

# TRBAC: A Temporal Role-Based Access Control Model

---

*Elisa Bertino*

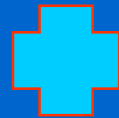
*CERIAS and CS Department*

*Purdue University*

# What is TRBAC?

---



RBAC Model [Sandhu 98]



Temporal constraints on role  
activations/deactivations

# What is TRBAC?

---

- ◆ An active role is a role that a user can activate during a session (that is, the user can acquire the role's)
- ◆ A role can be active in certain time periods and non active in other:
  - ⇒ Role activation: non active  active
  - ⇒ Role deactivation: active  non active

# Why TRBAC?

---

- ◆ Often roles are characterized by a temporal dimension :
  - ⇒ Job functions may have limited or periodic time duration
  - ⇒ There may be activation dependencies among roles

# TRBAC: Main Features

---

- ◆ Periodic activations/deactivations of roles
- ◆ Temporal dependencies among role activations/deactivations



ROLE TRIGGERS

# TRBAC: Main Features

---

- ◆ Role triggers may cause either:
  - ⇒ Immediate activations/deactivations, or
  - ⇒ Deferred activations/deactivations
- ◆ Run-time requests to dynamically change the status of a role

# TRBAC: Main Features

---

- ◆ Priorities for:
  - ⇒ Periodic activations/deactivations
  - ⇒ Role triggers
  - ⇒ Runt-time requests
- ◆ Priorities are used for conflict resolution

# TRBAC: Periodic Events

## Definition: (Periodic Event)

A periodic event is a tuple  $(\mathbf{I}, \mathbf{P}, \mathbf{p:E})$  where  $\mathbf{I}$  is a time interval,  $\mathbf{P}$  is a periodic expression,  $\mathbf{p:E}$  is a prioritized event expression,  $\mathbf{E} \in \{\text{activate } R, \text{deactivate } R\}$ ,  $R \in \text{Roles}$

$([7/1/00, 12/31/00], \text{night-time}, \text{VH: activate, doctor-on-night-duty})$

$([7/1/00, 12/31/00], \text{day-time}, \text{VH: deactivate, doctor-on-night-duty})$

# TRBAC: Role Triggers

## Definition: (Role Trigger)

Role triggers are of the form:

$$E_1, \dots, E_n, C_1, \dots, C_k \longrightarrow p:E \text{ after } \Delta t$$

where  $E_i$ 's are event expressions,  $E_i \in \{\text{activate } R, \text{ deactivate } R\}$ ,  $C_j$ 's are role status expressions,  $C_j \in \{\text{active } R, \text{ not active } R\}$ ,  $R \in \text{Roles}$ ,  $p:E$  is a prioritized event expression and  $\Delta t$  is a temporal displacement

# Role Triggers: Example

---

activate doctor-on-night-duty → VH: activate nurse-on-night-duty

activate nurse-on-day-duty → H: activate nurse-on-training after 2 Hours

# Role Activation Base

**RAB = Periodic Events + Role Triggers**

([1/1/00,12/31/00], night-time, VH:activate doctor-on-night-duty)

([1/1/00,12/31/00], day-time, VH:activate doctor-on-day-duty)

([1/1/00,12/31/00], night-time, VH:deactivate doctor-on-day-duty)

([1/1/00,12/31/00], night-time, VH:deactivate doctor-on-night-duty)

activate doctor-on-night-duty → H: activate nurse-on-night-duty

deactivate doctor-on-night-duty → H: deactivate nurse-on-night-duty

activate doctor-on-day-duty → H: activate nurse-on-day-duty

deactivate doctor-on-day-duty → H: deactivate nurse-on-day-duty

activate nurse-on-day-duty → H: activate nurse-on-training after 2 Hours

deactivate nurse-on-day-duty → VH: deactivate nurse-on-training

# TRBAC: Runtime Request Expressions

---

## Definition: (Runtime Request Expression)

A runtime request expression has the form:

**$p:E$  after  $\Delta t$**

where  $p:E$  is a prioritized event expression and  $\Delta t$  is a temporal displacement

deactivate nurse-on-training after 2 Hours

activate emergency-doctor

# TRBAC: Formal Aspects

---

- ◆ The *Execution Model* of a RAB specifies, for each instant  $t$ , the set of events that should occur at time  $t$  according to:
  - ⇒ periodic events & triggers in the RAB
  - ⇒ runtime request expressions
  - ⇒ priorities

# TRBAC: Formal Aspects

---

- ◆ Some specifications may yield no execution model, while some ambiguous specifications may admit two or more models

activate R  $\longrightarrow$  deactivate S

activate S  $\longrightarrow$  deactivate R

Requests: activate R, activate S

# TRBAC: Formal Aspects

---

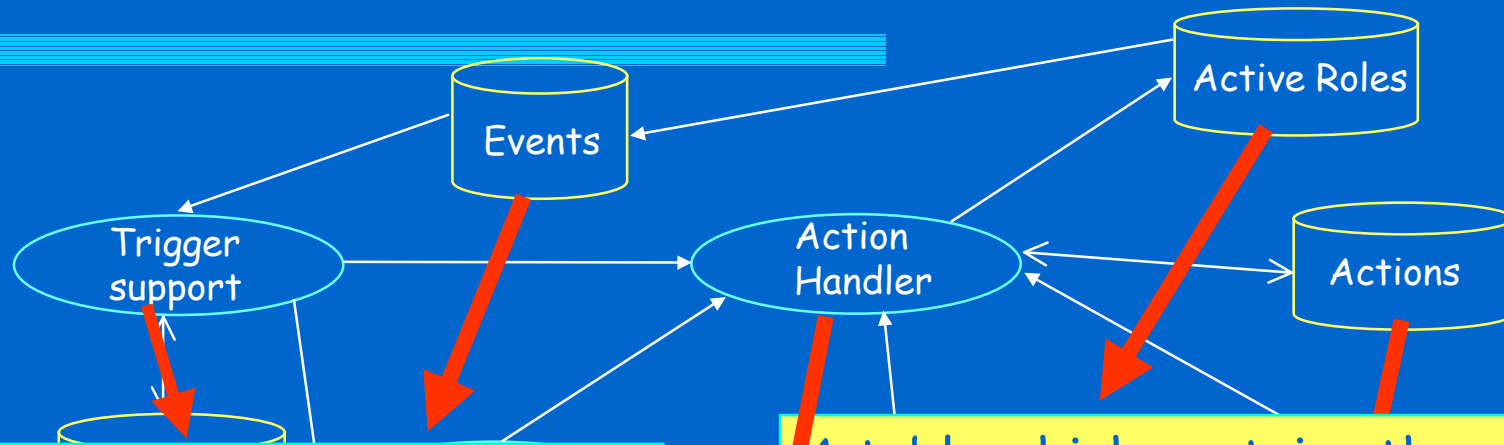
- ◆ Safeness condition that guarantees that a given RAB has exactly one model
- ◆ It exploits the notion of dependency graph
  - ⇒ No cycles involving conflicting events
- ◆ Safeness check is polynomial in the RAB dimension

# TRBAC: Architectural Aspects

---

- ◆ At each time it is necessary to know which are the active roles, on the basis of the RAB and runtime requests
- ◆ A request by a user to activate a role is authorized if:
  - ⇒ The user has the authorization to play that role
  - ⇒ The role is active at the time of the request

# A Possible Architecture



It is a global event base which registers the activations/deactivations of triggers. The activations/deactivations are instantaneous, and are caused by the trigger.

A table which contains the roles that can be activated

priorities. If the activation is caused by the trigger, it is instantaneous, and is caused by the trigger.

It is in charge of updating table Active\_Roles according to the requests of the other modules. It uses table Actions to solve potential conflicts

contains to be e on s for e

A table which contains specified triggers. If a trigger is deferred, it is inserted into Deferred\_Actions

Deferred\_Actions

It is the safe action into Deferred\_Actions

it is the

# Generalized TRBAC (GTRBAC)

---

## ◆ Motivations:

- ⇒ TRBAC does not distinguish between a role being *enabled* and a role being *active*
- ⇒ A role is enabled if the temporal conditions associated with it are satisfied
- ⇒ A role is *active* if a user has logged in the role
- ⇒ Only enabled roles can be activated
- ⇒ Because of such limitations, TRBAC cannot support some forms of constraints, such as the maximum number of activations of a role by a user in a given time interval

# GTRBAC

---

- ◆ GTRBAC extends TRBAC by introducing temporal conditions on:
  - ⇒ User-role assignments
  - ⇒ Role-permission assignments
- ◆ A large number of constraints can thus be supported

# GTRBAC – Examples of Constraints

---

- ◆ Constraints on the number of concurrent activations
  - ⇒ “there can be at most 10 users activating the role DayDoctor at a time”
- ◆ Constraints on the number of total activations in a given period
  - ⇒ “the role HeadNurse can be activated at most 2 times per day”

# X-GTRBAC - Motivations

---

- ◆ Role Based Access Control Model
  - ⇒ Many benefits over traditional access control models when applied to emerging applications
- ◆ XML is a uniform platform for information interchange

Our Goal  
XML + RBAC extension  
To provide access control framework for Web-  
Services environments

---

# X-GTRBAC - why XML?

---

XML - main benefits:

- ◆ Uniform, vendor-neutral representation of enterprise data
- ◆ Mechanism for interchange of information across heterogeneous systems
- ◆ Extensible syntax and semantics
- ◆ Widespread support from main platforms and tool vendors

# X-RBAC Language

---

## ◆ Modeling RBAC Elements

### ◆ Users

- credential types

XML User Sheet (XUS)

XML CredType Definition

### ◆ Roles

- separation of duty
- temporal constraints
- triggers

XML Role Sheet (XRS)

XML SoD Definition

XML TempConst Definition

XML Trigger Definition

### ◆ Permissions

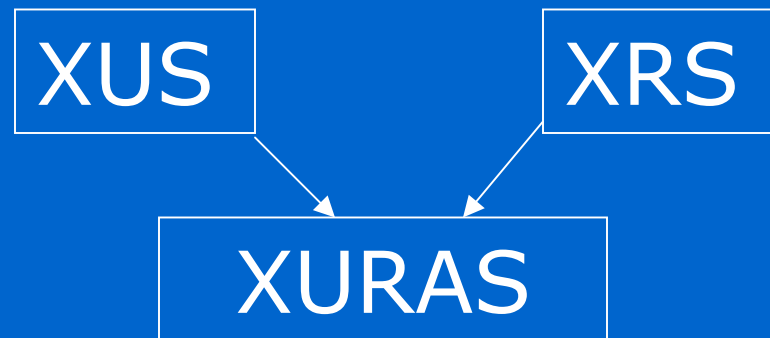
XML Permission Sheet (XPS)

---

# X-RBAC Language

---

- ◆ Policy Administration
  - ◆ User-to-Role Assignment

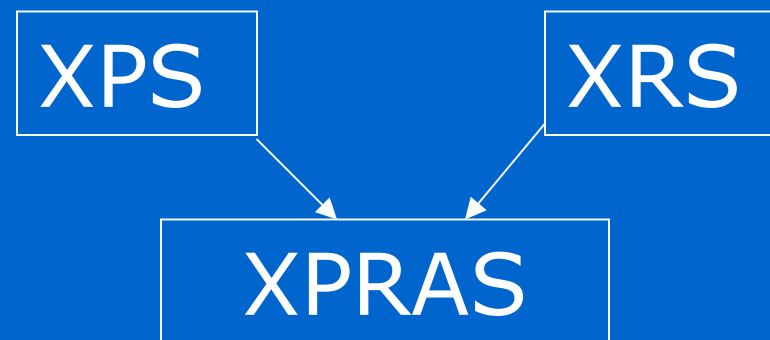


XML User-to-Role Assignment Sheet  
(XURAS)

# X-RBAC Language

---

- ◆ Policy Administration
  - ◆ Permission-to-Role Assignment



XML Permission-to-Role Assignment Sheet  
(XPRAS)

# XUS Grammar

```
</XUS>  
  <!-- User Definitions >  
</XUS>  
  
<!-- User Definitions > ::=  
<Users> {<!-- User Definition>}+  
</Users>
```

```
<!-- User Definition> ::=  
<User user_id = (id)>  
  <UserName> (name) </UserName>  
  {<!--CredType>}+  
  <MaxRoles>(number)</MaxRoles>  
</User>
```

```
<!--CredType > ::=  
<CredType cred_type_id =(id)>  
  <type_name> (name)</type_name>  
  <!-- Credential Expression>  
</CredType>
```

```
<!-- Credential Expression> ::=  
<CredExpr>  
  {<(attribute name)> (attribute value)  
  </(attribute name)> }+  
</CredExpr>
```

# An XML instance of XUS

```
<XUS>
  <User user_id="j1">
    <UserName >John</ UserName >
    <CredType cred_type_id ="C100">
      < type_name >Nurse</type_name>
      <CredExpr>
        <age> 30 </age>
        <field> ophthalmology </field>
        <level> 5 </level>
        <status> single </status>
      </CredExpr>
    </CredType>
    < MaxRoles>2</MaxRoles>
  </User >
  <User > ... </User >
```

# XRS Grammar

```
<!-- XML Role Sheet> ::=  
<XRS [xrs_id = (id) ]>  
  {<!-- Role Definitions>}+  
</XRS>
```

```
<!-- Role Definitions> ::=  
<Roles>  
  <Role role_id = (id)  
    <RoleName> (role name)> <RoleName>  
    [<!--{En|Dis}abling Constraint>]  
    [<!--[De]Activation Constraint>]  
    {<SSDRoleSetID> (id) </SSDRoleSetID>}*  
    {<DSDRoleSetID> (id) </DSDRoleSetID>}*  
    {<Junior> (name) </Junior>}*  
    {<Senior> (name) </Senior>}*  
    [<Cardinality>(number)</Cardinality>]  
  </Role>  
  <Role> .. </Role>  
  ..  
</Roles>
```

# An XML instance of XRS

```
XRS>
<Roles >
  <Role role_id = "R100">
    <RoleName> Nurse </ RoleName >
    <Senior> Eye_Doctor </ Senior>
    <Cardinality> 8 </ Cardinality >
  </Role>
  <Role role_id = "R200">
    <RoleName> Eye_Doctor </RoleName>
    < DSDRoleSetID>DSD1</ DSDRoleSetID >
    < Junior>Nurse</ Junior>
    <Senior> Eye_Surgeon </Senior>
    <Cardinality> 6 </Cardinality>
  </Role>
</Roles>
</XRS >
```

# XPS Grammar

---

```
<!-- XML Permission Sheet> ::=  
<XPS [xps_id = (id) ]>  
  {<!-- Permission Definitions>}+  
</XPS>
```

```
<!-- Permission Definitions> ::=  
<Permission perm_id = id  
  [prop= (prop op)] >  
  <Object type=(type name) id=(id) />  
  <Operation> (access op) </Operation>  
</Permission>
```

# An XML instance of XPS

```
XPS>
  <Permission perm_id ="P1">
    <Object type = "Schema" id = "XS101" />
    <Operation> all</operation>
  </Permission >
  <Permission perm_id ="P2">
    <Object type = "Instance" id = "XI100" />
    <Operation> all</operation>
  </Permission >
  <Permission perm_id ="P3">
    <Object type = "Element" id = "XE100" />
    <Operation> navigate </operation>
  </Permission >
</XPS>
```

# Example of XURAS

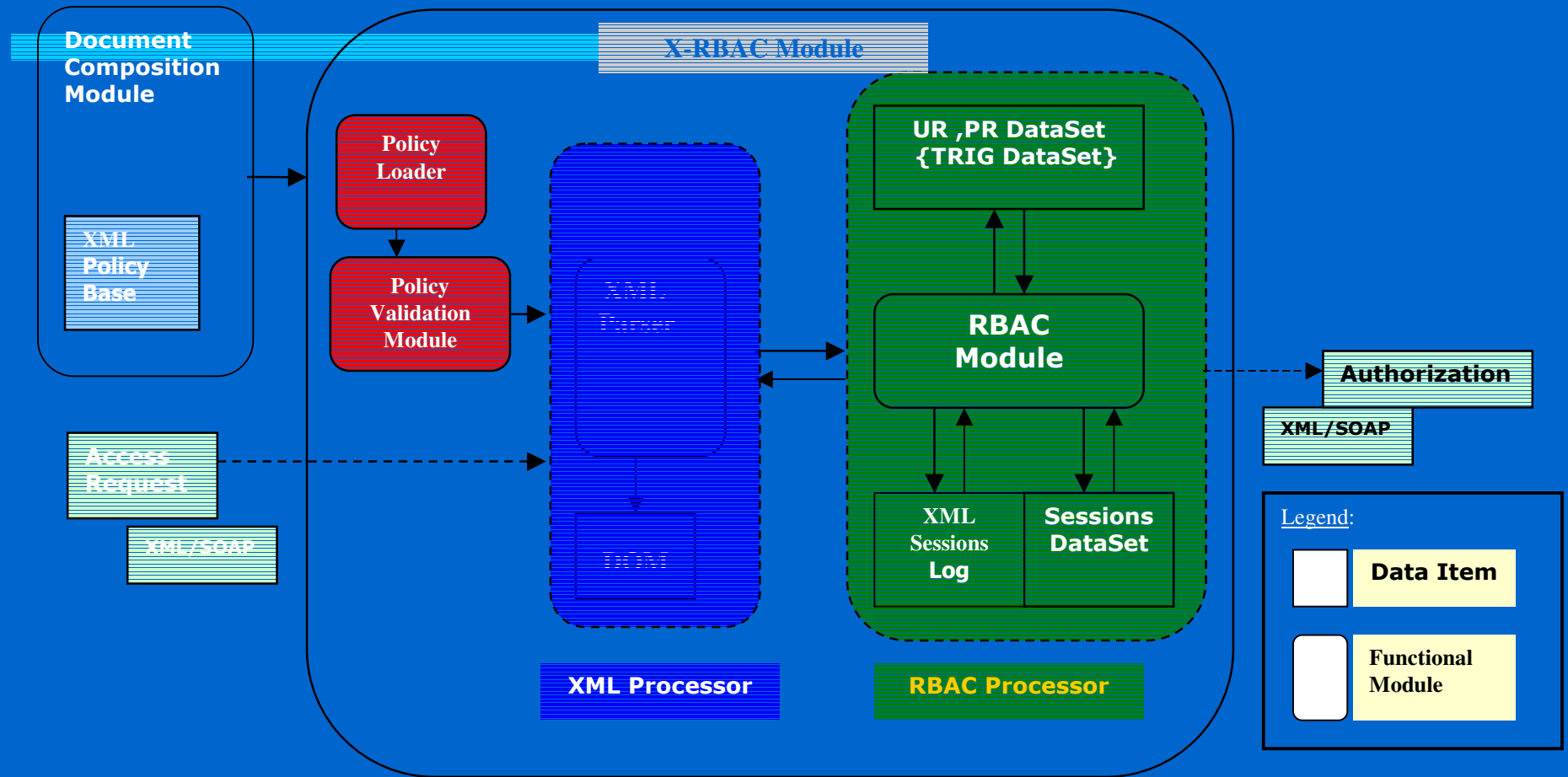
```
URA_id="URA1">
<RoleName> Eye_Doctor</ RoleName>
<Users>
  <User user_id="s1" />
  <User user_id="s2" />
</Users >
<CredConditions>
  <CredCondition>
    <CredType> Doctor </CredType>
    <LogicalExpr op="AND">
      <Predicate>
        <operator>eq</operator>
        <name_param>field</name_param>
        <value_param> Eye </value_param>
      </Predicate>
      <Predicate>
        <LogicalExpr op="OR">
```

```
<Predicate>
  <operator> lt </operator>
  <name_param> age </name_param>
  <value_param> 60 </value_param>
</Predicate>
<Predicate>
  <operator> gt </operator>
  <name_param> level </name_param>
  <value_param> 7 </value_param>
</Predicate>
</LogicalExpr>
</Predicate>
</LogicalExpr >
</CredCondition>
</CredConditions >
</URA>
</XURAS>
```

# Example of XPRAS

```
PRA pra_id="PRA1">
  <RoleName> Nurse </RoleName>
  <Permissions>
    <perm_id> P3 </perm_id>
  </Permissions>
</PRA>
<PRA pra_id="PRA2">
  <RoleName> Eye_Doctor
</RoleName>
  <Permissions>
    <perm_id> P1 </perm_id>
    <perm_id> P2 </perm_id>
  </Permissions>
</PRA>
</XPRAS>
```

# X-RBAC System Architecture



# On-going Work

---

- ◆ Extension of the constraint language
  - ⇒ Constraints on the set of roles a user can activate
- ◆ Obligations & Duties
- ◆ Development of graphical tools for TRBAC administration
- ◆ Testing on an Healthcare information system