal{V} \)

*any structured vocabulary can be utilized as long as monotonic reasoning is not violated*
AccessControl

Custodian

Hospital

Doctor

MedicalRecord

Policy

Patient

Resource

hasAccess

isa

requires

originated

employer

primaryOwner

hasScope

hasOverride

hasPolicy

hasPatient

hasPolicy

hasPolicy

hasPolicy

hasPolicy

isa

isa

isa

isa

isa

isa

OptOut

OptIn

None

SituationOverride

EntityOverride

hasScope

hasOverride
A POC Access Control Ontology

\[ O=\langle \mathcal{L}, \mathcal{V}, \mathcal{A} \rangle \]

- defining \( \mathcal{A} \)
  
  - all access control rules defined as entailment rules using \( \mathcal{V} \)

- consent preferences expressed as a policy
  
  - a policy is a set of access control rules
Outline

Background
- challenges of consent application
- knowledge engineering

Proposed Model
- control primitives
- POC ontological model
- example scenarios
- handshake protocol

Conclusion
Proposed Model

Example Scenarios

- medical records as resources requiring access control
- discovering resource owners
- OptIn
- OptIn & OptOut with conditions
- consent transitivity
- validation of system made decisions
- *breaking-of-the-glass*
Example *Domain of Discourse*

1: $H1 \ a \ :\text{Hospital}$.

2: $Dr1 \ a \ :\text{Doctor}; \ :employer \ :H1$.

3: $Dr2 \ a \ :\text{Doctor}; \ :employer \ :H1$.

4: $P1 \ a \ :\text{Patient}$.

5: $R1 \ a \ :\text{MedicalRecord}$;
   $\ :hasPatient \ :P1; \ :originated \ :H1$.

6: $P2 \ a \ :\text{Patient}$.

7: $R2 \ a \ :\text{MedicalRecord}$;
   $\ :hasPatient \ :P2; \ :originated \ :H1$. 
Example *Domain of Discourse*

**Rule:**
anything that is a medical record requires access control

```prolog
1 {?R a :MedicalRecord}
   =>{?R :requires :AccessControl}.
```
Example Domain of Discourse

**Rule:**
anything that is a medical record requires access control

1 {?R a :MedicalRecord}
   => {?R :requires :AccessControl}.

**Query**
what entities require access control?

1 :R1 :requires :AccessControl.
2 :R2 :requires :AccessControl.
Resource Owners

1 \{?H a :Hospital.\} => \{?H a :Custodian\}.

2

3 \{?C a :Custodian.
   ?R a :MedicalRecord.

4

5 \{?P a :Patient.
   ?R a :MedicalRecord.
   ?R :hasPatient ?P \} => \{?P :primaryOwner ?R\}.

6

7 \{?X :primaryOwner ?Y\}=>\{?X :owner ?Y\}.

8

9 \{?X :owner ?Y\}=>\{?Y :isOwnedBy ?X\}. 
Resource Owners

1. `{?H a :Hospital.} => {?H a :Custodian}.`
2. `{?C a :Custodian.
   ?R a :MedicalRecord.
4. `{?P a :Patient.
   ?R a :MedicalRecord.
6. `{?X :primaryOwner ?Y} => {?X :owner ?Y}.`
8. `{?X :owner ?Y} => {?Y :isOwnedBy ?X}.`
Resource Owners

1 {?H a :Hospital.} => {?H a :Custodian}.

2

3 {?C a :Custodian.  
   ?R a :MedicalRecord.  

4

5 {?P a :Patient.  
   ?R a :MedicalRecord.  

6

7 {?X :primaryOwner ?Y} => {?X :owner ?Y}.

8

9 {?X :owner ?Y} => {?Y :isOwnedBy ?X}.
Resource Owners

1 { ?H a :Hospital. } => { ?H a :Custodian }.

2

3 { ?C a :Custodian. 
   ?R a :MedicalRecord. 

4

5 { ?P a :Patient. 
   ?R a :MedicalRecord. 

6


8

Resource Owners

1: $P1 : primaryOwner : R1; : owner : R1.$
2: $P2 : primaryOwner : R2; : owner : R2.$
3: $H1 : owner : R1, : R2.$
4: $R1 : isOwnedBy : P1, : H1.$
5: $R2 : isOwnedBy : P2, : H2.$
Consent Expression

- Policy
  - hasScope
    - ConsentScope
      - isa
        - OptOut
        - isa
          - OptIn

OptOut
share nothing

OptIn
share all
OptIn Consent

1. C1 a :Policy;  
   :hasScope :OptIn; :hasOverride :None.

2. P1 :hasPolicy :C1.

1. {?P :owner ?R; :hasPolicy ?P.}  
   => {?R :hasPolicy ?P}.

Consent Transference: if a patient has a policy, then a record has the same policy by default if owned by the patient.
1 C1 a :Policy;
   :hasScope :OptIn; :hasOverride :None.

2 :P1 :hasPolicy :C1.
3 :P2 :hasPolicy :C1.

1 {?P :owner ?R; :hasPolicy ?P.} => {?R :hasPolicy ?P}. 

1 {?R a :MedicalRecord; :hasPolicy ?POL.
   ?POL :hasScope :OptIn; :hasOverride :None.
   ?DR a :Doctor.
} => {?DR :hasAccess ?R}. 

OptIn Consent
OptIn Consent

1: P1 :hasPolicy :C1. :R1 :hasPolicy :C1.
2: P2 :hasPolicy :C1. :R2 :hasPolicy :C1.
3
4: Dr1 :hasAccess :R1,:R2.
5: Dr2 :hasAccess :R1,:R2.
Consent Expression

Policy

hasScope

ConsentScope

hasOverride

Override

isa

OptOut

isa

OptIn

isa

None

isa

SituationOverride

isa

EntityOverride

OptOut
situation override

OptIn
entity override

OptOut
share nothing

OptIn
share all

Jun 10, 2013

CBAC
OptIn/OptOut with Overrides

1  :C2 :hasScope :OptOut; :hasOverride :LifeEmergency.
2  :C3 :hasScope :OptIn; :hasOverride :Dr2.
3
4  :P1 :hasPolicy :C2.
5  :P2 :hasPolicy :C3.
OptIn with Entity Override

1 { ?R a :MedicalRecord; :hasPolicy ?POL.  
2 ?POL :hasScope :OptIn;  
   :hasOverride ?DR_NOT_ALLOWED.  
3  
4 ?DR_NOT_ALLOWED a :Doctor.  
5 ?DR a :Doctor.  
6 ?DR :notEqualTo ?DR_NOT_ALLOWED.  
7  
8 } => {?DR :hasAccess ?R}.  

OptOut with Situation Override

1 \{ ?R a :MedicalRecord; :hasPolicy ?POL.
2  ?POL :hasScope :OptOut;
   :hasOverride :LifeEmergency.
3
4  ?P a :Patient; :owner ?R;
   :hasCondition :LifeEmergency.
5
6  ?DR a :Doctor.
7 \} => {?DR :hasAccess ?R}. 
OptOut with Situation Override


4: Dr1 : hasAccess : R2.

Observations

– no one has access to medical record R1
– Dr2 does not have access to medical record R2
### OptOut with Situation Override

1. \(P_1 : \text{hasCondition} : \text{LifeEmergency}.\)

2. \(P_1 : \text{hasPolicy} : C_2. \) \(R_1 : \text{hasPolicy} : C_2.\)

3. \(P_2 : \text{hasPolicy} : C_3. \) \(R_2 : \text{hasPolicy} : C_3.\)

4. \(\text{Dr}_1 : \text{hasAccess} : R_1, : R_2.\)

5. \(\text{Dr}_2 : \text{hasAccess} : R_1.\)

### Observations

- Dr2 has access to medical record R1
  but does not have access to medical record R2
Consent Transitivity

Extending existing consent

1. :C4 a :Policy;
   :hasScope :OptOut; :hasOverride :P1CareTeam.

2. :P1 :hasPolicy :C4.;
   :careTeam :P1CareTeam.

3. :Dr1 :isMemeber :P1CareTeam.
Consent Transitivity

Extending existing consent

1 \{\text{?D1} a :Doctor. \text{?D2} a :Doctor.\}
2 \text{?D1} :\text{consults} [:with \text{?D2}; :about \text{?P}].
3 \text{?P} :\text{careTeam} \text{?CT}.
4 \} \Rightarrow \{\text{?D2} :\text{isMemeber} \text{?CT}\}.
Consent Transitivity

Extending existing consent

1 {?D1 a :Doctor. ?D2 a :Doctor.
2   ?D1 :consults [:with ?D2; :about ?P].
3   ?P :careTeam ?CT.
4 } => {?D2 :isMemeber ?CT}.

1 {?R a :MedicalRecord; :hasPolicy ?POL.
2   ?POL :hasScope :OptOut; :hasOverride ?CT.
3   ?DR a :Doctor; :isMemeber ?CT.
4 } => {?DR :hasAccess ?R}. 
Consent Transitivity

Extending Existing Consent

1. :C4 a :Policy;
   :hasScope :OptOut; :hasOverride :P1CareTeam.

2. :P1 :hasPolicy :C4.;
   :careTeam :P1CareTeam.

3. :Dr1 :isMember :P1CareTeam.

1. :Dr1 :hasAccess :R1.
2. :Dr2 :hasAccess :R2.
Trusting System Made Decisions

Proof-of-Correctness

- triplestore
- query
- reasoner
- inference rules
- Result
- Proof

query answer

* logic-based,
* can be verified
Proof-of-Correctness

- semantic proof

access control (inference) rules are applied against the knowledge-base graph to find evidence (sub-graphs) towards fulfilment of the query → semantic proofs
Proof-of-Correctness

- semantic proof

  access control (inference) rules are applied against the knowledge-base graph to find evidence (sub-graphs) towards fulfilment of the query → semantic proofs

- validation

  traverse the sub-graphs (semantic proofs) to decide if the same decision can be reached
Recall

1 \{ ?R a :MedicalRecord; :hasPolicy ?POL.
2   ?POL :hasScope :OptIn;
   :hasOverride ?DR_NOT_ALLOWED.
3
4   ?DR_NOT_ALLOWED a :Doctor.
5   ?DR a :Doctor.
6   ?DR :notEqualTo ?DR_NOT_ALLOWED.
7
8 } => {?DR :hasAccess ?R}.
Trusting System Made Decisions

1. \{\{R2 a :MedicalRecord\} e:evidence <kb.n3#_36>. 
2. \{\{\{P2 a :Patient\} e:evidence <kb.n3#_35>. 
3. \{R2 a :MedicalRecord\} e:evidence <kb.n3#_36>. 
4. \{R2 :hasPatient :P2\} e:evidence <kb.n3#_36> \}
5. \Rightarrow \{\{P2 :primaryOwner :R2\} e:evidence <kb.n3#_42>\} \}
6. \Rightarrow \{\{P2 :owner :R2\} e:evidence <kb.n3#_43>\}. 
7. \{P2 :hasPolicy :C3\} e:evidence <kb.n3#_35> \}
8. \Rightarrow \{\{R2 :hasPolicy :C3\} e:evidence <kb.n3#_48>\}. 
9. \{C3 :hasScope :OptIn\} e:evidence <kb.n3#_22>. 
10. \{C3 :hasOverride :Dr2\} e:evidence <kb.n3#_22>. 
11. \{Dr2 a :Doctor\} e:evidence <kb.n3#_27>. 
12. \{Dr1 a :Doctor\} e:evidence <kb.n3#_26>. 
13. \{Dr1 :notEqualTo :Dr2\} e:evidence <log#kb> \}
14. \Rightarrow \{\{Dr1 :hasAccess :R2\} e:evidence <kb.n3#_59>\}. 

Observations

– semantic proofs are also represented in *triple* format
  \[ \therefore \text{can also be reasoned about} \rightarrow \text{automated validation} \]

– semantic proofs provide built-in auditing mechanism
'Breaking-of-the-Glass'

Definition

when it is necessary to **ignore policy** in the interest of **patient safety**

- a **common practice**
  due to limitations of traditional access control models
'Breaking-of-the-Glass'

Definition

*when it is necessary to ignore policy in the interest of patient safety*

- using consent transference
  - determine if existing consent policy can be applied
  - *breaking-of-the-glass* is exercised only when absolutely necessary
'Breaking-of-the-Glass'

Definition

when it is necessary to **ignore policy** in the interest of **patient safety**

- using semantic proofs
  - extend semantic proofs in support of *breaking-of-the-glass* decisions
  - provides logic-based reasons for “why”
Outline

Background
– challenges of consent application
– knowledge engineering

Proposed Model
– control primitives
– POC ontological model
– example scenarios
– handshake protocol

Conclusion
Handshake Protocol

Purpose

establish if a requester is allowed to receive patient information

- is not a communication protocol
  - can be integrated into existing protocols as a “pre-step”
Handshake Protocol

Purpose

establish if a requester is allowed to receive patient information

- 3 Phases Protocol
  - request for information
  - proof generation
  - proof validation
Handshake Protocol

\[ D^{isTreating} \rightarrow P \]

\[
\begin{align*}
Pol_{H2} & \quad R = \{r_1, r_2, \cdots\} \\
Pol_C & \quad P
\end{align*}
\]
**Handshake Protocol**

\[
D \xrightarrow{\text{isTreating}} P
\]

**Stage 1:** request for information

\[
M_{\text{req}}(P, f_1(\cdots) \rightarrow R \mid R, \text{session}_{ID})
\]

resolution function

\[
\begin{align*}
\text{Pol}_{H2} & \quad R = \{r_1, r_2, \cdots\} \\
\text{Pol}_{C} &
\end{align*}
\]
Handshake Protocol

Stage 1: request for information

\[ M_{req}(P, f_1(\cdots) \rightarrow R | R, session_{ID}) \]

\[ \text{isTreating} \]

\[ D \rightarrow P \]

\[ \begin{cases} \text{Pol}_{H2} \\ \text{Pol}_{C} \end{cases} \]

\[ R = \{ r_1, r_2, \cdots \} \]

\[ \text{Pol}_R = \{ \text{Pol}_{H2}, \text{Pol}_{C} \} \]
Handshake Protocol

Stage 1: request for information

\[ M_{req}(P, f_1(\ldots) \rightarrow R | R, session_{ID}) \]

Stage 2: proof generation

\[ Q = \{ Proof(p) \mid p \in Pol_R \} \]
Handshake Protocol

Stage 1: request for information
\[ M_{\text{req}}(P, f_1(\cdots) \rightarrow R | R, \text{session}_{ID}) \]

Stage 2: proof generation
\[ Q = \{\text{Proof}(p) | p \in \text{Pol}_R\} \]

\[ M_{\text{res}}(Q, \text{session}_{ID}) \]
Handshake Protocol

Stage 1: request for information
\[ M_{req}(P, f_1(\cdots) \rightarrow R \mid R, session_{ID}) \]

Stage 2: proof generation
\[ Q = \{ \text{Proof}(p) \mid p \in Pol_R \} \]
\[ M_{res}(Q, session_{ID}) \]

Stage 3: proof validation
\[ \text{validate}(Q) \begin{cases} 0 & \text{terminate} \\ 1 & \text{continue} \end{cases} \]
Handshake Protocol
Future Extensions

Trust

– utilizing semantic proofs to dynamically establish trust

Privacy

– validation of semantic proofs requires access to raw information
  
  • utilize cryptographic primitives for proof generation
    zero knowledge proof, cryptographic commitments, oblivious transfer

Model as MAS (*multi agent system*)

– offload proof generation & validation to agents
Outline

Background
- challenges of consent application
- knowledge engineering

Proposed Model
- control primitives
- POC ontological model
- example scenarios
- handshake protocol

Conclusion
Conclusion

Access Control Model

- *resource, owner, policy, domain*
  - structured knowledge representation (ontology)
  - logic-based reasoning/inference

- offers high degree of expression & precision

- suitable for healthcare domain
  - multiple owners
  - heterogeneous administrative domains
  - built-in validation
Conclusion

Realization

– a simple proof-of-concept ontology
– applied to core healthcare scenarios

Integration

– *hand shake* protocol
Thank You!