ApacheDS Access Control Administration; The X.500 Way

•Originally presented at *ApacheCon US 2006* in Austin, Texas

•Latest presentation materials are at *http://people.apache.org/~ersiner*

•Presented by *Ersin Er*, *ersiner@apache.org*

Agenda

- Access Control
- X.500 Access Control Model
- ACI Items
- Directory Access Control Domains
- Some Principals
- Access Control Administrative Areas
- Delegation of Authority
- Demos

Access Control

- Access: Performing an operation on a resource and getting back some result
- Control: Defining
  - **Who** can
  - access **How**
  - to **What**

X.500 Access Control

- Defined in terms of Access Control Information (ACI) Items
  - Who can: *User Classes*
  - access How: *Grants and Denials*
  - to What: *Protected Items*

ACI items are powerful

- Fine grained access control
  - Various options with different semantics and capabilities to specify User Classes and Protected Items
- Matter of style - different ways to define access control
  - Who can – (access How, to What)*
  - to What – (Who can, access How)*

Exploring ACIItem components: User Classes

- A specific user
- A user group
- All users
- The user who accesses his/her own (user) entry (himself/herself)
- Users (entries) falling under a *subtree (selection)*

Exploring ACIItem components: Protected Items

- An entry itself (but not any attribute types or values)
- All user attribute types and values
- All user attribute types (but not any values)
- All attribute values (of a type)
- An attribute type (but not any values)
- Attribute value (of a type)
- User’s DN as an attribute value (of a type)

Exploring ACIItem components: Grants and Denials

- Some *Grants and Denials* that can be associated with an *Entry*:
  - Read
  - Browse
  - ReturnDN
  - Add
  - Modify
  - Remove
  - Rename
  - Export
  - Import

Exploring ACIItem components: Grants and Denials

- Some *Grants and Denials* that can be associated with both an *Attribute Type* and an *Attribute Value*:
  - Read
  - Compare
  - FilterMatch
  - Add
  - Remove

DEMO 1

- Grant read access to an attribute for a user
  - Allow John to read his name!

DEMO 1 – entryACI

```
• { identificationTag  "allowJohnToReadHisName_ACI",
•   precedence 10, authenticationLevel simple,
•   itemOrUserFirst userFirst: {
•     userClasses {
•       name { "cn=John,ou=users,ou=system" }
•     },
•     userPermissions {
•       { protectedItems { entry },
•         grantsAndDenials { grantBrowse }
•       },
•       { protectedItems {
•           attributeType { commonName },
•           allAttributeValues { commonName }
•         },
•         grantsAndDenials { grantRead }
•       }
•     }
•   }
• }
```

Put this in John's user entry

DEMO 2

- Grant modify access to an attribute for a user
  - Allow Jane to change her password!

DEMO 2 – entryACI

```
{ identificationTag  "allowJaneToChangeHerPassword_ACI",
    precedence 10, authenticationLevel simple,
    itemOrUserFirst userFirst: {
      userClasses {
        name { "cn=Jane,ou=users,ou=system" }
      },
      userPermissions {
        { protectedItems { entry },
          grantsAndDenials { grantModify }
        },
        { protectedItems {
            allAttributeValues { userPassword }
          },
          grantsAndDenials { grantRemove, grantAdd }
        }
      }
    }
}
```

Put this in Jane's user entry

DEMO 3

- Grant read and modify access of a specific value to an attribute for a user
  - Allow Jim to subscribe to/unsubscribe from a mail list!

DEMO3 – entryACI

```
{ identificationTag "allowJimToSubsToUnsubsFromAMailList_ACI",
  precedence 10, authenticationLevel simple,
  itemOrUserFirst userFirst: {
    userClasses { name { "cn=Jim,ou=users,ou=system" } },
    userPermissions {
      { protectedItems { entry },
        grantsAndDenials { grantModify }
      },
      { protectedItems {
          attributeValue {
            uniqueMember="cn=Jim,ou=users,ou=system"
          }
        },
        grantsAndDenials { grantRemove, grantAdd }
      }
    }
  }
}
```

Put this in the mailing list entry

Questions

- Are we going to create
  - each of these entryACI values
  - for each user entry
  - if we want them all have same permissions?
- Changes from user to user seem to be few, don't they?

(Unsatisfactory) Answer

- We can replace user specific parts in each entryACI with generic components:
  - For 1st and 2nd cases:
    - UserClass *thisEntry*
      - Not for the 3rd case, because the accessed entry is not the user's entry
  - For 3rd case:
    - UserClass *allUsers*
    - ProtectedItem *selfValue*
  - can be used.

DEMO 4

- Grant read and modify access to an attribute for a user
  - Allow user to read/change his or her password!

DEMO 4 – entryACI

```
{ identificationTag  "allowUserToChangeHisOrHerPassword_ACI",
  precedence 10, authenticationLevel simple,
  itemOrUserFirst userFirst: {
    userClasses {
      thisEntry
    },
    userPermissions {
      { protectedItems {
          entry
        },
        grantsAndDenials {
          grantModify
        }
      },
      { protectedItems {
          allAttributeValues { userPassword }
        },
        grantsAndDenials {
          grantRemove, grantAdd
        }
      }
    }
  }
}
```

Put this in a user's entry

DEMO 5

- Grant read and modify access of a specific value to an attribute for a user
  – Allow users to subscribe to/unsubscribe from a mailing list!

DEMO 5 – entryACI

```
•   { identificationTag  "allowAllUsersToSubscribeToUnsubscribeFromAMailingList_ACI",
•     precedence 10, authenticationLevel simple,
•     itemOrUserFirst userFirst: {
•       userClasses {
•         allUsers
•       },
•       userPermissions {
•         { protectedItems { entry },
•           grantsAndDenials { grantModify }
•         },
•         { protectedItems {
•             selfValue
•           },
•           grantsAndDenials {
•             grantRemove, grantAdd
•           }
•         }
•       }
•     }
•   }
```

Put this in a mailing list entry

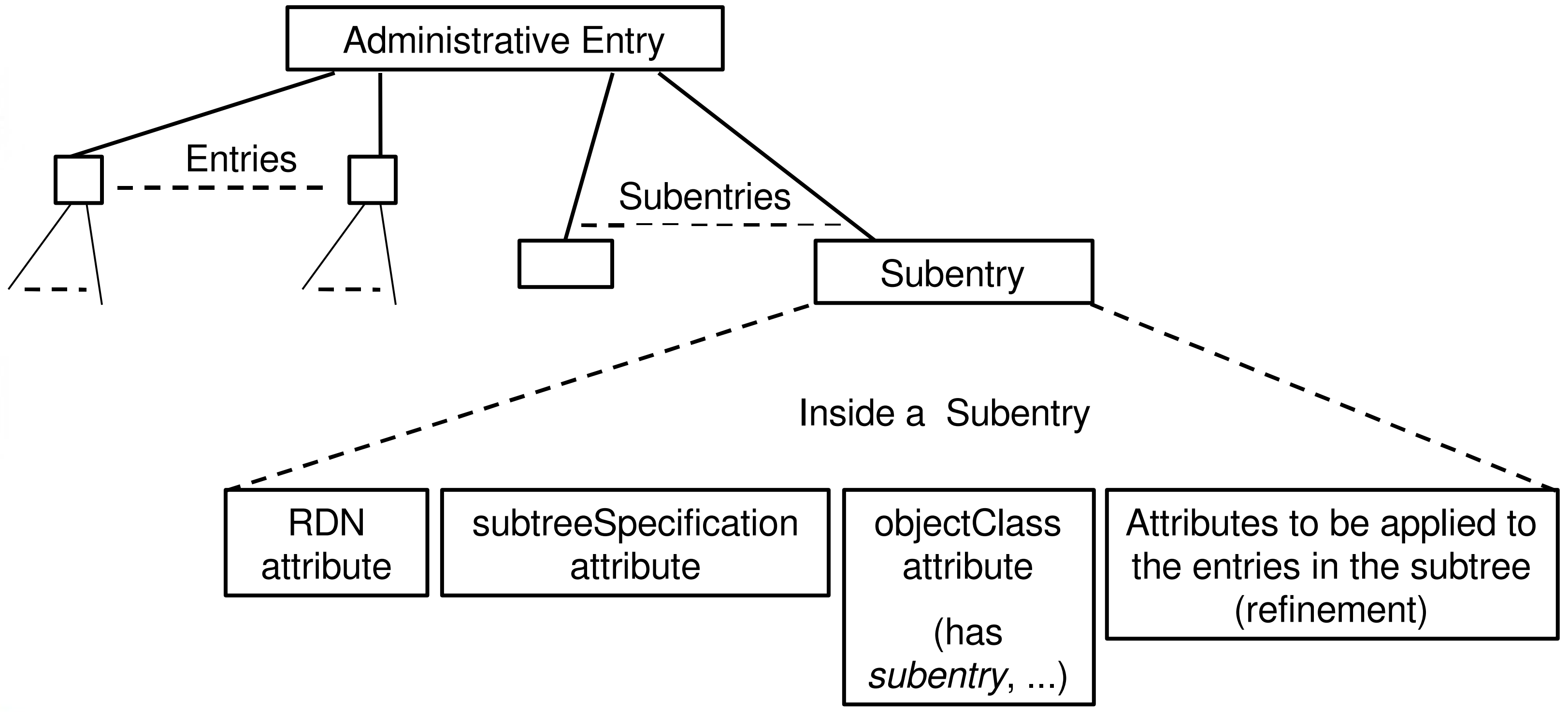
Why was the answer unsatisfactory? New Questions

- Solutions for the 3rd case is OK but,
- For 1st and 2nd cases,
  - What about hundreds of thousands of users that need the same permissions?
  - What about much more access control for each user?
  - Will we have to add each generic entryACI value to all users' entries?
- *So, do we always have to define access control information on a single entry only? What about a set of entries?*

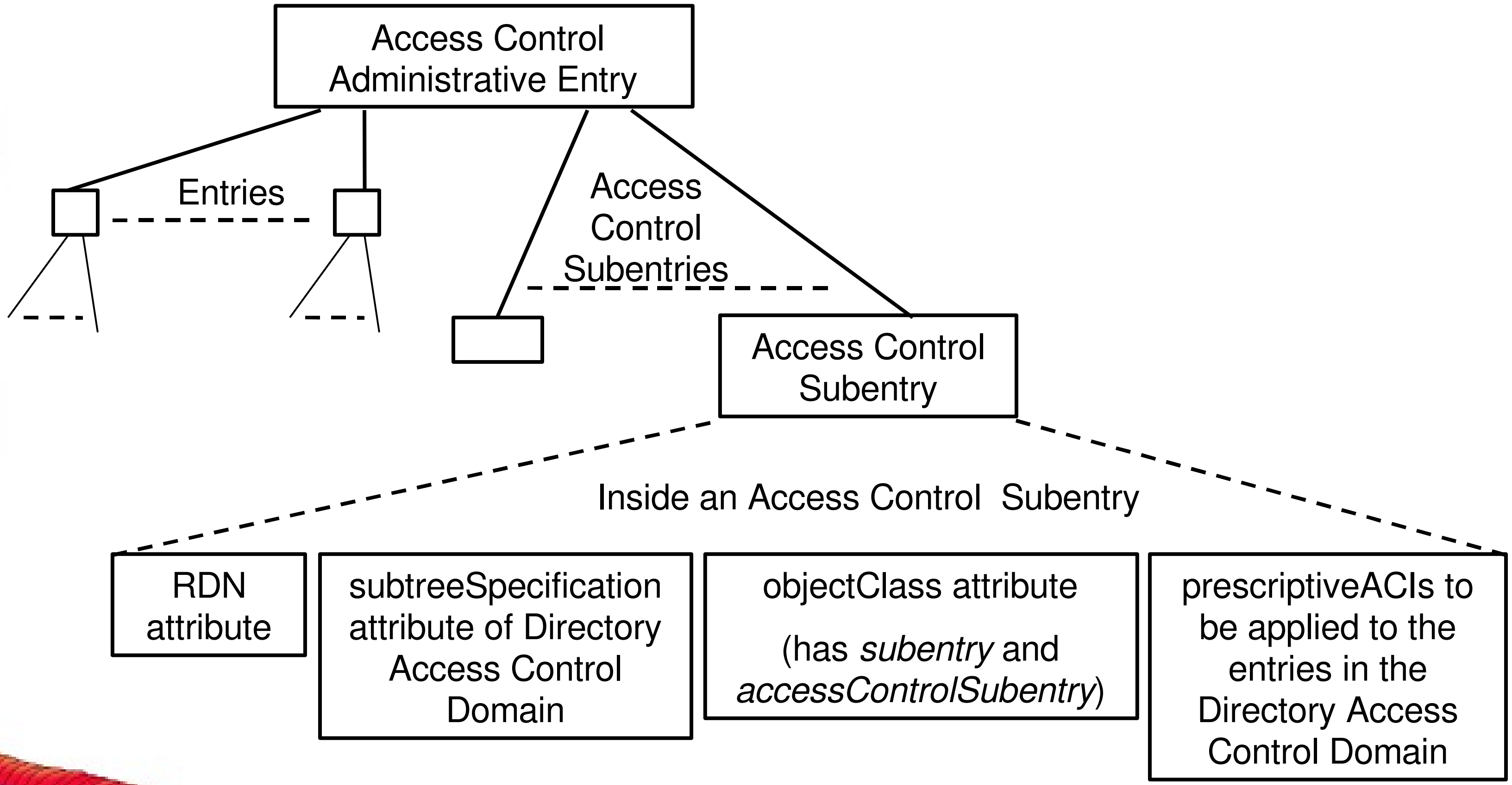
Satisfactory Answer: Directory Access Control Domains (DACD)

- X.500 Subentries and subtreeSpecification
  - A Subentry holds a subtreeSpecification attribute
  - subtreeSpecification allows specifying a *subtree of entries with chop specifications and refinements*
  - Other attributes in the Subentry are *applied* to the selection of entries
  - A building block of X.500 Administrative Model
  - RFC 3672 - Subentries in the Lightweight Directory Access Protocol
- Directory Access Control Domains
  - Instead of entryACI,
  - use *prescriptiveACI* in *accessControlSubentry*
  - to define access control rules on a set of entries

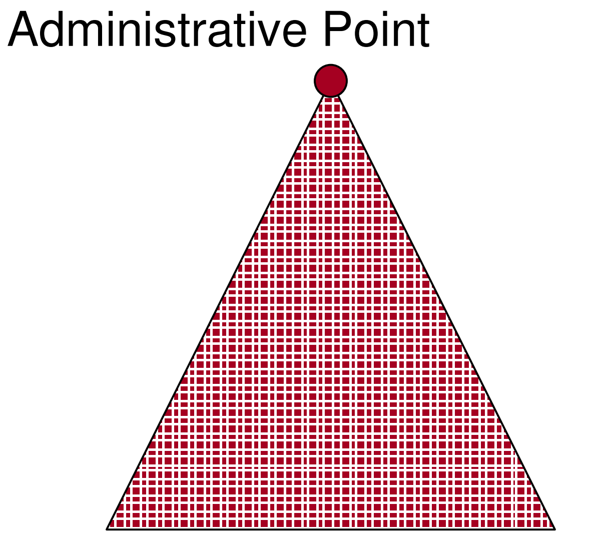
X.500 Administrative Model


Administrative Entry
Entries
Subentries
Subentry
Inside a Subentry
RDN attribute
subtreeSpecification attribute
objectClass attribute (has *subentry*, ...)
Attributes to be applied to the entries in the subtree (refinement)

X.500 Administrative Model – Access Control Aspect

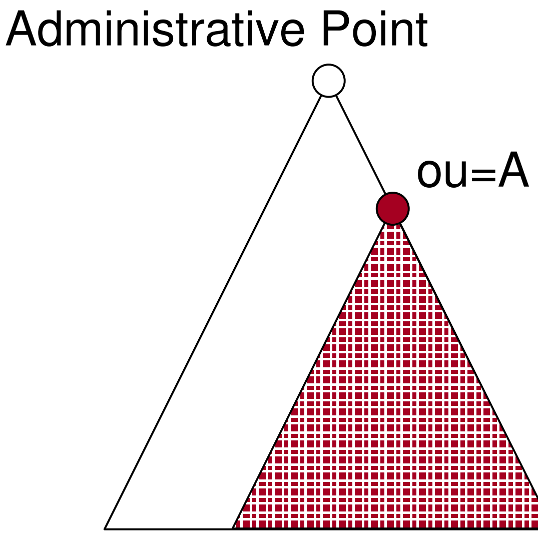

Access Control Administrative Entry
Entries
Access Control Subentries
Access Control Subentry
Inside an Access Control Subentry
RDN attribute
subtreeSpecification attribute of Directory Access Control Domain
objectClass attribute
(has *subentry* and *accessControlSubentry*)
prescriptiveACIs to be applied to the entries in the Directory Access Control Domain

What can be specified
(How a DACD can be specified)
with a subtreeSpecification ? (1)



subtreeSpecification=
{ }

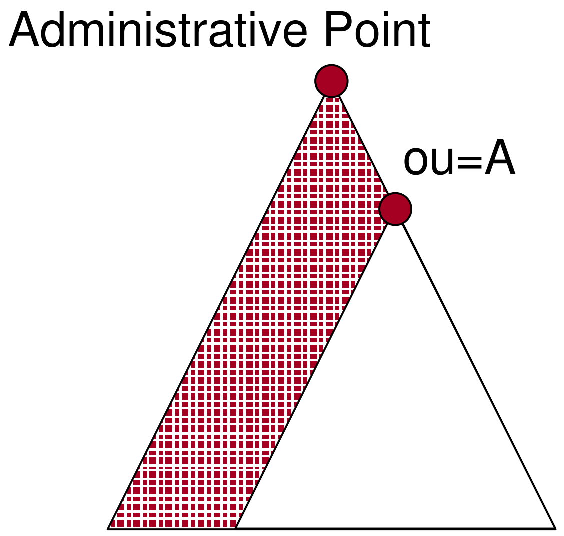
What can be specified *(How a DACD can be specified)* with a subtreeSpecification ? (2)



subtreeSpecification=

{ base "ou=A" }

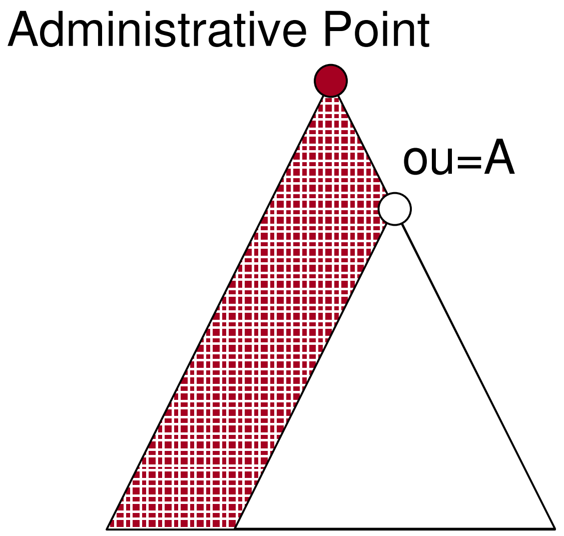
What can be specified *(How a DACD can be specified)* with a subtreeSpecification ? (3)



subtreeSpecification=

{ specificExclusions { chopAfter: “ou=A” } }

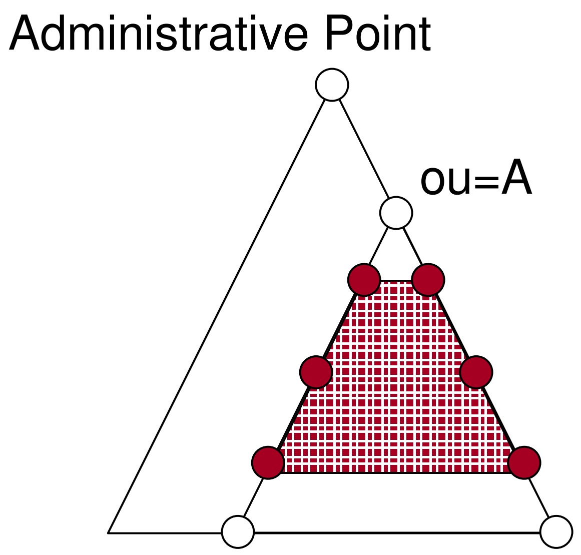
What can be specified *(How a DACD can be specified)* with a subtreeSpecification ? (4)



subtreeSpecification=

{ specificExclusions { chopBefore: “ou=A” } }

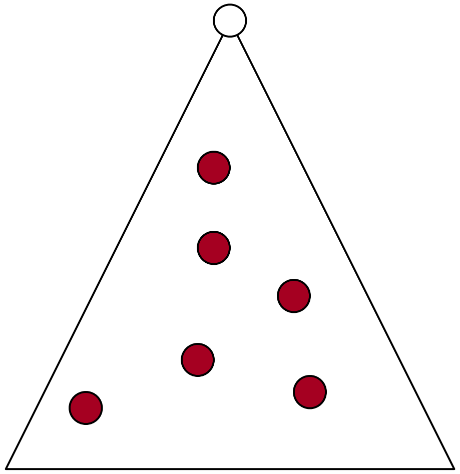
What can be specified *(How a DACD can be specified)* with a subtreeSpecification ? (5)



subtreeSpecification=

{ base “ou=A”, minimum 1, maximum 3 }

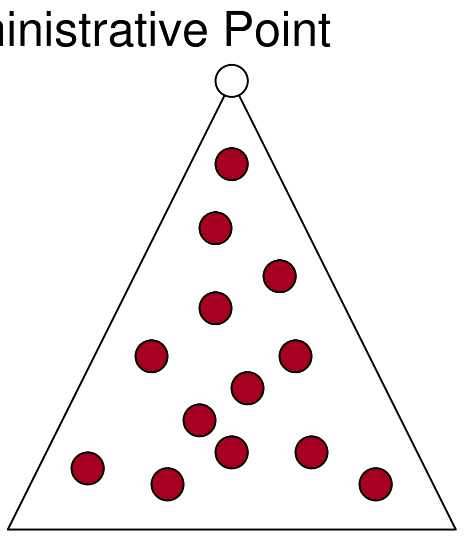
What can be specified *(How a DACD can be specified)* with a subtreeSpecification ? (6)



subtreeSpecification=

{ specificationFilter item:student }

What can be specified *(How a DACD can be specified)* with a subtreeSpecification ? (7)



subtreeSpecification=

{ specificationFilter or: { item:student,

item:faculty } }

DEMO 6

- Allow all users on the whole domain
  - to do search operations by commonName attribute
  - read (as search result) and compare commonName and telephoneNumber attributes

DEMO 6 – subtreeSpecification

- subtreeSpecification: { }
- All entries in the domain will be subject to access control by prescriptiveACIs defined in the accessControlSubentry

DEMO 6 – prescriptiveACI

```
{ identificationTag  "demo6_ACI",
  precedence 10, authenticationLevel simple,
  itemOrUserFirst userFirst: {
    userClasses { allUsers },
    userPermissions {
      { protectedItems { entry },
        grantsAndDenials { grantBrowse, grantRead, grantReturnDN }
      },
      { protectedItems {
          attributeType {
            commonName, telephoneNumber }
          allAttributeValues {
            commonName, telephoneNumber }
        },
        grantsAndDenials {
          grantRead, grantCompare, grantFilterMatch }
      }
    }
  }
}
```

DEMO 7

- Allow a group of user administrators on the whole domain
  - to do all operations on only “person” entries
  - to do all operations related to entryACIs in users’ entries

DEMO 7 – subtreeSpecification

- subtreeSpecification:

  { specificationFilter item:person }

- All “person” entries in the domain will be subject to access control by prescriptiveACIs defined in the accessControlSubentry

DEMO 7 – prescriptiveACI

```
•  { identificationTag  "demo7_ACI",
•    precedence 10, authenticationLevel simple,
•    itemOrUserFirst userFirst: {
•      userClasses { userGroup { "ou=User Administrators,..." } },
•      userPermissions {
•        { protectedItems { entry },
•          grantsAndDenials { grantBrowse, grantRead, grantModify }
•        },
•        { protectedItems {
•            allUserAttributeTypesAndValues,
•            attributeType { entryACI },
•            allAttributeValues { entryACI }
•          },
•          grantsAndDenials {
•            grantRead, grantCompare, grantFilterMatch, grantAdd, grantRemove
•          }
•        }
•      }
•    }
•  }
```

Some Principals

- Least privilige by default
  - Information is not present unless it's allowed to access it
- Precedence
  - Higher override Lower
- Specificity
  - If precedences are the same
    - For User Classes and Protected Items
      - Higher overrides Lower
- Denials vs. Grants
  - If precedences and specificities are same
    - Denials override Grants

Applications of Principals – Precedence

- Two ACIItems
  - precedence 10, deny search for all users
  - precedence 20, grant search for all users
- Result: Search is allowed for all users

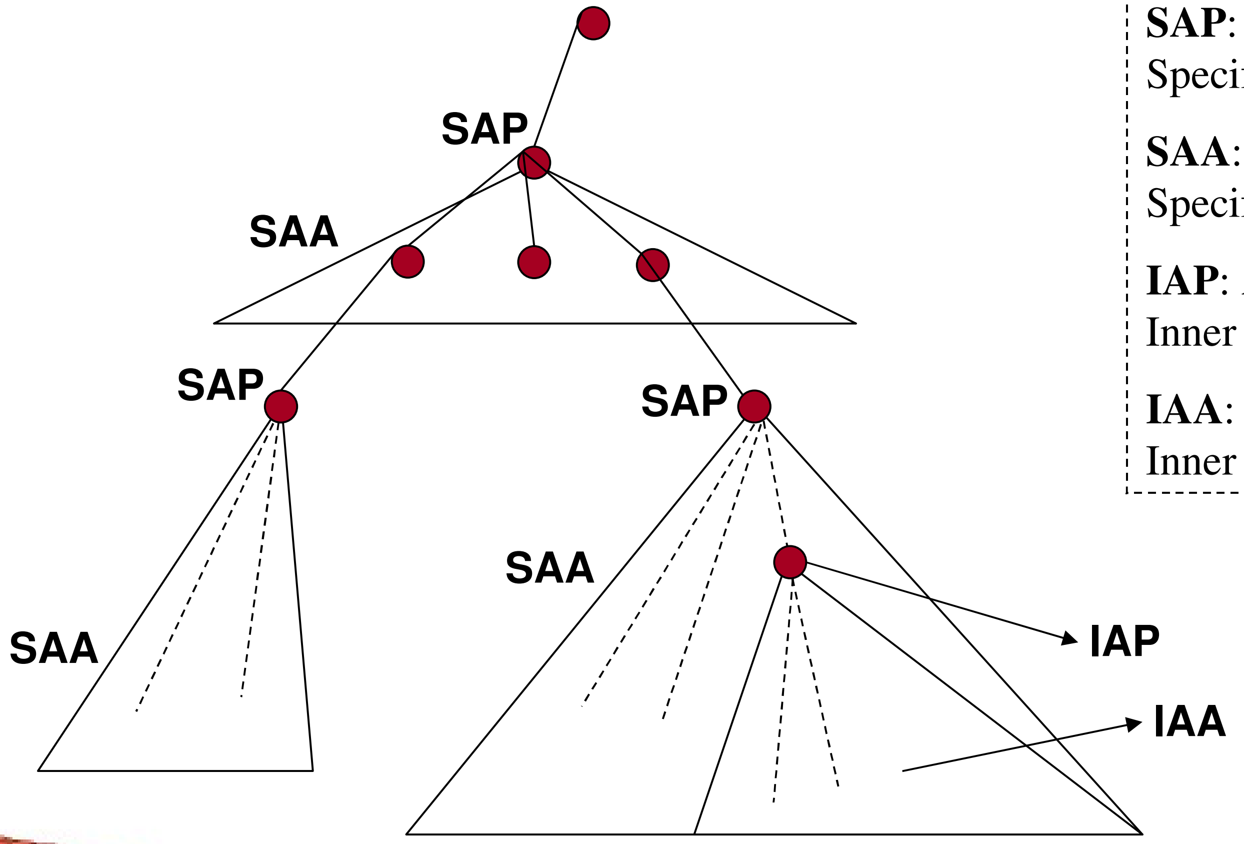
Applications of Principals – Specificity

- Two ACIItems with the same precedence
  - deny search for all users
  - grant search for user Jimmy
- Result: Search is allowed for Jimmy

Applications of Principals – Grants vs. Denials

- Two ACIItems with the same precedence
  - deny search for all users
  - grant search for all users
- Result: Search is not allowed for users

Access Control Administrative Areas



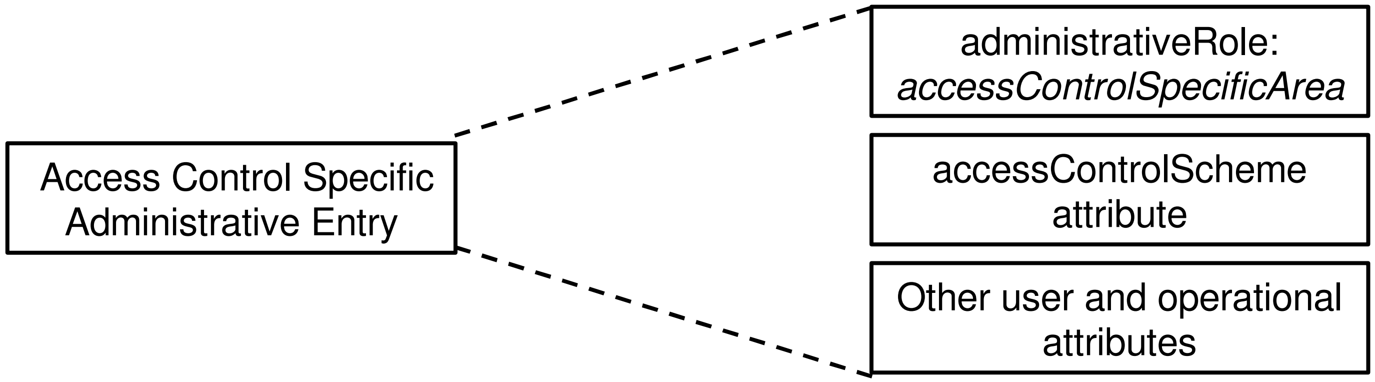
SAP: Access Control Specific Admin. Point

SAA: Access Control Specific Admin. Area

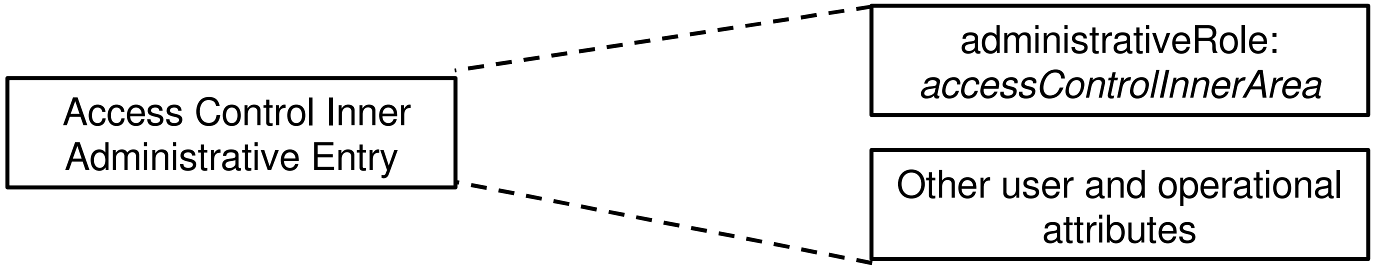
IAP: Access Control Inner Admin. Point

IAA: Access Control Inner Admin. Area

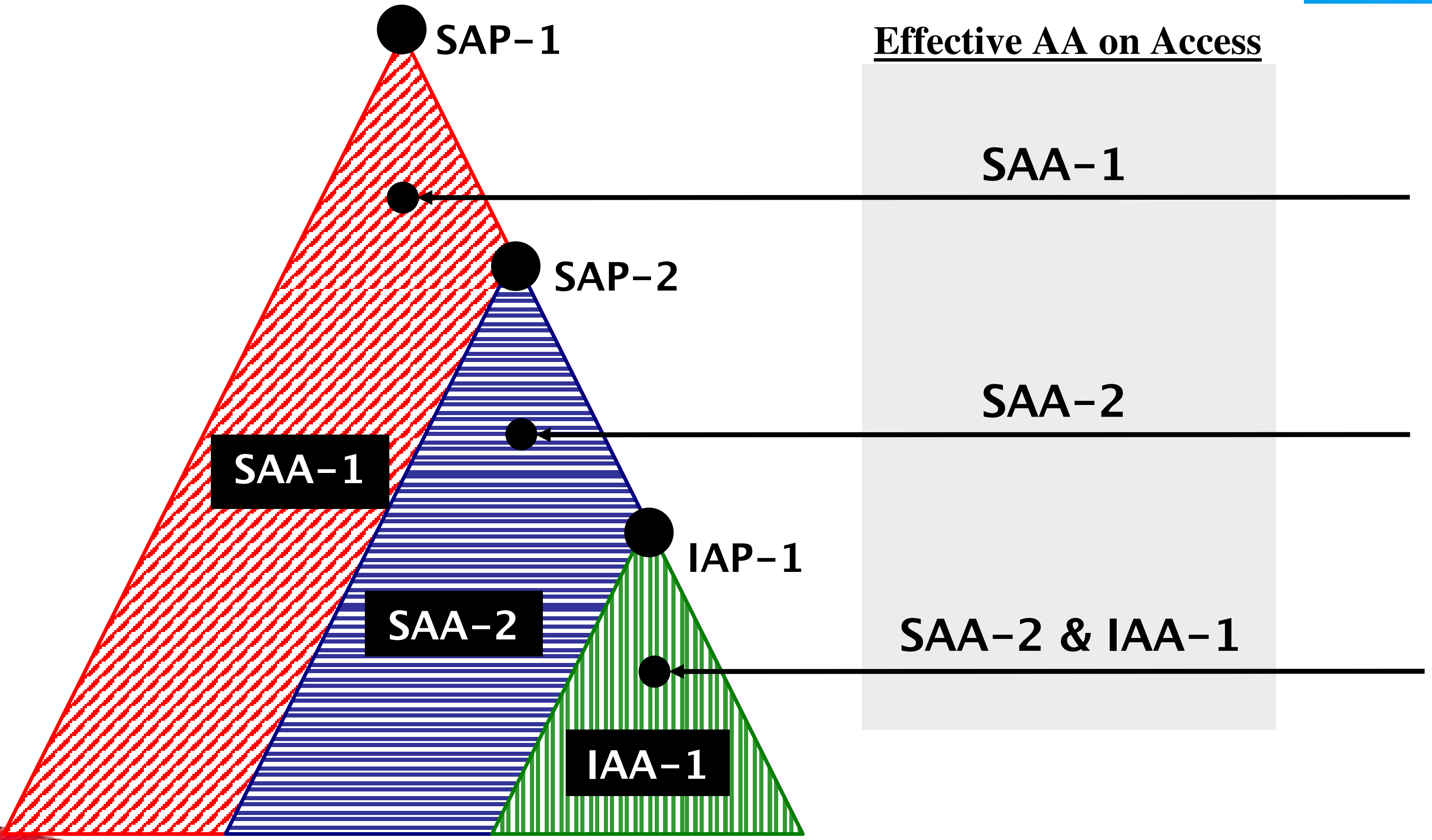
Inside an Access Control *Specific* Administrative Point (Entry)


Access Control Specific Administrative Entry
administrativeRole: *accessControlSpecificArea*
accessControlScheme attribute
Other user and operational attributes

Inside an Access Control *Inner* Administrative Point (Entry)


Access Control Inner Administrative Entry
administrativeRole: *accessControlInnerArea*
Other user and operational attributes

Effects of Administrative Areas on Access Control



Delegation of Authority

- Complete Delegation of Authority
  - Access Control Specific Areas
- Partial Delegation of Authority
  - Access Control Inner Areas
  - ACI precedence

ApacheDS Access Control Administration; The X.500 Way

•Originally presented at ***ApacheCon US 2006*** in Austin, Texas

•Latest presentation materials are at ***http://people.apache.org/~ersiner***

•Presented by ***Ersin Er***, ***ersiner@apache.org***