The CRUD Security Matrix: A Technique for Documenting Access Rights

Dale L. Lunsford, Michael R. Collins
Phillips School of Business
High Point University

Abstract

The CRUD matrix is an excellent technique to model processes and data and how they interact with respect to creation, reading, updating, and deleting of the data. In this paper, we extend the CRUD matrix to a CRUD Security Cube where we propose incorporating a third dimension on the matrix to include individuals or groups and the rights and security privileges granted to each. This additional dimension on the matrix provides significant information without using an additional model or losing any information from the original CRUD matrix in its design. Analysts may generalize the application of this extension to databases, information systems, or literally any information system's object that incorporates data, processes, and how individuals may interact with those within the object.

Keywords: *CRUD matrix, CRUD Security Cube, access controls, access rights, permissions*

Organizations need to ensure that each employee has the appropriate access to information, but does not have excessively powerful access rights. As systems become more complex, organizations grow in size, and the use of contractors and temporary employees increase, the difficulty of ensuring that only authorized users have access to information increases. Kamens (2007) describes challenges faced by auditors and organizations when a company hires, fires, loses, or moves employees. Given the size of staff and the quantity of information employees need access to, regulating and monitoring this access is difficult. At the same time, new laws such as Sarbanes-Oxley Act place greater importance on this. Kamens suggests that companies develop pre-defined specifications for the access rights employees need to do their jobs. Essentially, Kamens states the need for access control entries and access control lists as described by Govindavajhala (2006). While products exist to manage access rights (Hulme, 2003), first organizations must carefully define the access rights needed.

This paper outlines a technique for defining the appropriate access rights using an extension of the CRUD (create, read, update, and delete) matrix, referred to as the CRUD Security Cube in this paper. The CRUD Security Cube adds a third dimension in which analysts can model users and groups and their access rights. Incorporating this third dimension of information within the existing CRUD matrix is significant in aiding information technology professionals with modeling how access rights apply to both processes and the data within an organization. In practice, this cube extension to the

CRUD matrix allows analysts to view slices of the matrix. Depending on the informational needs, the analyst could 'slice' the cube as needed to find relevant data on any or all of the three dimensions modeled in the CRUD Security Cube. The user-dimension extension in the CRUD Security Cube applies to access rights in a variety of systems, including software suites, applications, or database management systems.
The organization of this paper is as follows. First, this paper describes the structure and role of the CRUD matrix. Second, this paper outlines techniques for specifying access rights in information system settings. Next, this paper describes the CRUD Security Cube in general terms. Finally, this paper discusses potential benefits of the CRUD Security Cube and outlines extensions to this line of research.

The CRUD Matrix

The traditional CRUD matrix is one popular approach to modeling data and process interactions enterprise wide (Politano, 2001). Knowing what processes Create, Read, Update, and Delete (CRUD) what data assists the database designers as well as the systems analysts within organizations in storing data and creating processes that effectively manage and manipulate specific data (Oppel, 2004). Figure 1 is an example of a CRUD matrix.

Figure 1: Sample CRUD Matrix

	Customer	Customer Order	Customer Account	Customer Invoice	Vendor Invoice	Product
Receive Customer Order	R	C	CR			
Process Customer Order	CRU		RU			R
Maintain Customer Order	U		U		RU	
Terminate Customer Order	U		U		RU	
Fill Customer Order	RU		RU			RU
Ship Customer Order			U		C	
Validate Vender Invoice					R	
Pay Vender Invoice					RU	
Invoice Customer	RU		RU	C		
Maintain Inventory						CRUD

Extracted from: (Borysowich, 2007)

As you can see from the example above, the CRUD matrix provides us with a two-dimensional representation of data and processes and the interaction that exists between them with respect to creating, reading, updating, and deleting of data. As businesses move into the 21st century, more and more emphasis is being placed on security in terms of individual and group access rights with systems and data access to specific processes.

Controlling Access to Resources

Access rights, also known as permissions or privileges, define the types of access that a user or group has to a securable object. Common options for defining access rights include individual users and groups. Unix identifies three recipients of access rights: the object's owner, a group, and the world (December, 2008). Starting with the NT File System (NTFS), users and groups may be recipients of access rights (Melber, 2006) under Microsoft Windows. In many settings, it is more efficient to define access rights in terms of groups, rather than individual users, and then assign individual users to groups. This simplifies the security management process as needs change. In this case, users with extensive access needs are members of multiple groups. The nature of securable objects varies across systems; common securable objects include directories and files, devices, executables, and other objects (Changing Access Security on Securable Objects, 2008). While access types vary across operating systems, common access types include full control, modify, read & execute, read, and write under NTFS (Melber, 2006; Eckel, 2007) and read, write, and execute under Unix (December, 2008). As illustrated in Figure 2, NTFS provides extensive advanced access types, depending on the securable object (Mullins, 2006).

Figure 2: NTFS Advanced Access Types (Mullins, 2006)

Traverse Folder/Execute File
List Folder/Read Data
Read Attributes
Read Extended Attributes
Create Files/Write Data
Create Folders/Append Data
Write Attributes
Write Extended Attributes
Delete
Read Permissions
Change Permissions
Take Ownership

NTFS provides advanced mechanisms to extend access rights, including inheritance and the ability to deny access (Melber, 2006; Mullins, 2006; Eckel, 2007). In NTFS, specification of access rights is either explicit or inherited. Securable objects such as files and directories may inherit access rights from parent securable objects, or may receive access rights through explicit assignment. Additionally, NTFS provides a mechanism to deny a user or group any particular access type. In NTFS, each access

right forms an access control entry (ACE) and the collection of access control entries form an access control list (ACL) (Govindavajhala, 2006; Introduction to Securing Your Windows Computer Files, 2001).

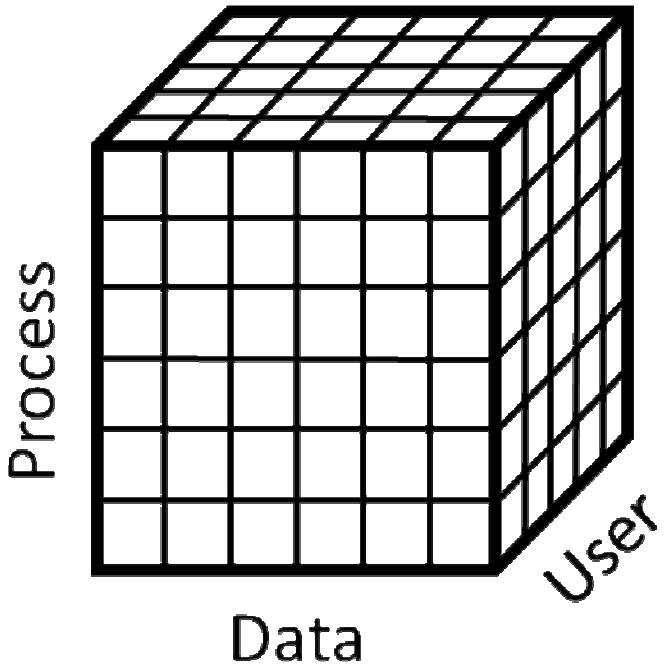
Ferraiolo (1992) and Ferraiolo (1995) describe three types of access controls: mandatory access controls (MAC), discretionary access controls (DAC), and role-based access controls (RBAC). Under DAC, users may grant or deny access to objects under their control. DAC is the weakest form of access control; however, it is appropriate in many settings. With MAC, before a user gains access to an object, the user must receive formal clearance to access the object. An administrator is responsible for clearing the user based on the sensitivity of the information in the object and the authorization level of the user. The user does not receive authorization to grant other users access to the object. RBAC is a type of MAC. In RBAC, the user has a specific role to perform in the organization. An administrator determines the transactions associated with the role. Each transaction has associated objects and types of access to these objects. From this, the user's roles specifically determine the types of access the user has to various objects.

Challenges implicit with all three types of access controls include identifying and documenting the appropriate access rights associated with users and groups. In many cases, administrators employ a trial-and-error strategy that may result in over or under assignment of access rights. In the case of under assignment of access rights, users are unable to perform necessary job functions until an administrator resolves the problem. With over assignment of access rights, the consequences have the potential to be more significant, enabling users to perform undesired actions or resulting in the disclosure of information (Kamens, 2007). Alternatives to the trial-and-error approach include employing use cases (Firesmith, 2003), extensions to the Unified Modeling Language (UML) to incorporate security information (Breu, 2007), and the use of RBAC (Ferraiolo, 1992; Ferraiolo, 1995). Additionally, conventional data flow diagramming techniques provide sufficient information to identify necessary access rights.

The CRUD Security Cube

The CRUD Security Cube extends the standard CRUD matrix by adding a third dimension representing users or groups of users (Figure 3). Using this dimension, system analysts and security analysts can document the appropriate access rights for users or groups to processes and data. The CRUD Security Cube is appropriate when designing security for information system applications that employ processing units such as programs, data access forms, or similar objects to access data in a database. Additionally, this specification is appropriate in any setting where the user employs specific programs to access data objects; this provides a mechanism for documenting the programs that users need access to and the data that the programs access.

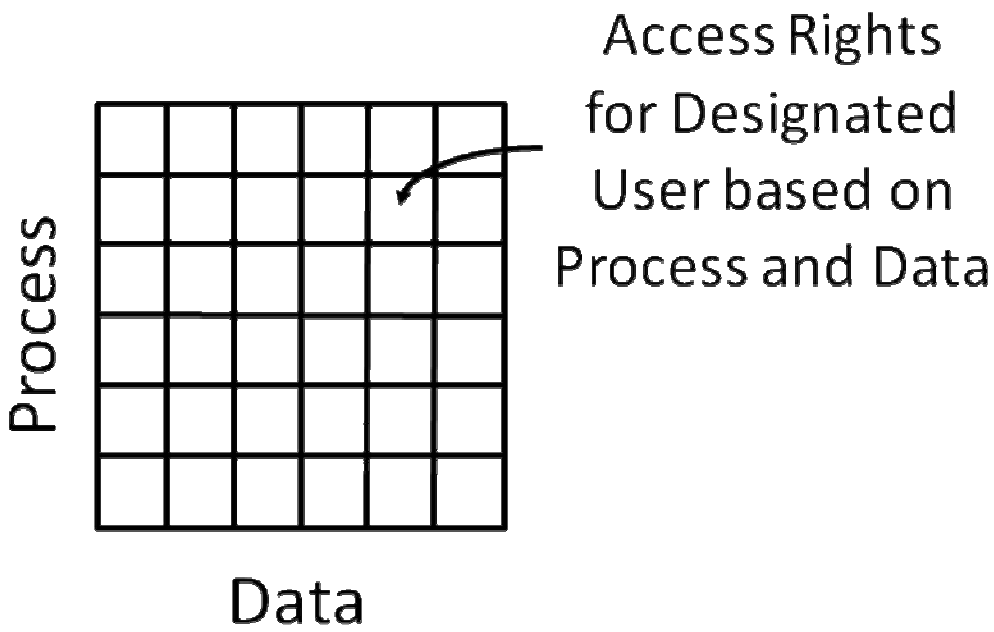
Figure 3: CRUD Security



Cube

Figure **4** illustrates a user slice from the CRUD Security Cube. The user slice provides a mechanism for clearly depicting the access rights for a single user or group. Essentially, the set of user slices comprise the CRUD Security Cube.

Figure 4: User Slice



In addition to the user slice, by rotating the cube, one may look at access rights from a variety of perspectives. A process slice highlights the types of access that users have to data objects utilizing a designated process. A data slice highlights the types of access that users have to process objects for a given data object.

The CRUD designations provide a convenient mechanism for the initial specification of access rights. Systems analysis models such as data flow diagrams or use cases help to identify the roles that users or groups play in the system, thus determining the types of access required to processes and data. Depending on the specific capabilities of the applications or database management system, more refined access type specifications may be possible beyond the CRUD access types.

The process for developing the CRUD Security Cube will vary depending on the setting. In an information system setting, the process will consist of the following major steps.

The systems analyst develops standard analysis models depending on the method employed.

The systems analyst creates a CRUD matrix that maps processes and data.

Using the CRUD matrix and analysis models such as data flow diagrams or use cases, the systems analyst identifies the users or groups employing the processes to access data, yielding a collection of user slices.

The systems analyst and security analyst determine the appropriate user, process, and data interactions required in the system, in terms of the types of access required. The analysts record this information on the user slices.

Benefits of the CRUD Security Cube

There are a number of potential benefits from the CRUD Security Cube technique for defining and documenting access rights. A significant benefit is that this technique presents access rights in an easily understood tabular format. Second, this technique provides a means of clearly depicting access rights prior to the implementation of these access rights. Third, the advanced documentation of access rights provides a baseline for security analysts and auditors to assess the implementation of the access rights to ensure an adequate level of information and system protection. Fourth, this technique does not rely on a specific analysis method and should work well with structured and object oriented methods. Fifth, this technique is adaptable to various types of securable objects, including information systems, directory structures, workstation resources, network resources, as well as others. Finally, as described earlier, different systems provide different access types for objects. The CRUD access types provide a good foundation for specifying access types; however, the CRUD Security Cube technique is easily extensible to various types of access such as those available in NTFS.

Potential Extensions to this Research

This paper presented an outline of the CRUD Security Cube. Initially extensions to this research will focus on the proof of concept. To demonstrate the value of this technique, two proof-of-concept cases are appropriate. One proof-of-concept case will focus on the use of the CRUD Security Cube for documenting access rights for an information system. The second proof-of-concept case will document access rights for user documents on a

computer running the Microsoft Windows Vista operating system with an NTFS file system. A third practical extension to this research is to develop a tool for constructing the CRUD Security Cube with functions to set and audit access rights on a target system.

Works Cited

Borysowich, C. (2007, May 18). Data Driven Design: Other Approaches. Retrieved from ITToolBox.Com: http://blogs.ittoolbox.com/eai/implementation/archives/data-driven-design-other-approaches-16350

Breu, R. P. (2007, October). Model based development of access policies. International Journal of Software Tools for Technology Transfer , 9 (5-6), pp. 457-470.

Changing Access Security on Securable Objects. (2008, February 14). Retrieved February 26, 2008, from MSDN: http://msdn2.microsoft.com/en-us/library/aa384905(VS.85).aspx

December, J. (2008, January 21). Permissions. Retrieved February 22, 2008, from December.com: http://www.december.com/unix/tutor/permissions.html

Eckel, E. (2007, January 22). How do I... Secure Windows XP NTFS files and shares? Retrieved February 7, 2008, from TechRepublic.com: http://articles.techrepublic.com.com/5100-10877_11-6152061.html

Ferraiolo, D. a. (1995, December). An Introduction to Role-Based Access Control. Retrieved February 23, 2008, from NIST.gov: http://csrc.nist.gov/groups/SNS/rbac/documents/design_implementation/Intro_role_based_access.htm

Ferraiolo, D. a. (1992). Role-Based Access Controls. National Computer Security Conference (pp. 554-563). Baltimore: National Institute of Standards and Technology/National Computer Security Center.

Firesmith, D. G. (2003, May-June). Security Use Cases. Journal of Object Technology , 2 (3), pp. 53-64.

Govindavajhala, S. a. (2006, January 31). Windows Access Control Demystified. Retrieved February 26, 2008, from www.cs.princeton.edu: http://www.cs.princeton.edu/~sudhakar/papers/winval.pdf

Hulme, G. V. (2003, January 20). Businesses Get Tools to Manage Access Rights. InformationWeek , p. 26.

Introduction to Securing Your Windows Computer Files. (2001, August 22). Retrieved May 27, 2005, from Stanford Windows Infrastructure: http://windows.stanford.edu/docs/IntroSecurity.htm

Kamens, M. (2007). Making user access policies work for you. Network World , 24 (12), 33.

Melber, D. (2006, May 3). Understanding Windows NTFS Permissions. Retrieved January 25, 2008, from WindowsSecurity.com: http://www.windowsecurity.com/articles/Understanding-Windows-NTFS-Permissions.html

Mullins, M. (2006, June 15). Windows 101: Know the basics about NTFS permissions. Retrieved June 19, 2006, from TechRepublic.com: http://techrepublic.com.com/5102-1009-6084446.html

Oppel, A. (2004). Databases DeMystified. Emeryville: McGraw Hill.

Politano, A. L. (2001). Salvaging Information Engineering Techniques in the Data Warehouse Environment. Information Science , 35-44.

Swift, M. M. (2002). Improving the Granularity of Access Control for Windows 2000. ACM Transactions on Information and System Security , 5 (4), 1-44.

Swift, M. M. (2001). Improving the Granularity of Access Control in Windows NT. Proceedings of Sixth ACM Symposium on Access Control Models and Technologies, (pp. 87-96). Chantilly.