**Week 3: Using Roles**

Nathan Bachmeier  
CMGT 430: Enterprise Security

University of Phoenix   
September 1st, 2014

# Using Roles

McBride Financial Services has completed their initial assessment of security principals, and is now ready to asset their user and permission management strategy. The senior leadership would like to understand weaknesses in their current implementation and how risk could be further reduced. To gain this understanding it is imperative to gain insight into separation of duties, RBAC, distributed trust, and the value these concepts provide.

# Separation of Duties

In *mom and pop shops*, it is common for the single one person to champion requirements for multiple departments or responsibilities. For instance Sally might be the HR representative and also in charge of payroll services. In these smaller organizations they are able to operate with more global permissions due to the eased of auditing and visibility.

As the organization grows in size this becomes more complex and the carefree catch ups at the water fountain do not scale. Image that same Sally in a Fortune 100 enterprise, there are no checks and balances between these two operations, introducing unnecessary risk.

Sally is a model employee, but what about Todd? Todd has the ability to create expense reports, approve (execute) them, and access to auditing records (write). Unless Todd has been vetted by background checks and frequent audits, his employer might find their money gone without a trace.

In these and similar scenarios it is critical to isolate group membership with an intended set while also excluded intersects of unintended parties (Grimaldi, 2004). If the security context cannot be assured the system will be left in an arbitrary and undefined state. Unless context can defined it is impossible to reliably make any proceeding security decisions (Conklin, White, Williams, Davis, & Cothren, 2012).

# Segregate Data and Access

Pretend that an actor has broken into the backend data store and is egressing the data to their servers. The game is lost, check mate so to speak; luckily our design used an assume breach mindset and did not leave all *the eggs in one basket*. While that actor was stealing the customer account information, they could not also get the customer’s credit card numbers and Personally Identifiable Information (PII). This was due to the proper separation of data (state).

State integrity suffers the same challenges of separation of duties, where access to one collection of data should not directly grant access to another one (Whitman & Mattord, 2010). Take for instance a file server that holds both low risk and high risk documents. If someone were to compromise that file server they would have control over both sets of documents. Changing the state of this information could lead to an elevation of privilege vulnerability resulting in complete compromise of the environment (Conklin, White, Williams, Davis, & Cothren, 2012).

To mitigate this, state needs to be properly secured through isolation policy (Hernandez, 2012). Each storage device needs to be assigned a trust level. Data which is of trust level X cannot be stored with data of trust level Y, or else the containing data store with have a trust level of both (X∪Y).

Similar to the isolation of security context isolation previously discussed, it is imperative that storage context be properly secured. Without it state can no longer be trusted resulting in an arbitrary and undefined state. If the state cannot be defined it not possible to reliably make future security or runtime decisions.

# Role Based Control

In an enterprise environment it is not uncommon to have over 10, 000 different legitimate entities within an environment. This includes every domain joined desktop, laptop, and printer; along with every full time, partner team, or temporary employee login; service, automation, and virtual accounts; etc (Morimoto, 2010).

Addressing a large volume of entities is a complex and error prone. To perform this action more reliably larger containers need to exist (Morimoto, 2010). For instance maybe accounting is grouped as cluster one while legal is grouped as cluster two. Once a container has been defined, policy can be targeted at the container and propagated to its children. This enables eased deployment of executive, diagnostic, restrictive, and other compliance or remediation business policy.

Role Based Access Control (RBAC), is a natural extension of this idea where these entity containers include actors of a certain similarity (Ng, 2014). The similarity based on permission to perform an action, or reason for exclusion. For instance all of the senior management in financing should be permitted to approve a payroll expense. Meanwhile none of the temporary staff should not.

# Distributed Trust Management

Consider, a scenario where Abby is introduced Charlie by Betty; if Abby trusts Betty; Betty trusts Charlie. Should Abby trust Charlie? More importantly to what degree, they are a business partner though that must not give them keys to the kingdom.

## Federated Authentication

The first issue that must be addressed in a distributed trust scenario is the identity of the external entity. Charlie might send his friend David over to Abby’s business, and his identity will need to be verified. Depending on the required level of security this could be as informal as the pass phrase, “Charlie sent me,” to a biometric marker (Whitman & Mattord, 2010).

The Achilles heel to this design is that identity information is coming from Charlie. What assurances did Charlie make of David to ensure he’s not a crook? If Charlie did not perform proper checks, David could use the clearance levels of Charlie or Betty to gain entry to a secure area (Hernandez, 2012). This violates security restrictions and leaves the system in an arbitrary and undefined state.

To reduce the impact of this attack vector, limited rights need to be given to the external user until they have been cleared by internal requirements. The addition of an external entity to a role must result in idempotent level of security (Erl, Carlyle, & al., 2014). For example a group of *XYZ Certified Employees* must not allow the addition of a noncertified member.

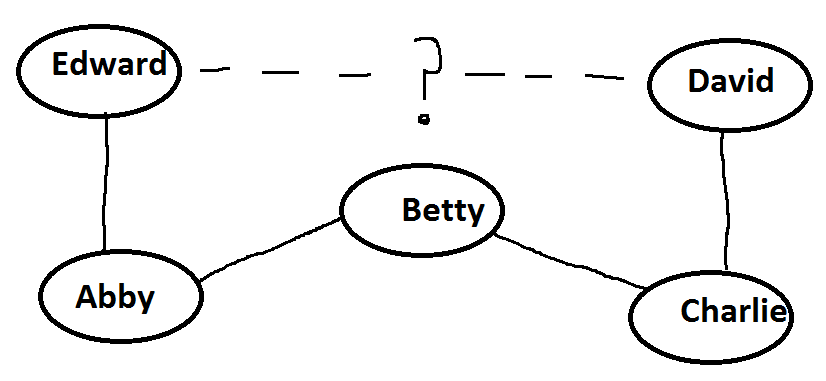
## Federated Rights

A potential vulnerability might be a member of the Charlie friends (domain members) uses the transitive trust to perform an unauthorized action on Abby’s house (domain). For example maybe Charlie’s friend is permitted to set things on fire (claims based rights), that should not implicitly allow the same action at Abby’s house.

Perhaps Abby is willing to let some of Charlie’s friends set things on fire, and can white list them into a security group. This group can then have role based controls applied to it, and be treated similar to other internal groups.

Tradeoffs between degree of control and easy of management need to be made when importing federated groups. When opaquely accepting an entire group isolation needs to be reviewed and cleared with compliance groups. Their rights must be restricted such that they can only access the explicitly required resources.

## Being the Intermediate Node (Abby)



**Figure 1: Logical Layout**

Another serious attack can exist where David gets Abby to vouch for him in a claim toward Edward. Perhaps David has tricked Charlie and is using the connections to Betty to get Abby. Now that Abby has provided some level of clearance Charlie is able to communicate with Edward.

Poor Abby, is made an intermediate to a spoofing, repudiation, or elevation of privilege attack (Hernan, Lambert, Ostwald, & Shostack, 2006). Within an enterprise this can occur when a certificate authority signs a document without reading and validating the contract.

One more example might include cross site response forging/scripting (XSRF/XSS) vulnerabilities. This can occur when David is allowed to communicate with Abby, and then exploits that relationship to cause Abby to perform an action on his behalf.

This attack is common in RESTful environments, where hypermedia has turned networks in a distributed heap and the Uri is just an evolved pointer (Erl, Carlyle, & al., 2014). For example if David can get Abby to invoke //Edward/Expense/1million/to/David, he can quit and retire to a nice second world country.

# Conclusions

Managing large numbers of entities is a complex task, which requires encapsulation of containers. Entities within these containers need to possess some similarity, so they can be used as targets of business policy. These similarities can be used as part of a role based access control system.

Once these security boundaries have been defined they need to be enforceable by context isolation, data isolation, and validation assurances. These challenges become more difficult as the environment becomes more distributed. Thought needs to be given to trust models especial as transitive relationships are leveraged.

# References

Conklin, A., White, G., Williams, D., Davis, R., & Cothren, C. (2012). *Principles of Computer Security: CompTIA Security+™ and Beyond (Exam SY0-301), Third Edition.* McGraw-Hill Company.

Dennis, A., Wixom, B., & Roth, R. (2012). *System Analysis and Design, Fifth Edition.* John Wiley & Sons Inc.

Erl, T., Carlyle, B., & al., e. (2014). *SOA with REST: Prinicpals, Patterns, & Constraints for Building Enterprise Solutions with REST.* Peason Education.

Grimaldi, R. (2004). *Discrete and Combinatorial Mathematics: An Applied Introduction, 5e.* Pearson Education.

Hernan, S., Lambert, S., Ostwald, T., & Shostack, A. (2006). *Uncover Security Design Flaws Using the STRIDE Approach*. Retrieved from MSDN Magazine: http://msdn.microsoft.com/en-us/magazine/cc163519.aspx

Hernandez, S. (2012). *Official Guide to the CISSP CBK, Third Edition.*

Morimoto, R. (2010). *Windows Server 2008 R2 Unleased, 1st Edition.* Pearson Education, Inc.

Ng, A. (2014). *Machine Learning.* Stanford.

Whitman, M., & Mattord, H. (2010). *Management of Information Security; 3rd Edition.* Cengage Learning.