

ONLY  
THE BEST  
GET IN

# JavaPolis 2004

## Access Control Architectures: COM+ vs. EJB



Dr. Konstantin Beznosov  
Assistant Professor  
University of British Columbia



ONLY  
THE BEST  
GET IN

## Overall Presentation Goal

**Learn about the capabilities of  
COM+ and EJB  
access control mechanisms**

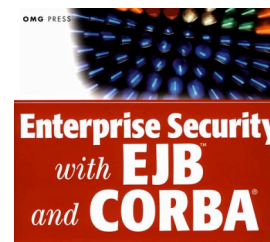


ONLY  
THE BEST  
GET IN

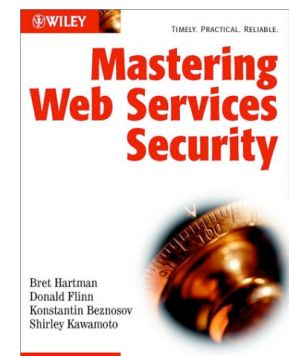
# Speaker's Qualifications

## Konstantin

- Worked for end-user, consulting, and developer organizations
- Co-authored CORBA Security specification proposals
  - Resource Access Decision
  - Security Domain Membership Management (SDMM)
  - CORBA Security
- Co-authored



Bret Hartman, Donald J. Flinn,  
and Konstantin Beznosov  
Foreword by Steve Vinoski, JONA Technologies



Bret Hartman  
Donald Flinn  
Konstantin Beznosov  
Shirley Kawamoto



ONLY  
THE BEST  
GET IN

## Conclusions

- Access control capabilities in both COM+ and EJB suck!
- One of them, however, is extensible



ONLY  
THE BEST  
GET IN

# outline

- Introduction
  - distributed security basics
  - distributed access control
  - evaluation criteria
- COM+
- EJB
- Conclusions



ONLY  
THE BEST  
GET IN

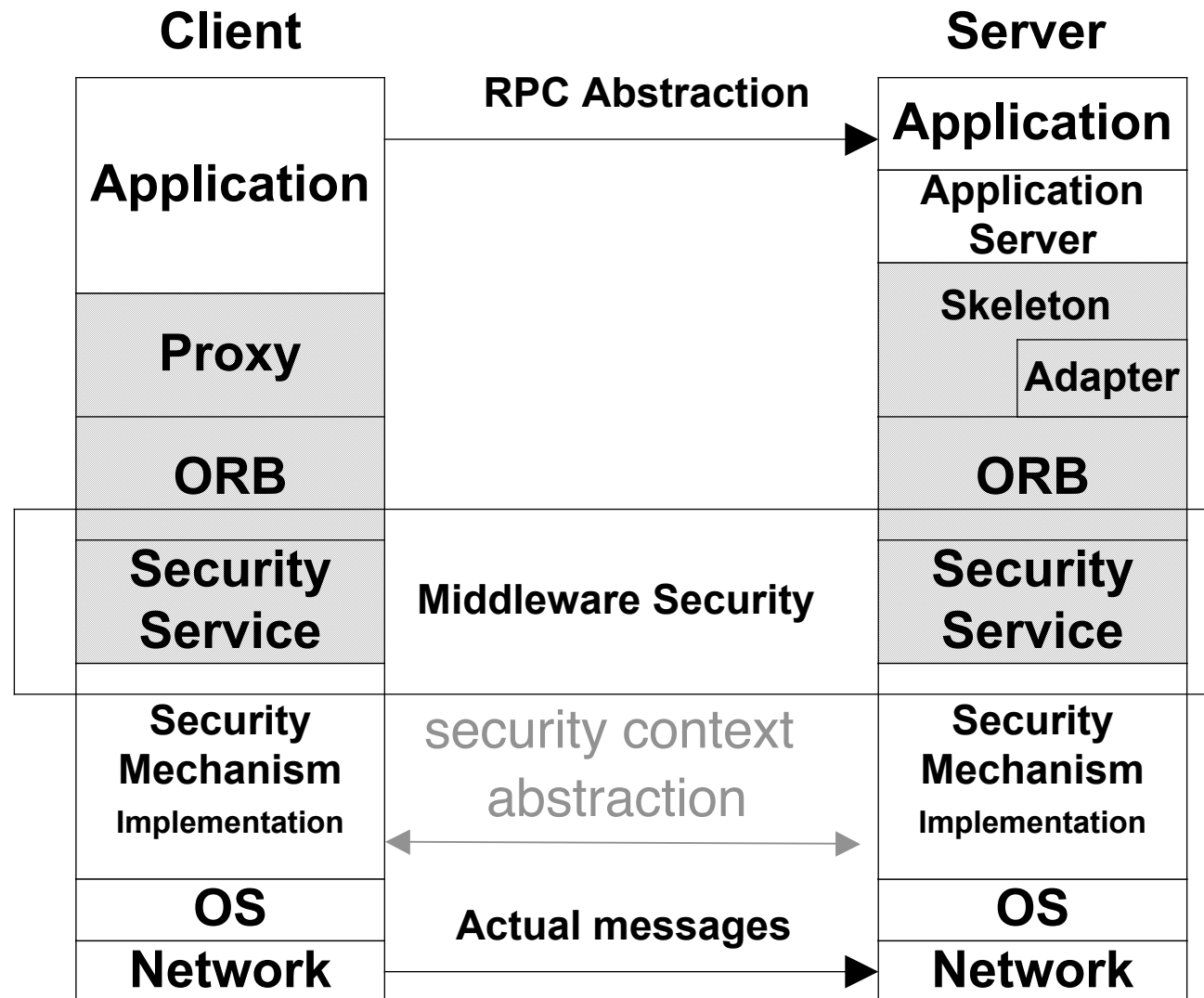


# distributed security basics



ONLY  
THE BEST  
GET IN

# Middleware Security Stack





ONLY  
THE BEST  
GET IN

# requirements due to distribution

- centralized administration
- localized run-time decisions





ONLY  
THE BEST  
GET IN

# object paradigm & security (1/2)

- objects
  - small amounts of data ==> large numbers
    - o R: **Scale on large numbers of objects and methods**
  - diverse methods ==> complex semantics
    - o R: **Security administrators should not have to understand semantics of methods**
- collections
  - R: **Similar names or locations should NOT impose membership in same collection(s).**
  - R: **For an object to be assigned to the same collection, name similarity and/or co-location should not be required.**



ONLY  
THE BEST  
GET IN

## object paradigm & security (2/2)

- many layers of indirection and late binding
- names
  - multi-name, nameless and transient objects
  - R: **Transient objects should be assigned to security policies without human intervention.**
  - less rigid naming hierarchies
  - R: **No assumptions that administrators know a name of each object in the system.**



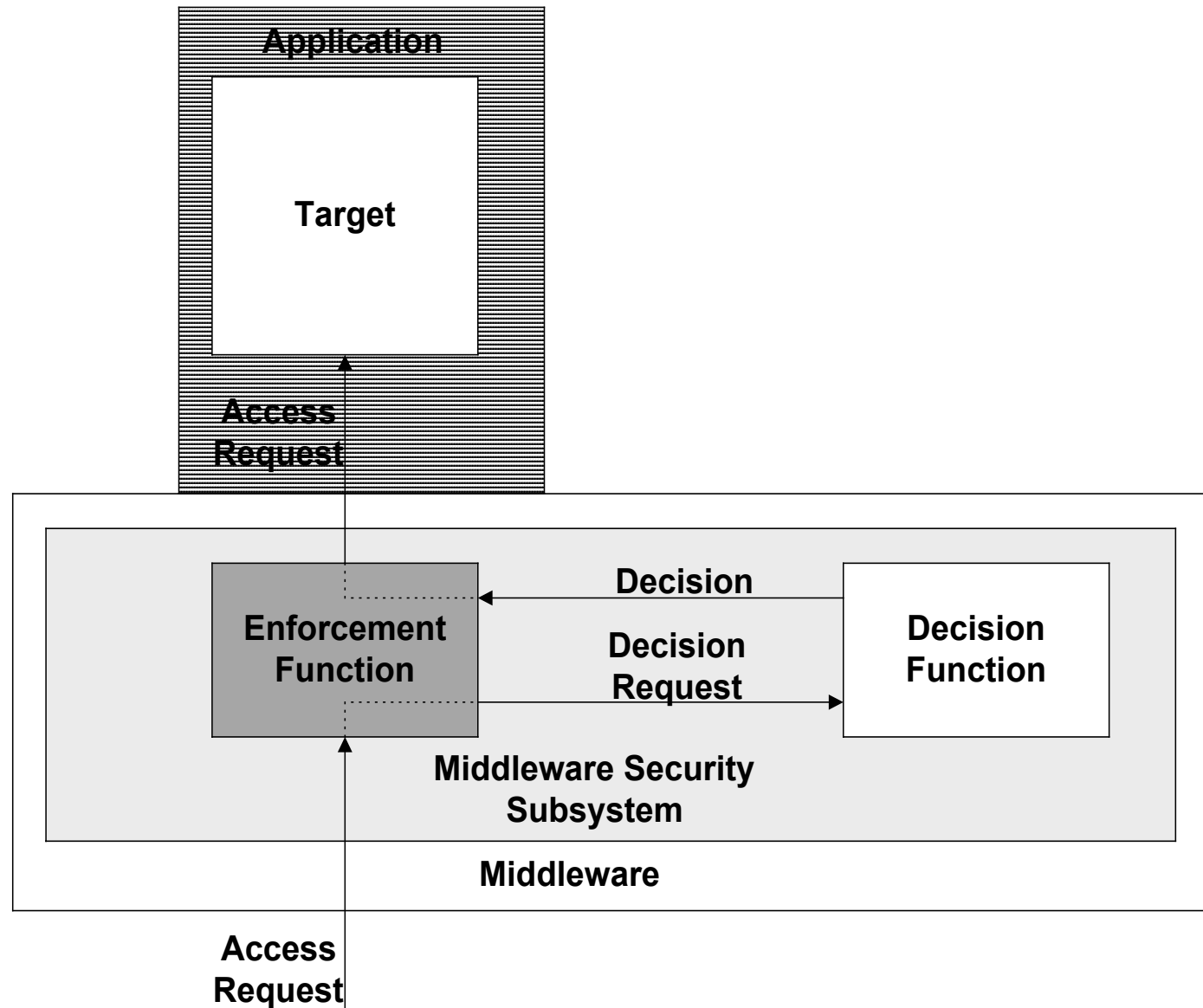
ONLY  
THE BEST  
GET IN

distributed access control



ONLY  
THE BEST  
GET IN

# Access Control at Run-time



ONLY  
THE BEST  
GET IN

# authorization decisions

- which policies?
  - which collections
- policy composition
- policy evaluation
  - information push vs. pull



ONLY  
THE BEST  
GET IN

## evaluation criteria 1/2

- **GRANULARITY** -- granularity of protected resources
  - application, interface, method, arbitrary resource.
- **EXPRESSIVENESS** -- support for different access control models
- **RICHNESS** -- the variety of information available for making authorization decisions, including application-specific information
- **CONSISTENCY** -- support for consistency of policies across multiple applications



ONLY  
THE BEST  
GET IN

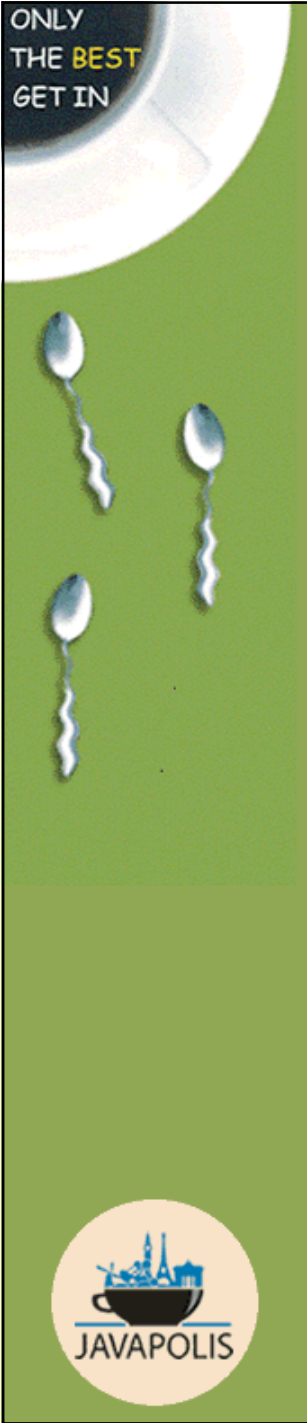
## evaluation criteria 2/2

- **MANAGEABILITY** -- support for insertion and deletion of applications, changes in policies, user population and the computing environment
- **SCALABILITY** -- performance and administration scalability
- **OBJECT PARADIGM REQUIREMENTS** -- satisfying the requirements due to the object paradigm
- **EXTENSIBILITY** -- support for unforeseen policies





ONLY  
THE BEST  
GET IN



COM+



ONLY  
THE BEST  
GET IN

# Administering Access Control

The screenshot displays the Visual Studio IDE with a tree view on the left and a 'GetPrice Properties' dialog box on the right. The tree view shows the project structure for 'Storefront Server', including components, interfaces, methods, and roles. The 'GetPrice' method is selected in the tree, and the 'GetPrice Properties' dialog is open to the 'Security' tab. The dialog shows the roles inherited by the selected item (Staff) and the roles explicitly set for the selected item (Staff, Member, Customer, Visitor, Marshaler). The 'Member' role is checked in the 'Roles explicitly set' list.

**Storefront Server**

- Components
  - Ebusiness.StoreFrontMiddleTier.Helper
  - Ebusiness.StoreFrontMiddleTier.Product
  - Ebusiness.StoreFrontMiddleTier.ProductManager
  - Ebusiness.StoreFrontMiddleTier.ShoppingCart
  - Ebusiness.StoreFrontMiddleTier.SpecialProduct
    - Interfaces
      - \_Object
      - \_SpecialProduct
      - IDisposable
      - IManagedObject
      - IProduct
        - Methods
          - GetPrice
          - Initialize
          - SetPrice
        - IRemoteDispatch
        - System\_EnterpriseServices\_IServiceContract

- Roles
- Customer
- Marshaler
- Member
- Staff
- Visitor
- Users
- KONSTANTIN-1\bcustomers
- KONSTANTIN-1\bmembers
- KONSTANTIN-1\bstaff
- KONSTANTIN-1\bvisitors

**GetPrice Properties**

General Security

Roles inherited by selected item(s):

Name
Staff

Roles explicitly set for selected item(s):

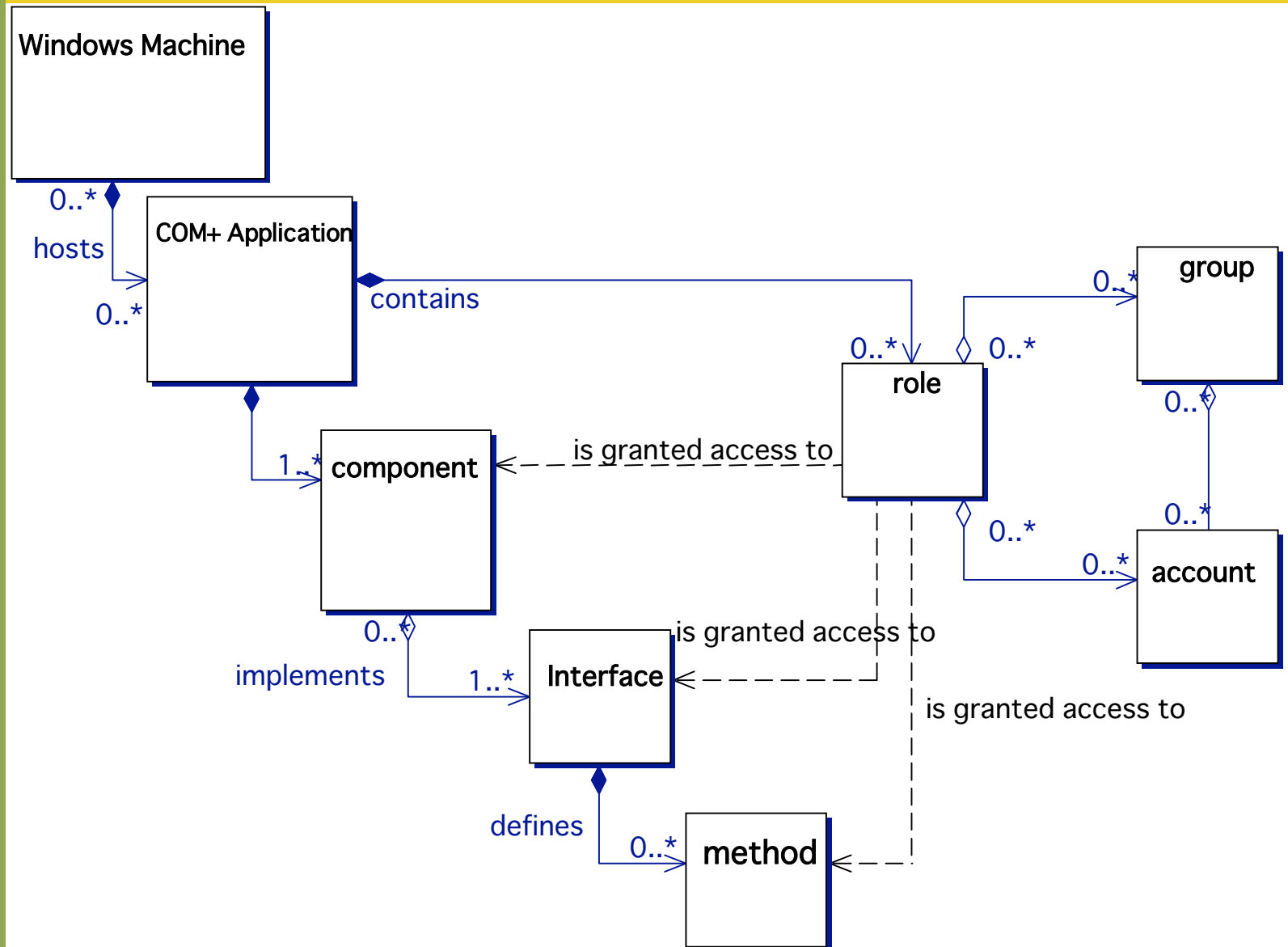
Name
<input type="checkbox"/> Staff
<input checked="" type="checkbox"/> Member
<input type="checkbox"/> Customer
<input type="checkbox"/> Visitor
<input type="checkbox"/> Marshaler

OK Cancel Apply



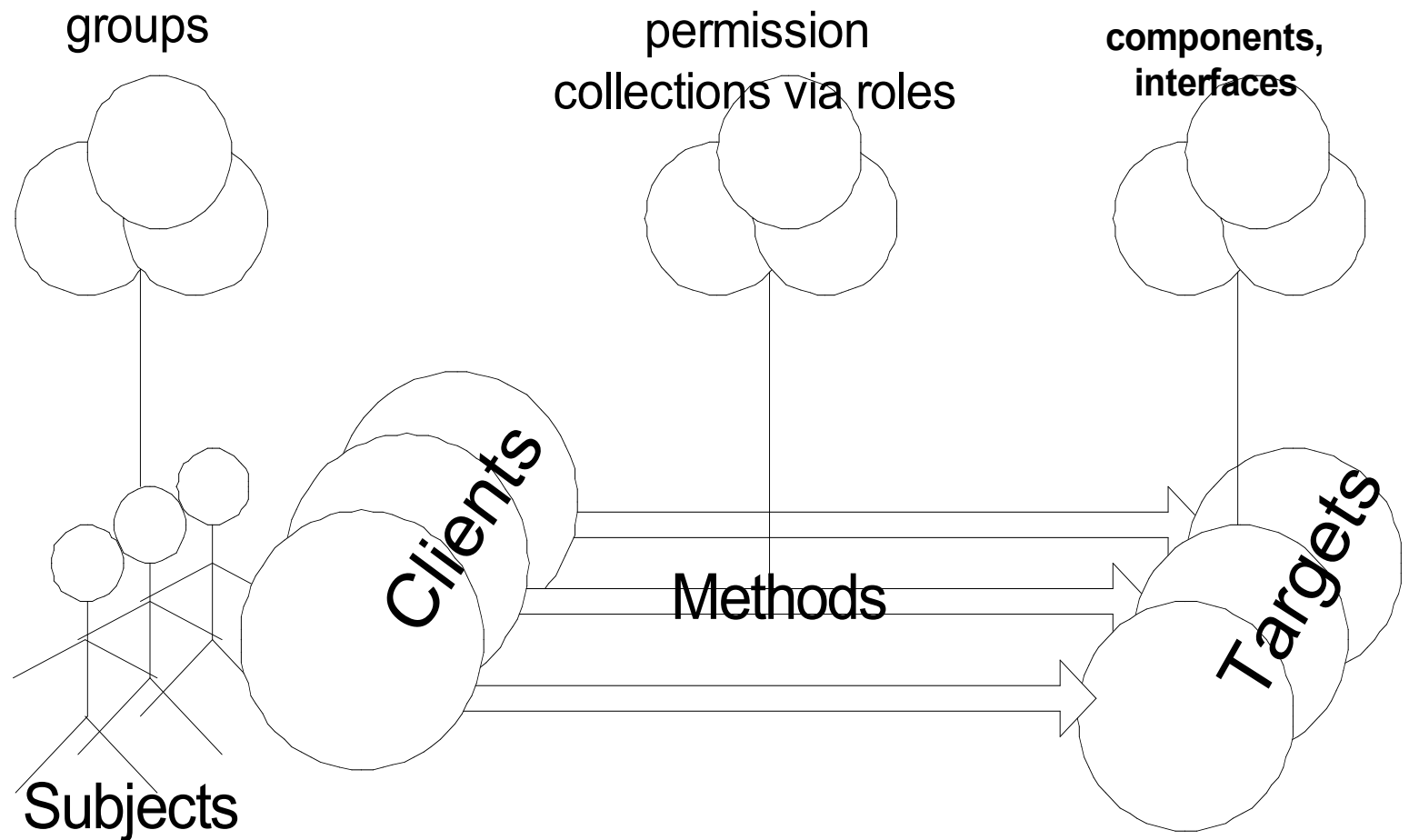
ONLY  
THE BEST  
GET IN

# COM+ Access Control Architecture



ONLY  
THE BEST  
GET IN

# scaling with collections



ONLY  
THE BEST  
GET IN

# evaluating COM+ 1/2

- **Granularity**
  - + component method
  - but not component instance method
- **Expressiveness** -- supporting different policies
  - + RBAC<sub>0</sub>
  - + RBAC<sub>1</sub> through W2K domain nested groups
- **Richness** -- information for making decisions
  - subject group attributes, component type and method
- **Consistency** -- across multiple applications
  - requires application redeployment, or manual changes in each application instance



ONLY  
THE BEST  
GET IN

## evaluating COM+ 2/2

- **Manageability** -- changes to policies, users, appl-s
  - + user population -- Windows domain groups could help
  - application population
    - + replication -- easy to use packaging
    - o new -- labor intensive and error prone
  - changes in policies -- labor intensive and error prone
  - computing environment -- easy to use packaging
- **Scalability** -- performance and admin. scalability
  - + subject groups, several levels of permission granularity, permission collections
  - permissions (collections) local to the application
  - bound by the underlying OS scalability
- **Object paradigm requirements**
  - + roles isolate administrators from method semantics
  - machine co-located instances of the same component are governed by one policy
- **Extensibility** -- support for unforeseen policies
  - only through “programmatically security” inside of application





ONLY  
THE BEST  
GET IN

## further reading on COM+

- B. Hartman, D. J. Flinn, K. Beznosov, and S. Kawamoto, chapter 7, ***Mastering Web Services Security***, 1st ed. New York: John Wiley & Sons, Inc., 2003.
- K. Brown, ***Programming Windows Security***, First ed. Upper Saddle River, NJ: Addison-Wesley, 2000.
- M. Howard, M. Levy, and R. Waymire, ***Designing Secure Web-based Applications for Microsoft Windows 2000***. Redmond, Washington USA: Microsoft Press, 2000.
- MSDN Knowledge Base. <http://msdn.microsoft.com>





ONLY  
THE BEST  
GET IN



EJB

ONLY  
THE BEST  
GET IN

# Defining Roles in EJB

```
<assembly-descriptor>
  <security-role>
    <description>
      blah-blah-blah ...
    </description>
    <role-name>member</role-name>
  </security-role>

  <security-role>
    <description>
      blah-blah-blah ...
    </description>
    <role-name>customer</role-name>
  </security-role>
  <security-role>
    <description>
      blah-blah-blah ...    </description>
    <role-name>staff</role-name>
  </security-role>
  ...
</assembly-descriptor>
```



ONLY  
THE BEST  
GET IN

## Assigning Users to Roles in EJB

```
<security-role-mapping>  
  <role-name>member</role-name>  
  <principal-name>jgarcia</principal-name>  
  <principal-name>mwebster</principal-name>  
  <group-name>team-leads</group-name>  
</security-role-mapping>
```

```
<security-role-mapping>  
  <role-name>customer</role-name>  
  <principal-name>dsmith</principal-name>  
</security-role-mapping>
```



ONLY  
THE BEST  
GET IN

# Assigning Methods to Roles in EJB

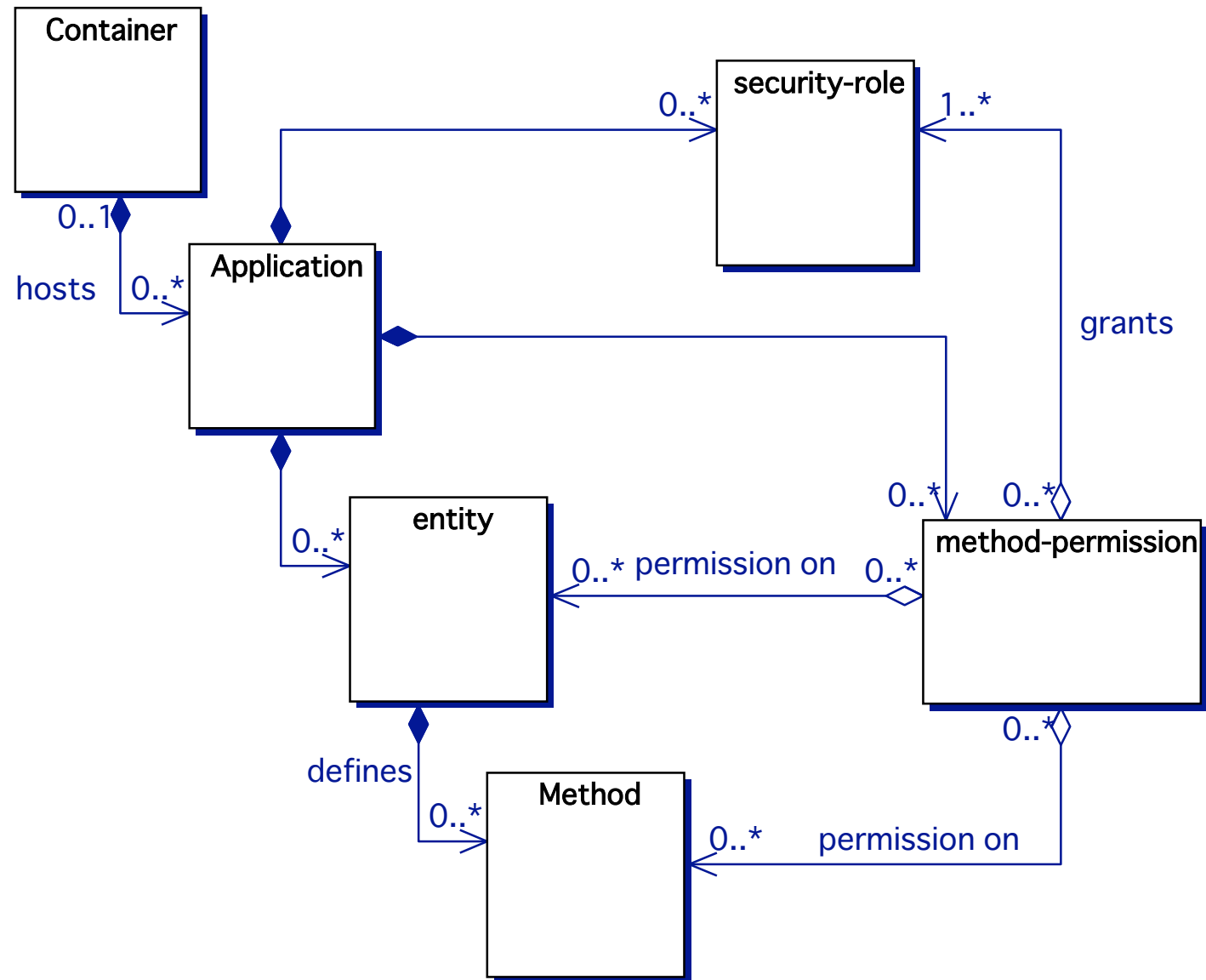
```
<method-permission>  
  <role-name>staff</role-name>  
  <method>  
    <ejb-name>Product</ejb-name>  
    <method-name>*</method-name>  
  </method>  
</method-permission>
```

```
<method-permission>  
  <role-name>customer</role-name>  
  <role-name>member</role-name>  
  <method>  
    <ejb-name>Product</ejb-name>  
    <method-name>getPrice</method-name>  
  </method>  
</method-permission>
```



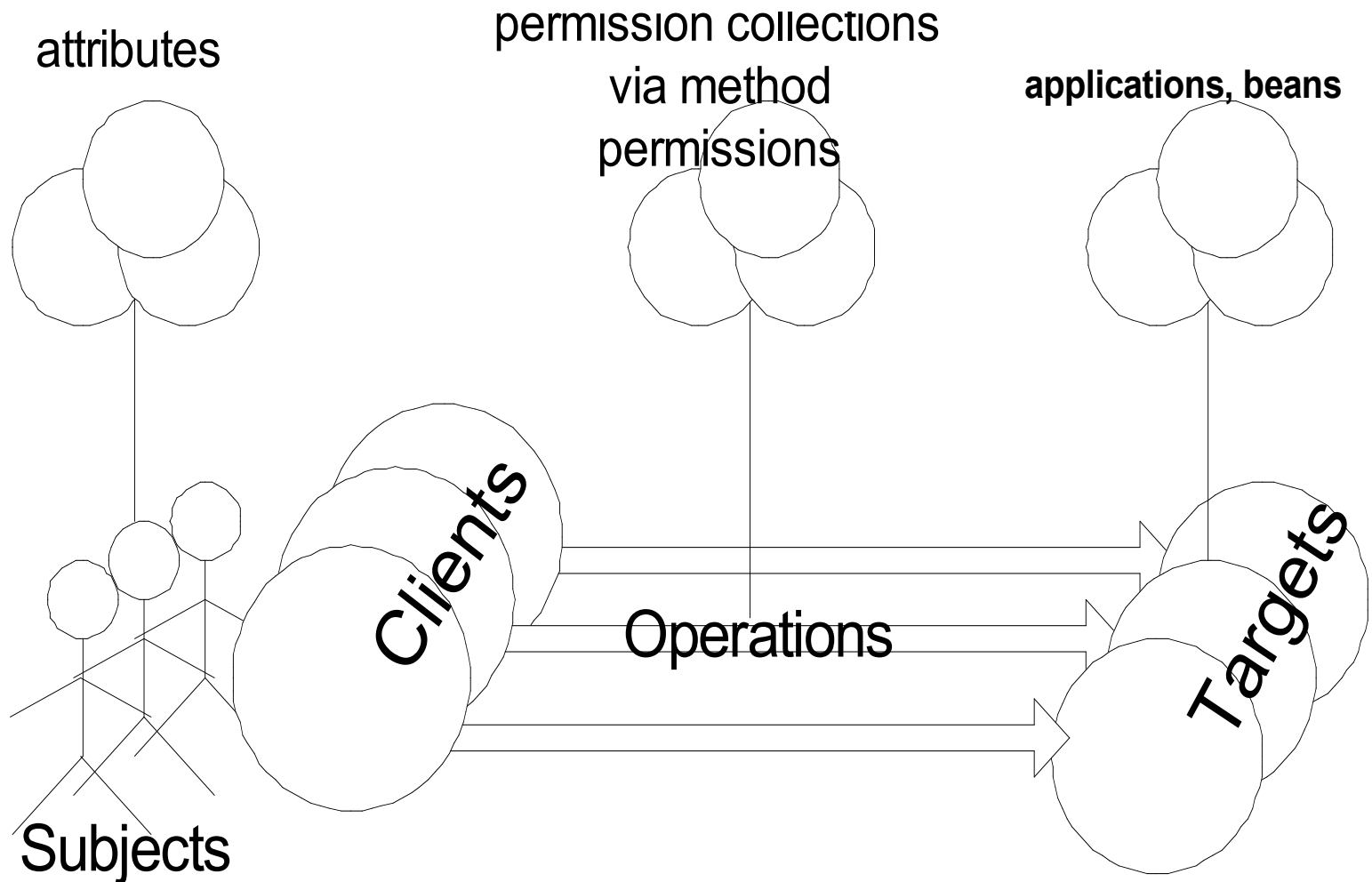
ONLY  
THE BEST  
GET IN

# roles and permissions in EJB



ONLY  
THE BEST  
GET IN

# scaling with collections



ONLY  
THE BEST  
GET IN

# Custom Authorization in EJB

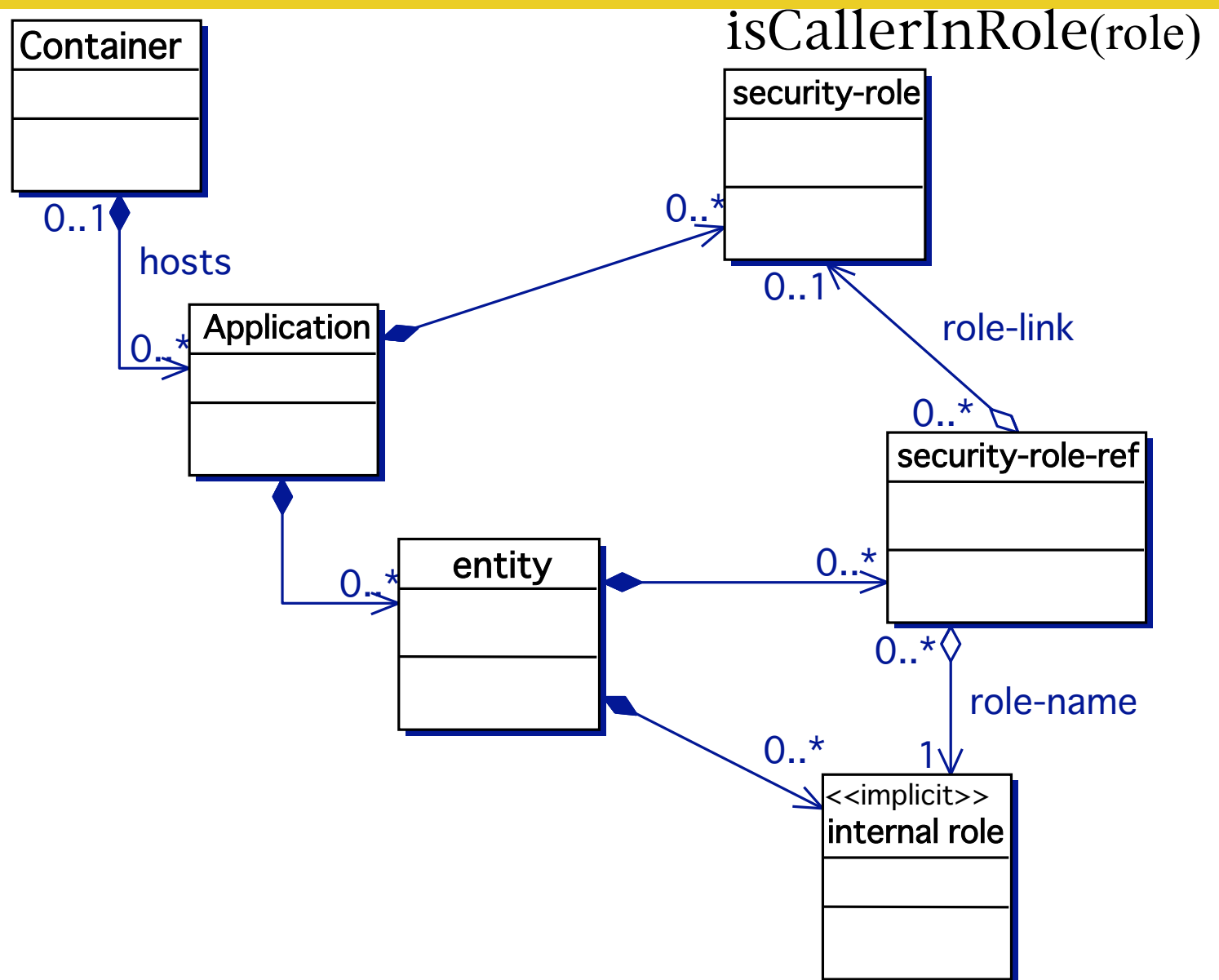
- Java Authorization Contract for Containers (JACC) (formerly known as JSR 115)
  - Part of J2EE v1.4
  - defines an interface for pluggable authorization providers





ONLY  
THE BEST  
GET IN

# Fine-grain authorization in EJB



ONLY  
THE BEST  
GET IN

# evaluating EJB 1/2

- **Granularity**
  - + bean method in application
  - not at bean instance
  - + arbitrary resource with security-role-ref
- **Expressiveness** -- supporting different policies
  - + RBAC<sub>0</sub>
    - RBAC<sub>1-3</sub> -- product specific
- **Richness** -- information for making decisions
  - any user attributes are reduced to roles -- product specific
- **Consistency** -- across multiple applications
  - product specific
  - requires application redeployment, or manual changes in each application instance



ONLY  
THE BEST  
GET IN

## evaluating EJB 2/2

- **Manageability** -- changes to policies, users, appl-s
  - user population -- product specific
  - application population
    - + replication -- easy to use packaging
    - o new -- labor intensive and error prone
  - changes in policies -- labor intensive and error prone
  - + computing environment -- easy to use packaging
- **Scalability** -- performance and admin. scalability
  - subject groups -- product specific
  - + three levels of permission granularity, permission collections
  - local to the application permissions (collections)
- **Object paradigm requirements**
  - + roles isolate administrators from method semantics
  - container co-located instances of the same bean are governed by one policy
- **Extensibility** -- support for unforeseen policies
  - mostly through “programmatic security” inside of application
  - + allows mapping from “external” to “internal” roles
  - + JSR 115 “Java Authorization Contract for Containers” JACC



ONLY  
THE BEST  
GET IN

## further reading on EJB

- Sun, ***Enterprise JavaBeans Specification, Version 2.0***, Sun Microsystems Inc., October 23 2000.
- E. Roman, S. Ambler, and T. Jewell, ***Mastering Enterprise JavaBeans***, Second ed: Wiley Computer Publishing, 2002.
- B. Hartman, D. J. Flinn, and K. Beznosov, ***Enterprise Security With EJB and CORBA***. New York: John Wiley & Sons, Inc., 2001.
- B. Hartman, D. J. Flinn, K. Beznosov, and S. Kawamoto, chapter 7, ***Mastering Web Services Security***, 1st ed. New York: John Wiley & Sons, Inc., 2003.
- Sun Microsystems, ***JSR 115: Java Authorization Service Provider Contract for Containers***, 2002, [http://java.sun.com/aboutJava/communityprocess/jsr/jsr\\_115\\_authorization.html](http://java.sun.com/aboutJava/communityprocess/jsr/jsr_115_authorization.html).



ONLY  
THE BEST  
GET IN

## Conclusions

- Access control capabilities in both COM+ and EJB suck!
- EJB is extensible through JACC

