Strong Access Controls with Capabilities

WISAC, September 25, 2000
Alan Bomberger
Agorics, Inc.

Overview

- In 1980 ASP was called timesharing
- Security requirements today are very similar
- The same security technology that worked in 1980 works today - capabilities
The Computing Environment

- Network connected servers
- Competitors sharing the same server
- Proprietary data to be protected
- Large numbers of clients
- User and 3rd party software
- 24x7 availability
- Outside penetration attempts

 Been There Done That

- Privacy of data
- Integrity of data
- Mediated sharing
- High performance
- High reliability

- High availability
- Fast restart
- Penetration resistant
- Security through restarts
Definition of User has Changed

- Millions instead of thousands of users
- Anonymous users
- More diverse participants
  - Suppliers, vendors, customers, researchers
  - Application vendors, data vendors
  - Information brokers
- Client software is more sophisticated
  - Java powered browsers

The Environment has Changed

- Server functions are richer
  - more exposure
- Customer supplied software on server
- Multi-vendor networks with uncontrolled communication paths
The Enemy is Stronger

- Many more hackers
- Automated assault tools
- Denial of service attacks

Requirements for Strong Security

- Small protection domains
- Extended-Type objects
- Single location for policy enforcement
Small Protection Domains
Principle of Least Privilege

- Non-Capability
  - Sendmail runs as root to distribute mail
  - Bugs can misuse root privileges
- Capability
  - Grant sendmail only write access to mailboxes
  - No extra privilege; can only misdirect or fail to deliver mail

Extended-Type Objects

- Non-Capability
  - Each new data type has a protection mechanism
  - New features -> new types -> new mechanism
    - .rhost, .xhost
- Capability
  - Only one mechanism
  - New features -> new types -> same mechanism
Single Policy Enforcement Point

- Non-Capability
  - Each mechanism must have its own enforcement point
  - Each must also trust all others
- Capability
  - All policy is enforced using capabilities
  - Only single path to trust (kernel-enforced)

Pure Capability System

- Compact, efficient kernel and TCB
- System functions built outside kernel based on objects and capabilities
- Factory builds confined objects
- Protection domains defined by capabilities held
Pacific Based on KeyKos

- Object reference with authority - car key
- Adds capability registers
- Adds invocation instructions
- The factory initializes capability registers
- All system actions are invocations
  - I/O requests, exception handling
  - Memory allocation, CPU scheduling

Capabilities achieve Strong Access Controls

- A pure capability-based OS addresses the security needs of shared server solutions through
  - Small protection domains
  - Single air tight mechanism
  - Policy separated from mechanism