A Logic Specification for Usage Control

Xinwen Zhang, Jaehong Park
Francesco Parisi-Presicce, Ravi Sandhu

George Mason University
SACMAT 2004

Outline

• Introduction of UCON
• Temporal Logic of Action (TLA)
• Logic Model for UCON with TLA
• Specification of Authorization Core Models
• Obligation and Conditions
• Conclusions and Future Work
UCON

- A unified framework for next generation access control
- A comprehensive model to represent the underlying mechanism of existing access control models and policies.
- Try to extend the limits of traditional access control models:
  - Authorization only – No obligation or condition based control
  - Identity based only – No attributes based support
  - Decision is made before access – No ongoing control
  - No consumable rights - No mutable attributes
  - Rights are pre-defined and granted to subjects

UCON

- UCON provides a general model beyond DRM and Trust management:
  - Digital Rights Management (DRM)
    • Mainly focus on intellectual property rights protection with architecture and mechanism level studies
    • Lack of access control model
  - Trust Management
    • Authorization for strangers’ access based on credentials
    • Lack of an abstract model with attribute-based.
OM-AM Layered Approach

<table>
<thead>
<tr>
<th>What?</th>
<th>Objective</th>
<th>Policy Neutral</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Model</td>
<td>UCON_{ABC} model</td>
</tr>
<tr>
<td></td>
<td>Architecture</td>
<td>CRM/SRM, CDID architectures</td>
</tr>
<tr>
<td></td>
<td>Mechanism</td>
<td>DRM technologies, Trusted computing, etc.</td>
</tr>
</tbody>
</table>

OM-AM Framework | Usage Control System | Assurance

- Model examples: Access Matrix, Lattice-based model, Role-base access control model

UCON Model

- Basic components:
  - Subjects and attributes
  - Objects and attributes
  - Rights
- Logically, UCON is a mapping from a set of \{subject/object attributes, right\} to \{true, false\}
- Usage control decisions are based on authorization, obligations, and conditions.
- Referred as UCON_{ABC} model
Continuity and Mutability of UCON

- A single usage process has three phases
  - before access, during usage, and access
- Continuity: control decision can be checked before or during access
- Mutability: attribute updates can be performed before, during or after access
  - Pre-update, on-update, and post-update

### Core Authorization Models

- According to the authorization control attribute update points, we have seven core authorization models:
  - $\text{pre}A_0$: control decision is determined before access, and there is no attribute update.
  - $\text{pre}A_1$: control decision and attribute update before access.
  - $\text{pre}A_2$: control decision is determined before access, and attribute update after access.
  - $\text{on}A_0$: control decision is checked and determined during usage, and there is no attribute update.
  - $\text{on}A_1$: control decision is checked and determined during usage, and there is attribute update before access.
  - $\text{on}A_2$: control decision is checked and determined during usage, and there is attribute update during usage.
  - $\text{on}A_3$: control decision is checked and determined during usage, and there is attribute update after usage.
- A real UCON system may be a hybrid of them.
Outline

- Introduction of UCON
- Temporal Logic of Action (TLA)
- Logic Model for UCON with TLA
- Specification of Authorization Core Models in UCON
- Obligation and Conditions
- Conclusions and Future Work

Temporal Logic of Action

- Basic Terms:
  - Variables: x, y
  - Values: 5, “abc”
  - Constants
  - A state is an assignment of values to variables
- Functions: nonboolean expression with variables and constants
  - Semantically, a function is a mapping from states to values.
- State Predicates: boolean expression with variables and constants
  - Semantically, a predicate is a mapping from states to booleans.
- Actions: boolean expression with variables, primed variables, and constants
  - Semantically, an action is a function assigning a boolean to a pair of states \( (s, t) \), where \( s \) is the old state with variables, and \( t \) is the new state with primed variables.
TLA

- **Behavior**: a sequence of states 
  \(<s_0, s_1, s_2, \ldots,>\)

- **Semantics of an action** \(A\):
  \(<s_0, s_1, s_2, \ldots> [A] \equiv s_0[A]s_1\)

- **Temporal operator**: (always)
  \(<s_0, s_1, s_2, \ldots> [\square A] \equiv \forall n \in N : s_n[A]s_{n+1}\)

- **Temporal Formula**:
  \(F \equiv <predicate \triangleright [\square <action > |F |F \land F |F |F |F |F \rightarrow F |F |F\)

- **Semantics**:
  \(<s_0, s_1, s_2, \ldots> [F] \equiv s_0[F]s_1\)
  \(<s_0, s_1, s_2, \ldots> [\square F] \equiv \forall n \in N : <s_n, s_{n+1}, s_{n+2}, \ldots> [F]\)

TLA

- **Other temporal operators**:
  - “Eventually”:
    \(<s_0, s_1, s_2, \ldots> [\diamondsuit F] \equiv \exists n \in N : <s_n, s_{n+1}, s_{n+2}, \ldots> [F]\)
    \(\diamondsuit F \equiv \neg \square \neg F\)
  - “Next”:
    \(<s_0, s_1, s_2, \ldots> [\diamond F] \equiv s_1[F]s_2\)
  - “Until”:
    \(<s_0, s_1, s_2, \ldots> [F \triangleright G] \equiv \exists i \geq 0 : (s_i[G]s_{i+1} \land (0 \leq j \leq i \rightarrow s_j[F]s_{j+1}))\)

- **Past temporal operators**:
  - *Has-always-been, Once, Previous, Since*
    \(<\ldots, s_{n-2}, s_{n-1}, s_0, s_1, s_2, \ldots> [\blacksquare F] \equiv \forall n < 0 : s_n[F]s_{n+1}\)
    \(<\ldots, s_{n-2}, s_{n-1}, s_0, s_1, s_2, \ldots> [\diamondsuit F] \equiv \exists n < 0 : s_n[F]s_{n+1}\)
    \(<\ldots, s_{n-2}, s_{n-1}, s_0, s_1, s_2, \ldots> [\circ F] \equiv s_{n-1}[F]s_n\)
    \(<\ldots, s_{n-2}, s_{n-1}, s_0, s_1, s_2, \ldots> [F \ast G] \equiv \exists i < 0 : (s_i[G]s_{i+1} \land (i < j < 0 \rightarrow s_j[F]s_{j+1}))\)
Outline

- Introduction of UCON
- Temporal Logic of Action (TLA)
- Logic Model for UCON with TLA
- Specification of Authorization Core Models in UCON
- Obligation and Conditions
- Conclusions and Future Work

Logical Model of UCON: Attributes

- A state of UCON is an assignment of values to attributes:
  - Subject attributes: role, security clearance, credit amount, etc.
  - Object attributes: type, directory, etc.
  - System attributes: time, location, etc.
  - A special system attribute:
    - $\text{state}(s,o,r) = \{\text{initial, requesting, denied, accessing, revoked, end}\}$
    - To specify the status of a single access process $(s,o,r)$
    - Authorization actions defined to change this state.
Logical Model of UCON: Predicates

- Predicates: boolean expression built from subject attributes, object attributes, and system attributes:
  - Unary predicates:
    - Alice.credit > $1000, file1.classification = “secure”
  - Binary predicates:
    - Dominate(Alice.clearance, file1.classification)
    - in((Bob, read), file2.ACL)
  - Ternary predicate permit(s,o,r):
    - usage control decision
    - True if s is allowed to access o with r.

Logic Model of UCON: Actions

- Two types of actions:
  - Actions performed by a subject
  - Actions performed by the system

- state(s,o,r) transition with actions:
Logic Model of UCON

**Definition 1.** A logical formula in UCON is defined by the following grammar in BNF:

\[ Φ ::= a[p(t_1, ..., t_n)]((-Φ) | (\langle Φ \land Φ \rangle | (Φ \to Φ)) \forall x : Φ | \exists x : Φ | a] [α | Φ | <Φ | □Φ | ◇Φ | Φ[αSΦ] \]

where \( a \) is an action, \( p \) is a predicate of arity \( n \), \( t_1, ..., t_n \) are terms, and \( x \) is a variable.

**Definition 2.** An logic authorization model of UCON is a triple \( (S, P, A) \), where

- \( S \) is a sequence of states of subject, object, and the system attributes,
- \( P \) is a finite set of state predicates on subject and/or object attributes,
- \( A \) is a finite set of state actions.

---

Logical Model of UCON

If a model \( M \) with a state \( s \) satisfies a formula \( Φ \), we write \( M, s \models Φ \). Semantically,

1. \( M, s_0 \models p \) iff \( s_0[p] \), where \( p \in P \).
2. \( M, s_0 \models \neg Φ \) iff \( M, s_1 \not\models Φ \), where \( s_1 \) is next state of \( s \) in \( S \).
3. \( M, s_0 \models Φ[α] \) iff \( M, s_0 \models Φ \).
4. \( M, s_0 \models Φ \lor Φ_2 \) iff \( M, s_0 \not\models Φ_2 \) and \( M, s_0 \not\models Φ \).
5. \( M, s_0 \models Φ \land Φ_2 \) iff \( M, s_0 \models Φ \) or \( M, s_0 \not\models Φ_2 \).
6. \( M, s_0 \models \forall x : Φ \) iff for all \( a \), \( M, a \not\models Φ[x/a] \).
7. \( M, s_0 \models ∃x : Φ \) iff for some \( a \), \( M, a \not\models Φ[x/a] \).
8. \( M, s_0 \models Φ \lor \exists x \geq 0 : M, s_0 \not\models Φ \).
9. \( M, s_0 \models Φ \land \exists x \geq 0 : M, s_0 \not\models Φ \).
10. \( M, s_0 \models □Φ \) iff \( M, s_0 \models Φ \).
11. \( M, s_0 \models □Φ \lor Φ_2 \) iff \( M, s_0 \not\models Φ_2 \) and \( (0 \leq j \leq i \rightarrow M, s_j \models Φ) \).
12. \( M, s_0 \models ◇Φ \lor Φ_2 \) iff \( M, s_0 \not\models Φ_2 \) and \( (i \leq j \leq 0 \rightarrow M, s_j \models Φ) \).
13. \( M, s_0 \models Φ \lor Φ_2 \) iff \( M, s_0 \not\models Φ \) and \( (i \leq j \leq 0 \rightarrow M, s_j \models Φ_2) \).
Outline

- Introduction of UCON
- Temporal Logic of Action (TLA)
- Logic Model for UCON with TLA
- Specification of Authorization Core Models
- Obligation and Conditions
- Conclusions and Future Work

Specification of Core Models

- $preA_0$:
  
  $p_1 \land \ldots \land p_i \rightarrow \text{permit}(s,o,r)$
  
  $\text{tryaccess}(s,o,r) \land \text{permit}(s,o,r) \rightarrow \Box(\text{permitaccess}(s,o,r))$

- Example 2: BLP model
  
  $\text{dominate}(s,\text{clearance},o,\text{classification}) \rightarrow \text{permit}(s,o,\text{read})$
  
  $\text{tryaccess}(s,o,\text{read}) \land \text{permit}(s,o,\text{read}) \rightarrow \Box(\text{permitaccess}(s,o,\text{read}))$
  
  $\text{dominate}(o,\text{classification},s,\text{clearance}) \rightarrow \text{permit}(s,o,\text{write})$
  
  $\text{tryaccess}(s,o,\text{write}) \land \text{permit}(s,o,\text{write}) \rightarrow \Box(\text{permitaccess}(s,o,\text{write}))$

- Example 3: DAC with ACL
  
  $\text{in}(s,ID,r),o,acl) \rightarrow \text{permit}(s,o,r)$
  
  $\text{tryaccess}(s,o,r) \land \text{permit}(s,o,r) \rightarrow \Box(\text{permitaccess}(s,o,r))$
Specification of Core Models

- \textit{preA}_j:

\[
p_1 \land \ldots \land p_i \rightarrow \text{permit}(s, o, r) \\
\text{permitAccess}(s, o, r) \rightarrow \Box(\text{tryAccess}(s, o, r) \land \text{permit}(s, o, r) \land \Diamond(\text{preupdate}(\text{attribute})))
\]

- Example 4: DRM pay-per-use application

\[
(Alice.\text{credit} \geq \text{ebook1.value}) \rightarrow \text{permit}(\text{Alice, ebook1, read}) \\
\text{permitAccess}(\text{Alice, ebook1, read}) \rightarrow \Box(\text{tryAccess}(\text{Alice, ebook1, read}) \\
\land \Diamond(\text{preupdate}(\text{Alice.credit}))) \land \text{permit}(\text{Alice, ebook1, read}) \\
\text{preupdate}: \text{Alice.\text{credit}}' = \text{Alice.\text{credit}} - \text{ebook1.value}
\]

\[
\]

Specification of Core Models

- \textit{preA}_3:

\[
p_1 \land \ldots \land p_i \rightarrow \text{permit}(s, o, r) \\
\text{permitAccess}(s, o, r) \rightarrow \Box(\text{tryAccess}(s, o, r) \land \text{permit}(s, o, r) \\
\land \Diamond(\text{postupdate}(\text{attribute})))
\]

\textbf{Example 5} DRM membership-based application:

- subject: Alice, with attributes of \textit{ID} and total \textit{expense}
- object: book1, with attributes of \textit{title} and \textit{readingCost}
- subject: readingGroup, with attribute \textit{readerList} = \{ID1, ID2, \ldots\} and \textit{bookList} = \\
  \{book1.title, book2.title, \ldots\}
- right:read

\[
in(\text{Alice, readingGroup.readerList}) \land in(\text{book1.title, readingGroup.bookList}) \rightarrow \\
\text{permit}(\text{Alice, book1, read}) \\
\text{permitAccess}(\text{Alice, book1, read}) \rightarrow \Box(\text{tryAccess}(\text{Alice, book1, read}) \land \\
\text{permit}(\text{Alice, book1, read}) \\
\text{endAccess}(\text{Alice, book1, read}) \rightarrow \Diamond(\text{postupdate}(\text{Alice.expense})) \\
\text{postupdate}: \text{Alice.expense}' = \text{Alice.expense} + \text{ebook1.readingCost}
\]
Specification of Core Models

• onA₀:
  \( \Box(\neg(p₁ \land \ldots \land pₙ) \land (\text{state}(s, o, r) = \text{accessing}) \rightarrow \Diamond(\text{revokeaccess}(s, o, r))) \)

• Example 6:
  \( \Box(\neg(\text{Bob}.\text{role} = \text{employee}) \land (\text{Bob}.\text{temp.cert} \in \text{RCL}) \land (\text{state}(\text{Bob}, o, r) = \text{accessing}) \rightarrow \Diamond(\text{revokeaccess}(\text{Bob}, o, r))) \)

Specification of Core Models

• onA₁:
  \( \text{permitaccess}(s, o, r) \rightarrow \Diamond(\text{tryaccess}(s, o, r) \land \Diamond(\text{preupdate(attribute)))) \)
  \( \Box(\neg(p₁ \land \ldots \land pₙ) \land (\text{state}(s, o, r) = \text{accessing}) \rightarrow \Diamond(\text{revokeaccess}(s, o, r))) \)

• onA₂:
  \( \Box(\neg(p₁ \land \ldots \land pₙ) \land (\text{state}(s, o, r) = \text{accessing}) \rightarrow \Diamond(\text{revokeaccess}(s, o, r))) \)
  \( \text{endaccess}(s, o, r) \land \text{revokeaccess}(s, o, r) \rightarrow \Diamond(\text{permitaccess}(s, o, r) \land \Diamond(\text{onupdate(attribute)))) \)

• onA₃:
  \( \Box(\neg(p₁ \land \ldots \land pₙ) \land (\text{state}(s, o, r) = \text{accessing}) \rightarrow \Diamond(\text{revokeaccess}(s, o, r))) \)
  \( \text{endaccess}(s, o, r) \land \text{revokeaccess}(s, o, r) \rightarrow \Diamond(\text{postupdate(attribute)}) \)
Specification: an Example

• Example 7: Resource-constrained access control
  – Limited number (10) of ongoing accessing for a single object
  – Object attribute: accessingS = \{s|s is accessing o\}
  – When 11th subject requesting new access, one ongoing accessing subject will be revoked.

• a. revocation by earliest usage will be revoked
  • Subject attribute: startTime

\begin{align*}
(1) \ \text{true} & \rightarrow \text{permit}(s, o, r) \\
(2) \ \text{permitAccess}(s, o, r) & \rightarrow \left(\text{preupdate}(o, \text{accessingS})\right), \text{where preupdate : o, accessingS}' = o, \text{accessingS} + \{s\} \\
(3) \ \text{tryaccess}(x, o, r) & \wedge (x \notin o, \text{accessingS}) \wedge (|o, \text{accessingS}| = 10) \wedge (s \in o, \text{accessingS}) \wedge (s, \text{startTime} = \text{Min}(o, \text{accessingS})) \rightarrow \bigcirc(\text{revokeaccess}(s, o, r)) \\
(4) \ \text{endaccess}(s, o, r) \wedge \text{revokeaccess}(s, o, r) & \rightarrow \bigcirc(\text{postUpdate}(o, \text{accessingS})), \text{where postUpdate : o, accessingS}' = o, \text{accessingS} - \{s\}
\end{align*}

• b. revocation by longest idle usage
  • Subject attribute: idleTime

\begin{align*}
(1) \ \text{true} & \rightarrow \text{permit}(s, o, r) \\
(2) \ \text{permitAccess}(s, o, r) & \rightarrow \left(\text{preupdate}(o, \text{accessingS})\right), \text{where preupdate : o, accessingS}' = o, \text{accessingS} + \{s\} \\
(3) \ \text{tryaccess}(x, o, r) & \wedge (x \notin o, \text{accessingS}) \wedge (|o, \text{accessingS}| = 10) \wedge (s \in o, \text{accessingS}) \wedge (s, \text{idleTime} = \text{Max}(o, \text{accessingS})) \rightarrow \bigcirc(\text{revokeaccess}(s, o, r))
\end{align*}

• c. revocation by longest total usage
  • Subject attribute: usageTime

\begin{align*}
(1) \ \text{true} & \rightarrow \text{permit}(s, o, r) \\
(2) \ \text{permitAccess}(s, o, r) & \rightarrow \left(\text{preupdate}(o, \text{accessingS})\right), \text{where preupdate : o, accessingS}' = o, \text{accessingS} + \{s\} \\
(3) \ \text{tryaccess}(x, o, r) & \wedge (x \notin o, \text{accessingS}) \wedge (|o, \text{accessingS}| = 10) \wedge (s \in o, \text{accessingS}) \wedge (s, \text{usageTime} = \text{Max}(o, \text{accessingS})) \rightarrow \bigcirc(\text{revokeaccess}(s, o, r)) \\
(4) \ \text{endaccess}(s, o, r) \wedge \text{revokeaccess}(s, o, r) & \rightarrow \bigcirc(\text{postUpdate}(\text{usageTime})) \wedge \bigcirc(\text{postUpdate}(\text{accessingS})), \text{where postUpdate : o, accessingS}' = o, \text{accessingS} - \{s\}
\end{align*}
Outline

- Introduction of UCON
- Temporal Logic of Action (TLA)
- Logic Model for UCON with TLA
- Specification of Authorization Core Models
- Obligation and Conditions
- Conclusions and Future Work

Obligations

- Two types of obligations in UCON:
  - pre-obligations, which must have been performed before access.
  - ongoing-obligations, which must be performed during usage.

Definition 3 An obligation is an action described by:
\[ a_b(s, o, r, s_b, o_b, r_b, para_1, ..., para_i, ...) \]
where \( a_b \) is the obligation name, \((s, o, r)\) is a particular usage process requiring the obligation, \( s_b, o_b, r_b \) are obligation subject, object and right, \( para_1, ..., para_i \) are optional parameters.

Definition 4 A logical model of UCON with authorizations and obligations is a 4-tuple:
\[ M = (S, P, A_A, A_B) \]
where \( S \) is a sequence of states, \( P \) is a finite set of predicates, \( A_A \) is a finite set of authorization actions (same as the \( A \) in the authorization model), \( A_B \) is a finite set of obligation actions.
Obligations

- Example: click license agreement before making order:

\[(s.\text{role} = \text{registered}) \rightarrow \text{permit}(s, \text{o}, \text{order})\]
\[\text{permit}(s, \text{o}, \text{order}) \land \neg \text{(click.agreement}(s, \text{o}, \text{order, s, agree.statement, click}))\]
\[\rightarrow \text{permitaccess}(s, \text{o}, \text{order})\]

Conditions

- Conditions are environment restrictions before or during usage.
- In UCON, a condition is a predicate built from system attributes, such as time and location.

**Definition 5** A logical model of UCON with authorizations, obligations, and conditions is a 5-tuple:

\[M = (S, P_A, P_C, A_A, A_B)\]

where \(S\), \(A_A\), and \(A_B\) are the same as before, \(P_A\) is a finite set of authorization predicates (the \(P\) before), and \(P_C\) is a finite set of condition predicates.

- Example:

\[(s.\text{role} = \text{dayshifter}) \land (8am \leq \text{currentT} \leq 5pm) \rightarrow \text{permitaccess}(s, \text{o}, r)\]
\[(s.\text{role} = \text{nightshifter}) \land \neg (8am \leq \text{currentT} \leq 5pm) \rightarrow \text{permitaccess}(s, \text{o}, r)\]
Outline

- Introduction of UCON
- Temporal Logic of Action (TLA)
- Logic Model for UCON with TLA
- Specification of Authorization Core Models
- Obligation and Conditions
- Conclusions and Future Work

Conclusions

- A logical model for UCON with:
  - States with:
    - subject attributes and values
    - Object attributes and values
    - System attribute and values
  - Predicates:
    - Authorization predicates built from subject and object attributes
    - Condition predicates built from system attributes
  - Actions:
    - Attribute update actions
    - Usage control actions
    - Obligation actions
  - Temporal formulas of usage control policies
- First-order logic specification of the UCON models with new features of:
  - Mutability
  - Continuality
Future Work

• **UCON:**
  – Enrich UCON model, such as constraints, delegations
  – Administrative UCON model
    • Attribute management
    • Administrative policies
  – Expressive power and safety analysis for UCON
  – Concurrency of UCON

• Development of architecture and mechanism for UCON system