Access Control Models, with Applications to Robotics

The European Robotics Forum

Stefan Rass
System Security Research Group (syssec), Institute of Applied Informatics
Alpen-Adria Universität Klagenfurt
stefan.rass@aau.at

March 3rd, 2020

Agenda

- Motivation
- Types of Access Control
- Discretionary Access Control
- Role Based Access Control
- Mandatory Access Control
- Appendix
  - MAC Variants
  - Clark-Wilson Model for Integrity Preservation

Motivation 1

- Application of IT, especially robotics, meant to increase efficiency of repetitive tasks →
  - cost reduction,
  - quality increase

  → competitive advantage
- Robotics is challenging here for
  - being highly distributed
  - involving lots of sensors and actors
  - collaborating or even cooperating with humans → security and safety risks

Motivation 2

- Security is an enabler for safety via classical goals:
  - confidentiality $\rightarrow$ encryption ✓
  - integrity $\rightarrow$ checksums ✓
  - availability $\rightarrow$ redundancy ✓
- But equally important are:
  - non-repudiation $\rightarrow$ secure logging (robot "black-box", block-chain?, ...)
  - authenticity $\rightarrow$ digital signatures (can be expensive and hard to maintain practically), ... ✓
  - authorization $\rightarrow$ many possibilities $\Rightarrow$ this talk
- Security models: technical and organizational procedures for authorization $\neq$ authentication
  - Authentication: proof of a claimed identity ($\neq$ identification $=$ determination of an unknown identity)
  - Authorization: verification of rights that are assigned to a certain identity, whose verification/determination is a separate issue!

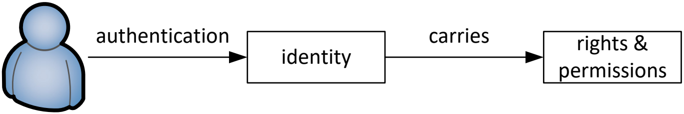
- Security models of different kind were developed in the 1970s and applied ever since then.
- Internal and external attacks lead to:
  - Loss of data and information (deletion, copying, ...)
  - Data manipulation (updates on database records, sending malicious commands to robots ...)
- Complex security rules are inevitable
- Access control systems for large distributed systems are mandatory

Motivation 4

- Access to resources (data, information, sensors, actors, ...) must be constrained/controlled.
  - Not everybody must be allowed to do everything. There are defined rules.
  - Dependent on time, context, privileges, ...
  - Goal in industrial production is the protection of integrity (other businesses may care more for confidentiality).

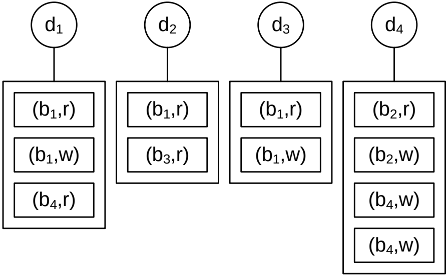
- Multi-user environments
  - Changing users, where user $\in$ {sensor, actor, human operator, ...}
  - Rights/roles of users may change
  - restrict information flow (no information leakage or injection of commands)
  - Access to common resources (e.g., sensor, actor, but also project directories, ...)

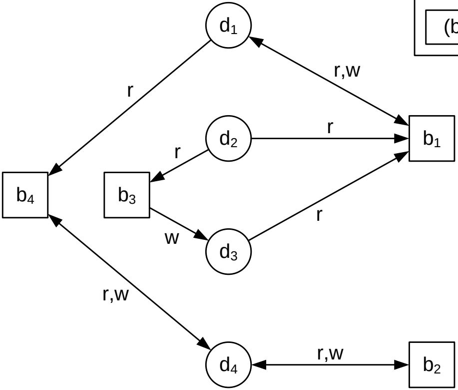
- DAC (User-definable access control) is also called identity based access control (IBAC).
- System consists of
  - Subjects (users , processes, groups)
  - Objects (devices, files, physical objects, ...)
  - Subjects and objects carry unique IDs.
- Owner defines rights on her/his objects.
  - Positive (permissions) and negative (restrictions) rights are possible.
  - Each object has exactly one owner.
  - The owner of an object may change.
  - Rights on an object are granted individually.
- Rights are maintained (enforced) via
  - access control list (ACL)
  - access matrices (rarely used)



- Subjects can grant rights to other subjects.
  - User A calls program B (both are subjects!)
  - Program B inherits the rights of user A.
  - Program B is also an object (a file in general).

- Representation as
  - Graph
  - ACL
  - access matrix



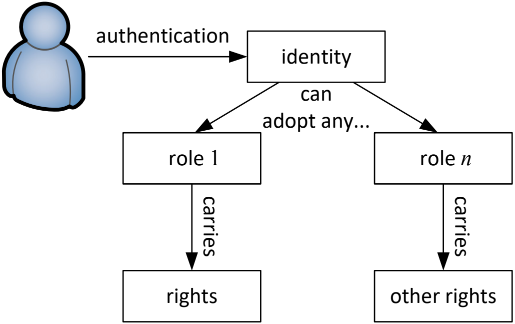


M	d_1	d_2	d_3	d_4
b_1	r,w	r	r	
b_2				r,w
b_3		r	w	
b_4	r			r,w

Discretionary Access Control (DAC) 4

- DAC is implemented in many operating systems.
- (One) problem due to exclusive use of DAC
  - Login-passwords (or hash values thereof) are stored in a file owned by the administrator.
  - Users cannot access that file (otherwise, a user could modify another user's password).
  - If a user wishes to change her/his password, access to the file is required.
- Solution (least privilege)
  - User is temporarily elevated to the administrative role (in the background), so that access to the password file becomes possible. This happens transparently for the user.
  - A pure use of DAC would otherwise require the administrator's intervention to change a password (as this is the owner of the password file).

- Role Based Access Control was first proposed in 1992 by D. F. Ferraiolo and D. R. Kuhn.
- A role (user role, function) defines
  - Duties (specialized sensors, specialized actors, ...)
  - Rights (access to local resources, sensors, physical parts, ...)
- Users adopt and may switch roles ← multiple access credentials

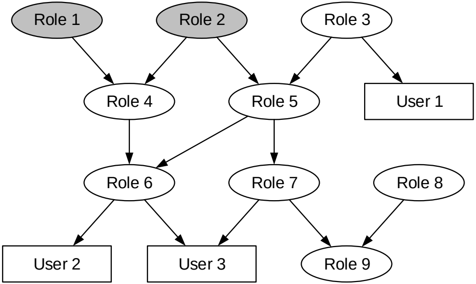


- The assignment of subjects to roles enables:
  - Easy administration (few roles instead of having to adapt many users)
  - Assignment of minimal privileges to fulfil duties (Need-to-Know-Principle).
  - Change of roles (owner of a part may change along a production line)
  - Separation-of-duty principle (e.g., robot must not take commands from sensors not carrying the proper role).
- Roles can inherit from one another.
  - Rights can be inherited (also restricted), e.g., main vs. deputy, ...
  - Complex interdependencies (hierarchies) are easy to model (e.g., worker $<$ supervisor $<$ chief of production).
  - Multiple inheritance possible.
- Popular to grant rights in complex systems.
  - Hospital (doctor, department head, nurse, patient, ...)
  - University (principal, institute head, coordinators, ...)
  - Enterprise (executive board, manager, employee, ...)

Ellipses: Roles

grey ellipses: abstract roles (no physical role representative)

Rectangles: users



In a robotic system, e.g.,

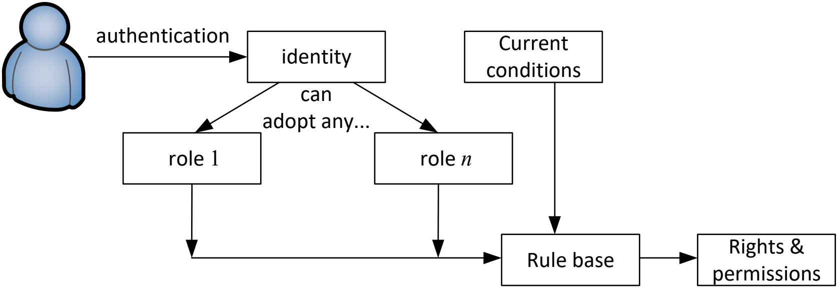
- sensor $<$ emergency sensor (could overrule the "normal sensor")
- robot operator $<$ robot programmer $<$ process administrator

UNIVERSITÄT KLAGENFURT

- Mandatory access control is in the literature also called rule-based access control.
- Decisions are made according to a central and mandatory rule base.
- Rules may specify:
  - Times (access permitted only during work hours)
  - Number of accesses (say, a file can be opened at most 100 times per day)
  - Attributes (ownership, security clearance, ...)
  - Statistical patterns (normal vs. irregular user behavior)
  - Age of objects (e.g., within the last n time units)

- Objects/Subjects get attributes assigned (MAC labels).
  - Security clearance (example: unclassified, ..., top secret)
  - Keywords (example: adminstration, production planning, ...)
  - resource type $\rightarrow$ resource-based access control
  - current role $\rightarrow$ combination with RBAC

- roughly comparable to a combination and refinement of DAC and RBAC



- very flexible, yet (for that reason) also very complex
- Object access constrained by rules (e.g., no use of certain sensors, actors, . . .)
- Popular in file system right management (Linux, AppArmor, various firewalls, . . .)

Comparative summary

- DAC: identity $\rightarrow$ rights and permissions
- RBAC: identity $\rightarrow$ role $\rightarrow$ rights and permissions
- MAC:
    1. identity $\rightarrow$ role
    2. role $+$ current system conditions $\rightarrow$ rights and permissions
- most other schemes: ... viewable as special cases or particular refinements of MAC for certain application contexts

Take-Home Message

- Recommendation: do not rely on "standard" security mechanisms too much (don't just use a password, plan for a highly diverse and changing environment in future)
- Determine the attributes, resources, ... general factors... that determine permissions and prohibitions
- Anticipate changes in future and adapt your access control to be flexibel for that → rule-based access control is often efficient to maintain, update and adapt
- Also check out other (more advanced) access control models: policy-based, attribute-based, risk-level determined, Brewer-Nash (Chinese-Wall), Bell-LaPadula, Clark-Wilson (← used in many databases) ...

[BL73] D. Elliott Bell and Leonard L. LaPadula.
Secure Computer Systems: Mathematical Foundations.
January 1973.

[BN89] David F.C. Brewer and Michael J. Nash.
The Chinese Wall Security Policy.
pages 206–214. IEEE, 1989.

[CW87] David D. Clark and David R. Wilson.
A Comparison of Commercial and Military Computer Security Policies.
pages 184–194. IEEE, 1987.

[Dec17] Mina Deckard.
The Security Requirements for A Global RPA Platform, 2017.
Sep. 14, url: https://www.uipath.com/blog/the-security-requirements-for-a-global-rpa-platform [retrieved: Feb. 24, 2020].

[FK92] D. F. Ferraiolo and D. R. Kuhn.
Role-Based Access Controls.
NIST, 1992.

[FSG+01] David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. Richard Kuhn, and Ramaswamy Chandramouli.
Proposed NIST Standard for Role-based Access Control.
ACM Trans. Inf. Syst. Secur., 4(3):224–274, 2001.

[HM13] Shabnam Mohammad Hasani and Nasser Modiri.
Criteria Specifications for the Comparison and Evaluation of Access Control Models.
pages 19–29, 2013.

[LLWC12] Debin Liu, Ninghui Li, Xiaofeng Wang, and L. Jean Camp.
Beyond Risk-Based Access Control.
In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, and George Danezis, editors, *Financial Cryptography and Data Security*, volume 7035 of *Lecture Notes in Computer Science*, pages 102–112. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[Sch19] Maria Schuett.
Robotic Process Automation Meets Identity and Access Management.
ISSA Journal, July:22–28, 2019.

[SP10] Ebrahim Sahafizadeh and Saeed Parsa.
Survey on access control models.
In *2010 2nd International Conference on Future Computer and Communication*, pages V1–1–V1–3, Wuhan, China, 2010. IEEE.

[ZW16] Yunpeng Zhang and Xuqing Wu.
Access Control in Internet of Things: A Survey.
arXiv:1610.01065 [cs], October 2016.
arXiv: 1610.01065 version: 1.

Access Control Models, with Applications to Robotics

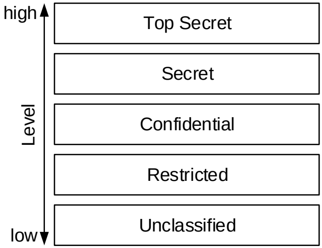
The European Robotics Forum

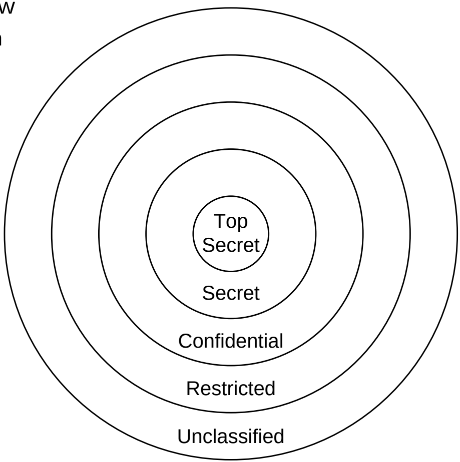
Stefan Rass
System Security Research Group (syssec), Institute of Applied Informatics
Alpen-Adria Universität Klagenfurt
stefan.rass@aau.at

March 3rd, 2020

Questions?

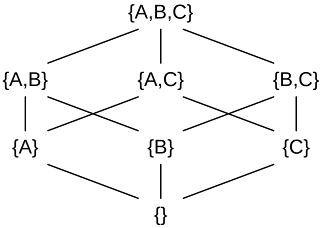
- multi level security – MLS
  - Simplest variants (1970s)
  - Classify data in terms of security levels
  - Limitation of information flow
  - Vertical classification of data
  - Example: Biba-Model

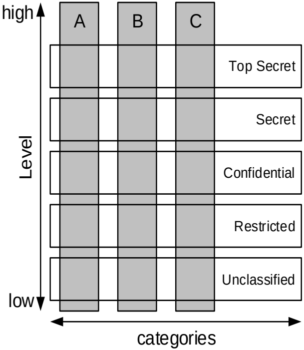




- Multilateral security models
  - A.k.a. rule based security models
  - More complex than multi-level security models
  - Use vertical and horizontal classification
  - Example: Bell-LaPadula-Model

Subset-relations among categories induces a lattice





Clark-Wilson – Basics

- Developed in 1987 by David Clark and David Wilson.
- Multilateral security model
- Goal is the protection of integrity (integrity policy)
- Strongly different to the Bell-LaPadula- and Biba model.
- The model uses transactions as basic operations.
- The CW model is – in principle – implemented in every database management system (DBMS).
  - Assumption: DBMS supports transactions
  - Examples: Oracle, MySQL, MS SQL, ...
- All subsequent explanations will refer to DBMS.

- Constrained Data Items – *CDI*
  CDIs are data entities that are subject to integrity conditions.

  Examples:
  - Foreign key entries
  - Not-null conditions (columns)
  - Tables or columns protected by stored procedures

- Unconstrained Data Items – *UDI*
  UDIs are arbitrarily malleable data entities.

  Examples:
  - Database records without integrity constraints
  - Intermediate results

Clark-Wilson – Components 2

- Integrity Constraints – *IC*
  ICs define conditions under which *CDIs* are considered as valid.

  Examples:
  - Foreign key value (referred record must exist in the other table)
  - Not-null conditions

- Integrity Verification Procedures – *IVP*
  IVPs are programs that check integrity. Violations of constraints initiate a rollback.

  Examples:
  - Parts of a DBMS that enforce constraints
  - INSERT/UPDATE/DELETE stored procedures

- Transformation Procedures – *TP*
  TPs are programs that modify *CDIs* while retaining the integrity

  Examples:
  - INSERT/UPDATE/DELETE commands
  - Transactions (set of INSERT/UPDATE/DELETE commands)
  - Stored procedures

- Certified Relation – *C*
  $C \subseteq TP \times CDI$
  The relation defines which *TPs* may modify which *CDIs*.

  Examples:
  - Write protection of tables ($TP =$ INSERT, $CDI =$ affected table)
  - Protection against unknown stored procedures.

Clark-Wilson – Components 4

- Users – U
  U represent instances using the system.

- Allowed – A
  $A \subseteq U \times TP \times CDI$
  Is a relation that defines which user can run which TP on which $CDIs$.
  A is often maintained as a separate table inside the DBMS and is under extra protection.

  Examples:
  - User "A" may access table "Mitarbeiter" using the TP UPDATE.
  - User may read and write booking records.

- Certification rules
  - CR 1 (Integrity protection)
    The *IVPs* must assure that all *CDIs* remain in a consistent state at all times
  - CR 2 (Transaction execution)
    TPs modify *CDIs* from one consistent state into another consistent state.
  - CR 3 (Separation of duties)
    The relation A must satisfy the separation-of-duty-principle. For example, a cashier must not update the table with the booking records
  - CR 4 (Recovery after failure)
    TPs must log their activities in an (append-only) file.
  - CR 5 (Correct insert of data)
    TPs that process *UDIs* must convert these into valid *CDIs* or otherwise do nothing.

- Enforcement Rules
  - ER 1 (Write-protected areas)
    TP t can operate on *CDI* c only if $(t, c) \in C$.
  - ER 2 (Access rules)
    A user u can access *CDI* c via *TP* t only if this is permited by A, i.e., $(u, t, c) \in A$
  - ER 3 (User login)
    Users must be authenticated before they can run any *TPs*.
  - ER 4 (Privilege escalation)
    A user who can modify relation C (or parts of it) must not gain rights to run any affected *TPs*. Thus, nobody can grant additional rights to her/himself.