Access Control for Extensible Systems

Robert Grimm and Brian N. Bershad
University of Washington, Seattle
Goals

- Uniform model
- Flexible and policy-neutral
- Centralized mechanism
- Transparent enforcement
- Aggressive policy-specific optimizations
Initial Design

- Based on domain and type enforcement
- Transparent support
  - Link- and call-time control on extensions
  - Protection domain changes
- Hooks for explicit security on other system services
Experiences

• Transparent access control on extensions
  – Minimizes and isolates changes to system
  – Enables access check optimizations

• Explicit access control on system services
  – Binds security to service implementation
  – Precludes access check optimizations

• Need for complex DTE/DTEL interface
Improved Design

- Separates management from policy
- Uses security identifiers and access modes
- Provides access control transparently and dynamically
- Caches policy decisions to improve performance
Advantages

• Supports different \textit{and} changing security requirements
• Straightforward to make fast
• Applies to other extensible systems (Java)
• Provides test-bed for future research

• \url{http://www.cs.washington.edu/research/projects/spin/www/}
Discussion

- Language support
- Model
- Safety and security
Language Support

• Introspection
• Binary interposition
• Thread management
• Object properties
• Exception handling
Model

- Associate threads and objects with security identifiers
- Map pairs of security identifiers into
  - Access mode
  - Pair of security identifiers
Safety and Security

• Design builds on safety
• Prudent interface design still necessary
  – Type hiding
  – Granularity of interfaces / operations
  – Some features very hard to make secure
• Alternatives to type-safety or SFI