Commercial Solutions for Classified (CSfC), Risk Analysis

Christopher Martinez[1], Robert Haverkos[2]

[1]Purdue University, Marti606@Purdue.edu, [2]Purdue University, RHaverko@Purdue.edu

Keywords: Interdependence of Risk, Risk Monitoring, Interdependency Modeling, Risk Modeling, Risk Analysis, Risk Management, Defense-in-depth, Layered Solutions, CSfC

The goal of the project is to develop a meaningful method of combining risk assessments (for individual security mechanisms) into a risk assessment for the overall capability package (the Layered Solution).

Table of Contents

Executive Summary

Commercial Solutions for Classified (CSfC), Risk Analysis

The transmission of classified (or highly sensitive) data requires a high degree of assurance. This is known as the problem of data-in-transit [2]. Oftentimes, this assurance is in excess of the implemented equivalents in the private sector [8]. Because of this, the assurance that is provided for classified data generally requires several layers of formal accreditation by various regulatory agencies [1]. Traditionally, the only means to achieving this level of assurance has been through the adoption of Government-Off-The-Shelf (GOTS) solutions. However, these specialized hardware and software arrangements require intensive resources to develop and implement. GOTS solutions also typically see limited Return on Investment (ROI) due to the isolated nature of their deployment [9]. It is for these reasons (among others presented in this article) that the National Security Agency (NSA) Central Security Service (CSS) established the Commercial Solutions for Classified (CSfC) Program.

The Commercial Solutions for Classified (CSfC) Program aims to leverage the economy of scale through expediting the deployment of commercial solutions in place of GOTS solutions. In doing this, commercially available mechanisms are assembled into *capability packages* (representing a layered solution to assurance in commercial products). However, the challenges for analyzing the risk associated with these solutions immediately become present. Where risks and vulnerabilities may be straightforward to assess for a single mechanism - the interaction of the risks (and more importantly, the overall risk landscape of the problem) become much more complex to define as these mechanisms are layered together. While it is simple to assume that (intuitively) these layered solutions (which individually offer adequate protection) will have a higher degree of security than any of their individual mechanisms, a secondary problem exists in quantifying this gain and making a meaningful judgment as to the precise level of assurance offered by the solution as a whole. With this in mind, is it conceivable to accurately model risk in a layered solution? If so, what is the most appropriate technique to modeling this risk?

The goal of the project has been to develop a meaningful method of combining risk assessments for individual security mechanisms into a risk assessment for the overall capability package (the layered solution). In this paper, we will present a model for the measurement and sharing of residual risk in layered solutions of the CSfC program. This model provides a means to viewing interaction in risk assessments among layered mechanisms; including the complexities of layered solutions.

1. Detailed Problem Description and Scoping of the Original Abstract

We began this project with an abstract from our technical directors at the Information Assurance Directorate (IAD) of the National Security Agency (NSA). This abstract outlined the desire for a model that would incorporate the measurement of residual risk (the risk associated with the use of any security solution) and the sharing of the residual risk findings with stakeholders in a more meaningful way. Other requests for this project included applying this risk measurement approach to a *two independent layer* situation (we will better define this later as *the Layered Solution*). This two independent layer situation is, in its most simplistic definition, a layering of multiple mechanisms that each provide an adequate (as defined by security requirements) protection. Furthermore, the two independent layer situation would need to be represented in the model with the incorporation of new risks that have relatively little significance to each individual mechanism, but impact the overall solution (or have added significance when considered in a two independent layer situation). After discussing these concepts (and their applied practices) with our technical directors, it was decided that it would be best to redefine the original abstract into a more specified research question. Simply, the abstract did not provide a stipulated area of research that could be completed by two individuals in a 12 week period.

After discussing these requirements of the project, we (the researchers of this project) agreed that the deliverable should be a model that employed the two independent layer situation (as specified by the technical directors and original abstract). We both felt as though this component (of the overall model) was crucial to the success of the model in fulfilling the needs of our technical directors. With this in mind, we could then begin considering the science of risk analysis. After discussing risk analysis practices with our technical directors, we found that the process of risk analysis was too subjective to an organization's individual practices, beliefs, and ideals. That is, the science of risks (and their probability and evaluation) is defined by a given threat exploiting vulnerabilities of an asset (or group of assets) causing harm to the organization. Because every organization quantifies the terminology of harm, asset, vulnerabilities, and threat differently - it was unbelievably difficult to represent actual risk analysis in our model. Regardless, we found the combination of risk assessments (based upon risk analysis within an organization) was of more value to practitioners in modeling interactions of risk and organizational risk. This was confirmed by the technical directors during a weekly meeting.

The most difficult aspect of our project was defining the research statement. Because we had diverted from risk analysis to combining risk assessments, we had to redefine the environment of the research statement. This new environment needed to represent an applied setting in any given organization. This meant that the model needed to have a modularized approach to combining risk assessments. With this in mind, we derived the following research (problem) statement:

- The purpose of the Risk Analysis (CSfC) project is to develop a meaningful method of combining risk assessments for individual security mechanisms into a risk assessment for the overall capability package (Layered Solution).*

  *We will define the terminology of Security Mechanisms, Layered Solution, and Capability Package later in this report.

2. Literature Review and Introduction to the Commercial Solutions for Classified Program

The Commercial Solutions for Classified (CSfC) Problem Space

The transmission of classified (or highly sensitive) data requires a high degree of assurance. This is known as the problem of data-in-transit [2]. Oftentimes, this assurance is in excess of the implemented equivalents in the private sector [8]. Because of this, the assurance that is provided for classified data generally requires several layers of formal accreditation by various regulatory agencies [1]. Traditionally, the only means to achieving this level of assurance has been through the adoption of Government-Off-The-Shelf (GOTS) solutions. However, these specialized hardware and software arrangements require intensive resources to develop and implement. GOTS solutions also typically see limited Return on Investment (ROI) due to the isolated nature of their deployment [9]. It is for these reasons (among others presented in this article) that the National Security Agency (NSA) Central Security Service (CSS) established the Commercial Solutions for Classified (CSfC) Program.

The Commercial Solutions for Classified (CSfC) Program aims to leverage the economy of scale and expedited deployment of commercial solutions in place of GOTS solutions. In doing this, commercially available mechanisms are assembled in *capability packages*. These packages represent a *Layered Solution* to assurance in commercial products. In doing this, new challenges for analyzing the risk associated with these solutions become present. Risks and vulnerabilities may be acceptably defined for a single mechanism, but the interaction of these risks, and more importantly, the overall risk landscape of the problem become much more complex and harder to define as these mechanisms are layered together. This is the foundation for many potential problems. While it is simple to assume that (intuitively) these Layered Solutions will have a higher degree of security than any of their parts (and that this separation will mitigate some of the risks associated with each layer), it is another problem altogether to quantify this gain and make a meaningful judgment as to the precise level of assurance offered by the solution as a whole.

The Challenges of Applying Risk Analysis to Layered Solutions

The most significant problem for conducting risk analysis on Layered Solutions is complexity (arising from interactions between layers). The interaction of risks between layers and their effect

on the package as a whole is hard to define, making meaningful assessments hard to conduct. When looking into the interconnected nature of these risks we uncovered a paper on cascade vulnerabilities [3]. This paper describes a method for modeling how failure at one element (or mechanism) in an interconnected system can transfer load to other elements (mechanisms) and how that changes their likelihood of failure. This model would look to hold a great deal of promise if the load on the system were to be defined as the amount of resources that a threat can place into breaking the solution. However, the correlation is not perfect. While there are many interconnected points of failure in Layered Solutions, they are not necessarily networked in a way that would work well with the requirements of the proposed model. Intuitively, there are a number of paths of failure starting at the attack surface on the outermost layer and ending in the leak of the targeted information. These paths are not necessarily connected to one another. Also, if the layers are not completely independent, there would exist a path to compromise shorter than the rest, even to the point of a single point of failure resulting in the leak of protected data, which would defeat the effect of having done such complex analysis.

Fault trees would be a logical way to model the paths to failure of a Layered Solution [5]. They would also be able to visually highlight the impact of a lack of independence. However, intuitively, as the number of layers increase, the complexity of the trees grows immensely and the may require a fair amount of maintenance as variables and relationships between layers are changed over time. Sommerstad et al. name this increase in complexity as one of their key motivations in wanting to develop a system to better handle more complicated systems [6].

Another issue with these Layered Solutions (but also with security mechanisms in general) is that some of the failures do not happen often enough to really generate a good statistical approximation. Pate-Cornell started to address specifically this issue in reference to natural disasters, but some of their approach may be of use to us here, if only conceptually [4].

Probabilistic Relational Modeling (PRM) in Risk Analysis

Our research for this project utilizes the probabilistic relational modeling approach proposed by Sommerstad et al. as a starting point. Their model for risk analysis is based around using a class-centric architecture model for a system built in Unified Modeling Language (UML) [5]. They use functions between these risks to change risk factors based on dependencies (driving the creation of unknown risk factors based on these same type of dependencies). In addition to being easy to represent in code, we believe that this class-centric approach offers a degree of internal modularity that we could leverage for the CSfC problem. For instance, we found this was an effective way to standardize the representation of individual mechanisms (making it possible to create such a representation whenever a new product is accepted into the CSfC product library). This also allows for fast “Drag-and-Drop” modeling and risk calculations for proposed layered capability packages.

Finally (and perhaps most importantly), the implementation of key risk factors coupled with the property of inheritance and dependency (created by this approach) would allow the calculations to be quickly modified and re-evaluated as new information regarding risk surfaced or the threat landscape changed.

3. Approach

As indicated in the detailed problem description of this report, it was necessary for us begin by redefining the original abstract into a more specified research question. Essentially, we needed to redefine the environment of the research statement to represent the environment of a Layered Solution. This new environment needed to represent an applied setting that is apparent in any given organization with an information system. Simply, our first milestone was to model the environment of the Layered Solution (incorporating the various components of the environment).

3.0 The Environment of a Layered Solution

3.0.1 The Asset and Security Measures

One of the primary purposes of our model is to assist in protecting the asset of a given information system. This asset is simply defined as a useful or valuable quality of an information system. It may represent controlled information, a specific system within the information system, or any other quality of the information system. It represented in the environment as Figure 3.1.

Figure 3.1

Using our model, an organization will immediately begin to associate security requirements for the asset. These requirements are typically in accordance with the organizational policy that requires enforcing. This can include perceived end user needs, regulations/laws, and best practices and standards (non-inclusive list). In our model and Figure 3.2, we represent these security requirements as being associated directly with security measures.

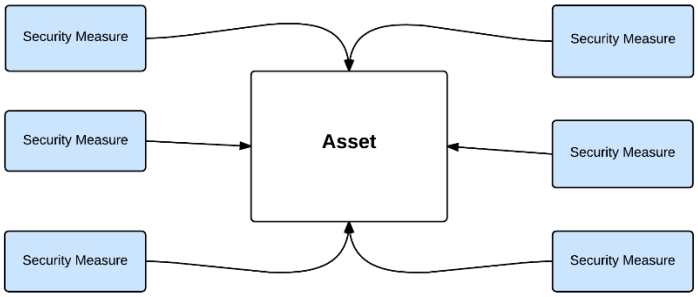

Figure 3.2

3.0.2 The Layered Solution

After defining the Asset and Security Measure(s), we could then begin modeling the Layered Solution. The Layered Solution is comprised of multiple security Mechanisms. These Mechanisms can represent both hardware and software means to addressing the Security Measure. In a Layered Solution, the Mechanisms are simply used in parallel. That is, each Mechanism is capable of fulfilling the given Security Measure; the layering of Mechanisms simply ensures that a new layer of protection is available in the event of a Mechanism failure. This is modeled in Figure 3.3. In theory, the difficulty of bypassing all of the layers of a solution will be more difficult than bypassing a single layer. We will discuss this concept later in this report.

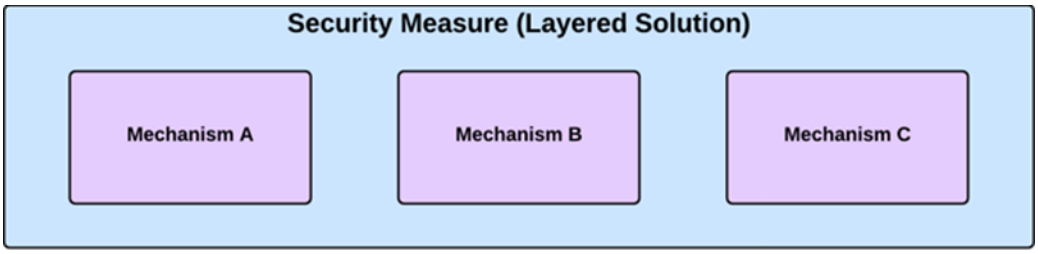

Figure 3.3

3.0.3 Independence and Critical Security Attributes

Each Mechanism in a Layered Solution is comprised of individual Critical Security Attributes. These Critical Security Attributes are qualities associated with the Mechanism. For example, a software-based Mechanism will have a Language, Administrator, Supplier, Development Tools, Compiler, and Developer associated with its implementation. We model this in Figure 3.4.

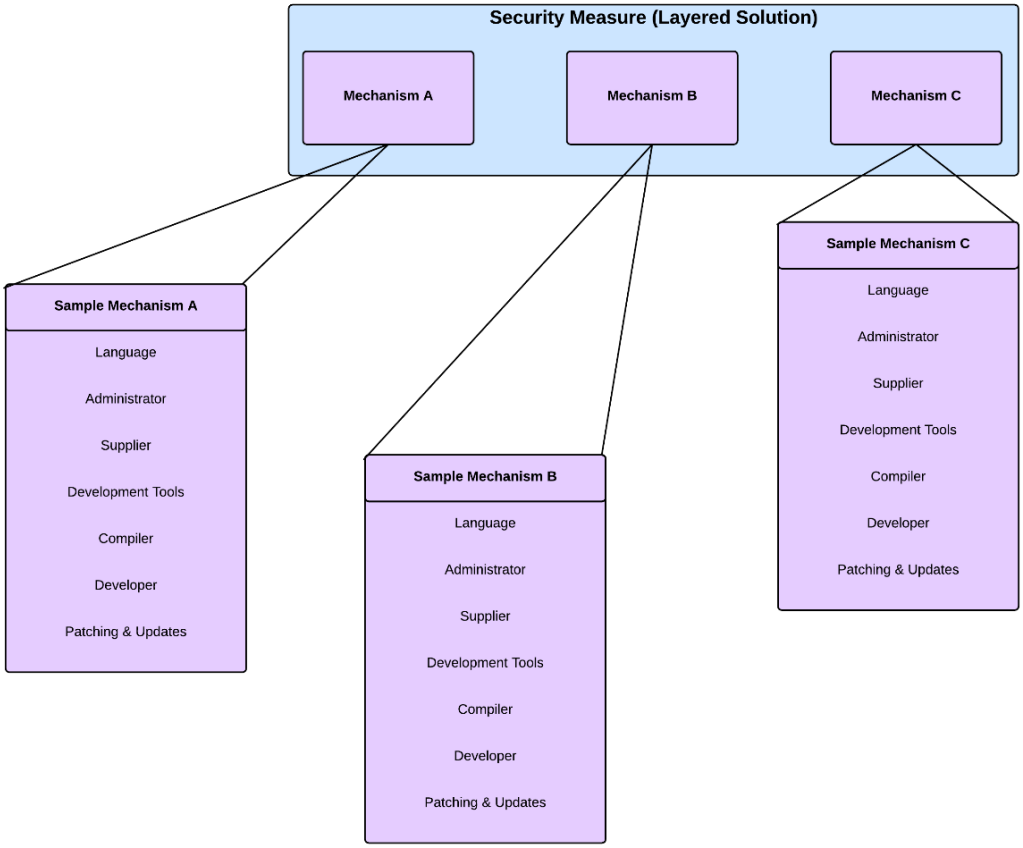

Figure 3.4

Specifically, it is at this level in the model that we begin to accommodate for the degree(s) of independence and interdepended between the Mechanisms in a Layered Solution. This poses a serious problem when we consider that each Critical Security Attribute is merely a potential point of failure in the overall Layered Solution. If a Critical Security Attribute is shared by more than one layer, then all layers with that trait in common could fail in the same way (negating the security advantage provided by multiple layers). We can imagine this phenomenon as being similar to a series of three locked doors that share the same key to a single room. If we wished to enter the room, we simple break all three doors. Of course, this can be a time-consuming process when we consider the complexity of modern Mechanisms in Security Measures. However, if we obtain the key to all three doors, entering the room becomes less difficult. The process of stealing the key only needs to be completed once (minimizing the overall work of an adversary). Considering this, it is feasible to believe that independence is a strong and intuitive way to view the situation. An example of this can be seen in a situation where all Mechanisms in the Layered Solution are software-based and use the same encryption algorithm. In this situation there is no guarantee that

all encryption algorithms are written by the same programmer (or have the same key). This poses as a weakness to their assumption(s) on Layered Solutions. We will address this item later in our report when we discuss risk resolution. The overall environment of a Layered Solution (for the purposes of this research) resembles figure 3.5.

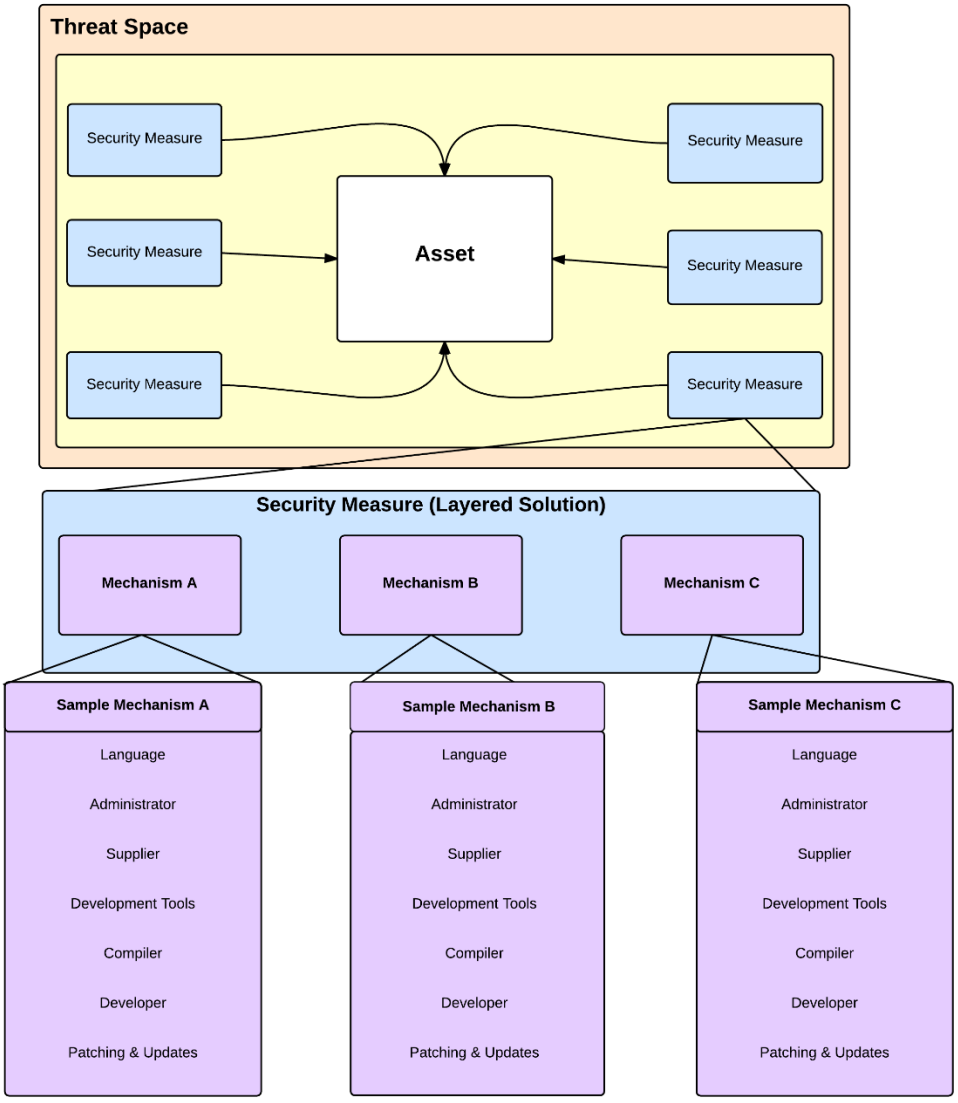

Figure 3.5

3.1 The Interactions of a Layered Solution

After defining the environment of the research statement (to represent a Layered Solution), we can now begin to define the internal mechanics of the model. As previously stated, one of our primary

goals was to ensure that our model be modular in design. This allows us to maintain scalability to different situations and easily modify individual pieces (either in future research or in operation).

To maintain this modularity, we utilize a function and class-based approach. This was influenced heavily by our discussion of Sommestad et al. [5]. In addition to maintaining the modularity and ease of modification, this method is amiable to representation in the form of programming (previously explored by Sommestad et al.). In order to accomplish this, we chose to dissect the model into three fundamental objects. They are: The Layered Solution Object, The Mechanism Object, and The Security Critical Attribute Object.

3.1.1 The Layered Solution Object

The Layered Solution Object is comprised of a number of fields that define its various properties. These can be seen in Figure 3.6. Residing at the highest level of the model, it is the most abstract and the simplest to define.

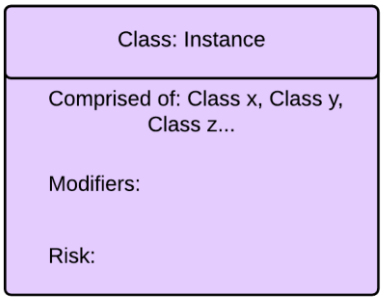

Figure 3.6

The first field is the "Class: Instance" field. This is where the specific Layered Solution is defined. The class identifies what sort of Mechanism the Layered Solution is using (e.g. Firewall, Intrusion Detection System Suite, End to end Encryption Solution, ECT.). The Mechanism may be a repeated (similar) device or series of different device. Specifically, the instance identifies the specific package. This could be a solution ID Number (primary key used to identify a specific configuration) or other value to represent the specific instance of a call solution.

The second field is "Comprised of: ". This contains a list of all the Mechanisms that make up the Layered Solution. That is, this field could contain any number of instantiations (although if there are one you don't necessarily have a Layered Solution). If there are too many instantiations, the feasibility of the solution may be in question. This is due to system efficiency vs. security. For the purposes of our research, we chose to use three instantiations. Merely, two mechanisms produces too few interactions. Four or more mechanisms begins to be too complex to model in a short research paper. However, the modeling we present can be used for any number of Mechanisms.

The third field is “modifiers: “. This field contains any special rules that may need to be applied to this specific implementation. These can be environmentally specific, or situational.

The final field is “Risk: “. This is what contains the risk score generated by the model. Currently, it is a result of the combination of risk scores at the mechanism level.

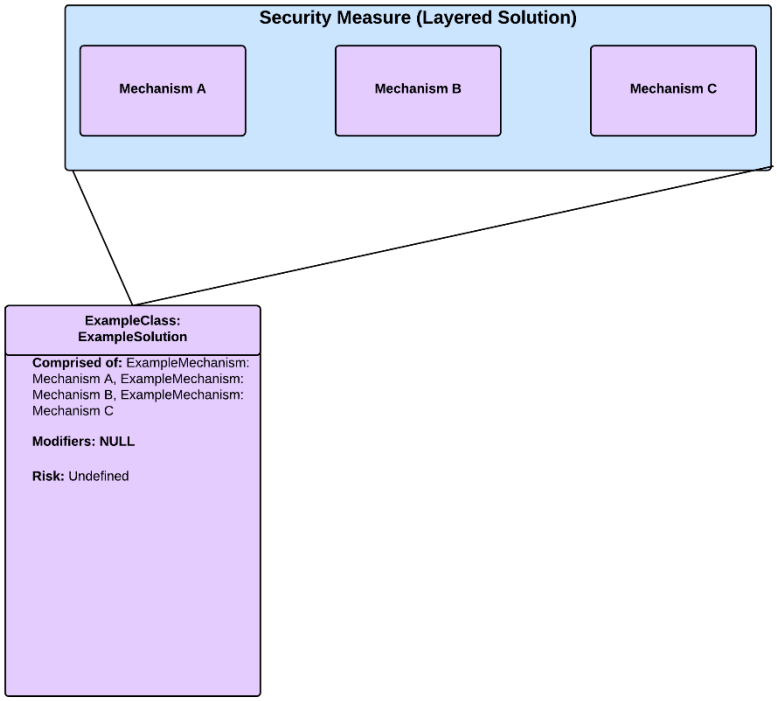

Figure 3.7: Overview of The Layered Solution Object

3.1.2 The Mechanism Object

The Mechanism object is one level below (in terms of abstraction) The Layered solution Object. Conceptually, it is constructed similarly as The Layered Solution object. This is represented in Figure 3.8.

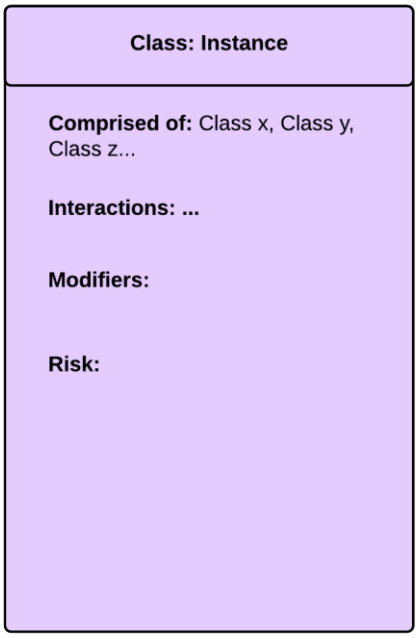

Figure 3.8

Much like The Layered Solution Object, the first field of The Mechanism Object is "Class: Instance". Again, class represents the type of Mechanism being represented. Some examples include: Firewall, Intrusion Detection System, Encryption Protocol, and ECT. While The Layered Solution Object represents this value as more of an identifier, The Mechanism Object can use the value to generate templates for how other fields are filled in (from the Security Critical Attribute Object). The template may require modification for specific instances, but it could at least provide some guidelines on how to look at each type of Mechanism to provide a level of uniformity. Instance is, again, the specific instance of the Mechanism. This value can represent an identifier, a serial number, or any other value that makes the Mechanism unique.

The "Comprised of: "field will contain more detail at this level. It contains a list of Security Critical Attributes that a Mechanism has. This field could be guided by a template from the class of the Mechanism, but is really only limited by the scope of the risk analysis.

The "Interactions: "field defines the type of interactions the Mechanism can have with other Mechanisms and the internet or external network. These can be used to generate rules for modifying or calculating risk scores or potential interactions between SCA's.

The "Modifiers: "field is, again, a placeholder for additional rules which may need to be applied in the future.

The “Risk: “field contains the risk score of the Mechanism. In most traditional risk assessments, this is the level at which risk is scoped for a product, and if need be, that value could be simply filled in here. However, in our model, this value is defined by interactions within the Security Critical Attribute (SCA) Object.

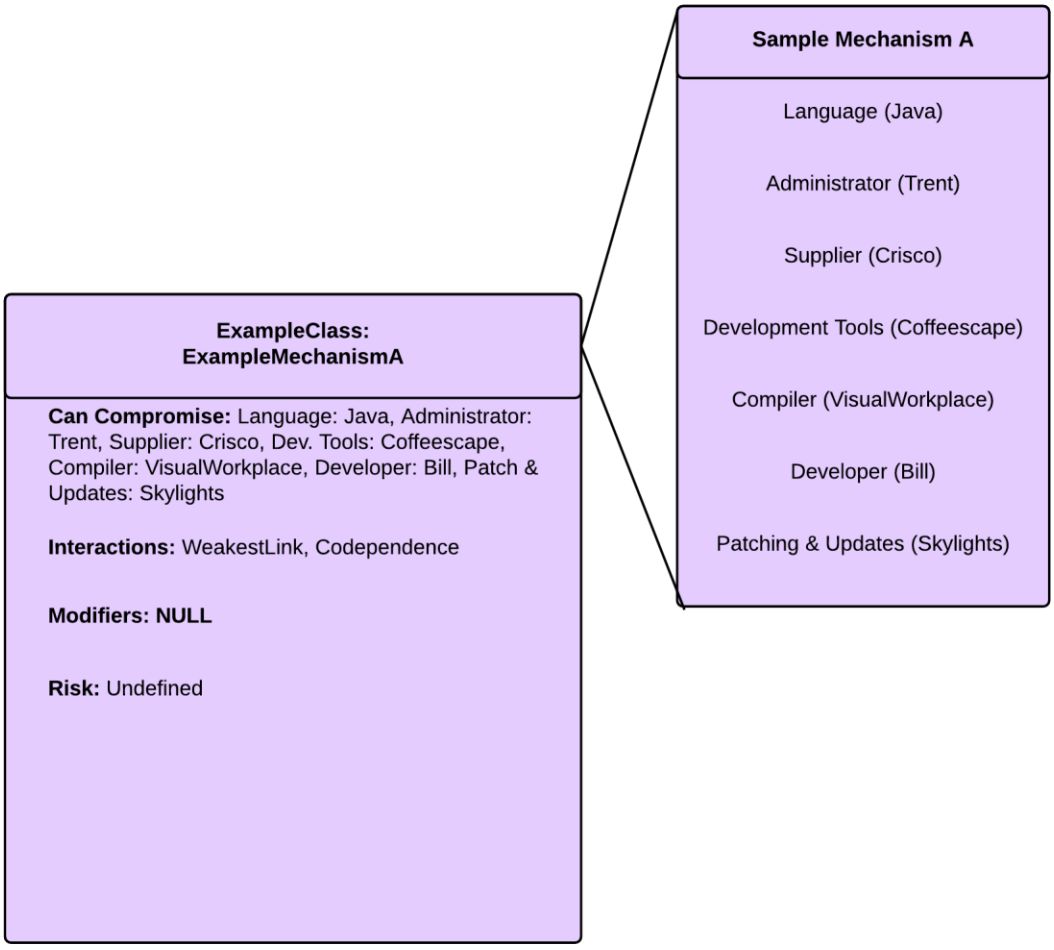

Figure 3.9: Overview of The Mechanism Object

3.1.3 The Security Critical Attribute Object

The Security Critical Attribute Object is the lowest level object in the model. It is also the most defined. The properties of the objects at this level truly define the calculations and combinations for everything up the chain. This is represented in figure 3.10.

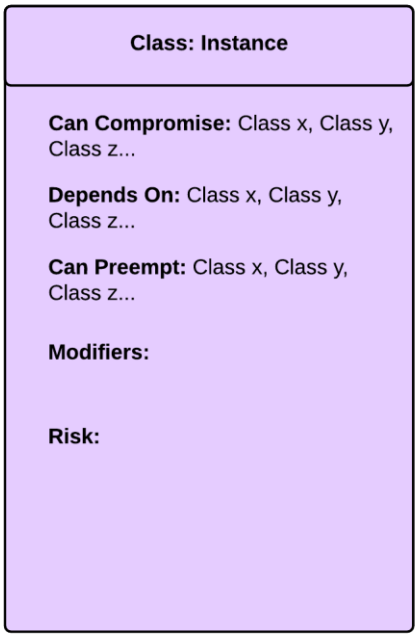

Figure 3.10

Again, the first field is the "Class: Instance" field. As before, class indicates what type of SCA is being represented. Example: Administrator, Compiler, Algorithm, Supplier, ECT. These can be used to generate Subject-matter Expert (SME) defined templates of how the SCA will behave. For instance, an administrator is going to behave and impact the system differently than say the compiler or operating system (due to the administrator's ability to manage associated aspects of the system). Instance, again, is a unique identifier to the SCA being represented. However, instance here takes on a new degree of importance; because it is at this level that points of independence and shared dependencies are evaluated. The model treats all instances of SCA's shared between mechanisms as points of shared dependence. Accuracy in this is important.

The "Can Compromise:" field is the first field representing interaction between different SCA's. It is comprised of a list of all SCA's that can be broken easily after this SCA has been compromised. Let's say an adversary can break the complier used to compile your mechanism - then the code is probably suspect. Or if the administrator has been compromised – the system settings may not be what they should be.

The "Depends On:" field is an alternative way of defining the interaction listed above from the other direction. If for some reason an SCA that this SCA depends on has not been included, or it did not have a compromise link, the interaction can be called out in this way. It compromises a list of SCA's that are subject to this interaction.

The “Can Preempt:” field functions same as the compromise field. It contains a list of SCAs that can be preempted by this layer. The difference between preempting and compromising is subtle. The principle is that - when compromising another SCA, it is being broken in place and may affect other mechanisms due to shared dependence. Preempting is a way of replacing another SCA with a compromised instance (in such a way that is independent from other mechanism instances). For instance, a malicious user performing a privilege escalation attack may gain administrative privileges - but that does not mean that the existing administrator has been compromised. It does mean, however, that the malicious user may be able to compromise and preempt SCA’s in the same way the original administrator could.

“Modifiers:” serves the same purpose at this level. It is simply a way to list out modifications or specific changes.

The “Risk:” field of the Security Critical Attribute, is the foundation of the whole model. These attributes are subjected to risk assessments by Subject-matter Expert (SME) in order to define this value.

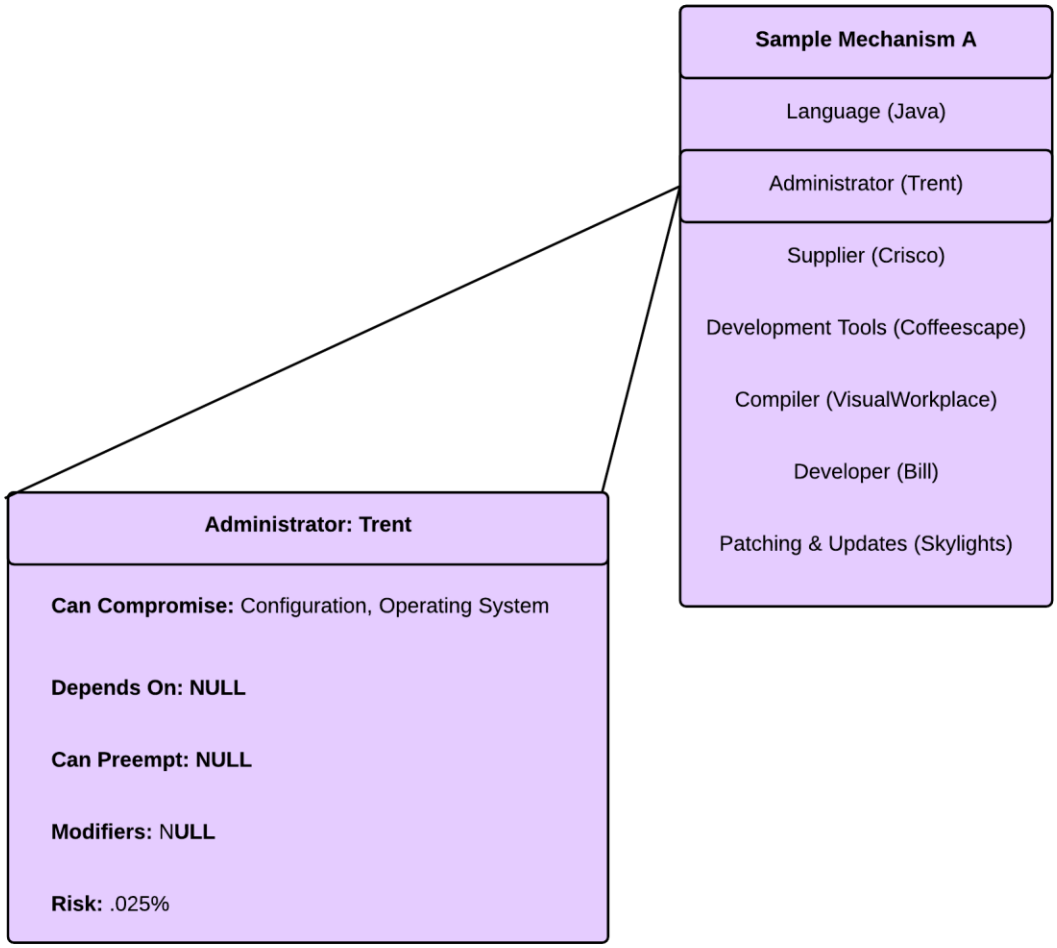

Figure 3.11: Overview of The Security Critical Attribute Object

3.1.4 Resolving Risk at The Mechanism Level

Continuing the discussion on The Mechanism Object, the risk score is derived from the interaction of SCAs and their instantiated risk scores. This is best described by showing the SCAs as nodes in a graph linked by their interactions

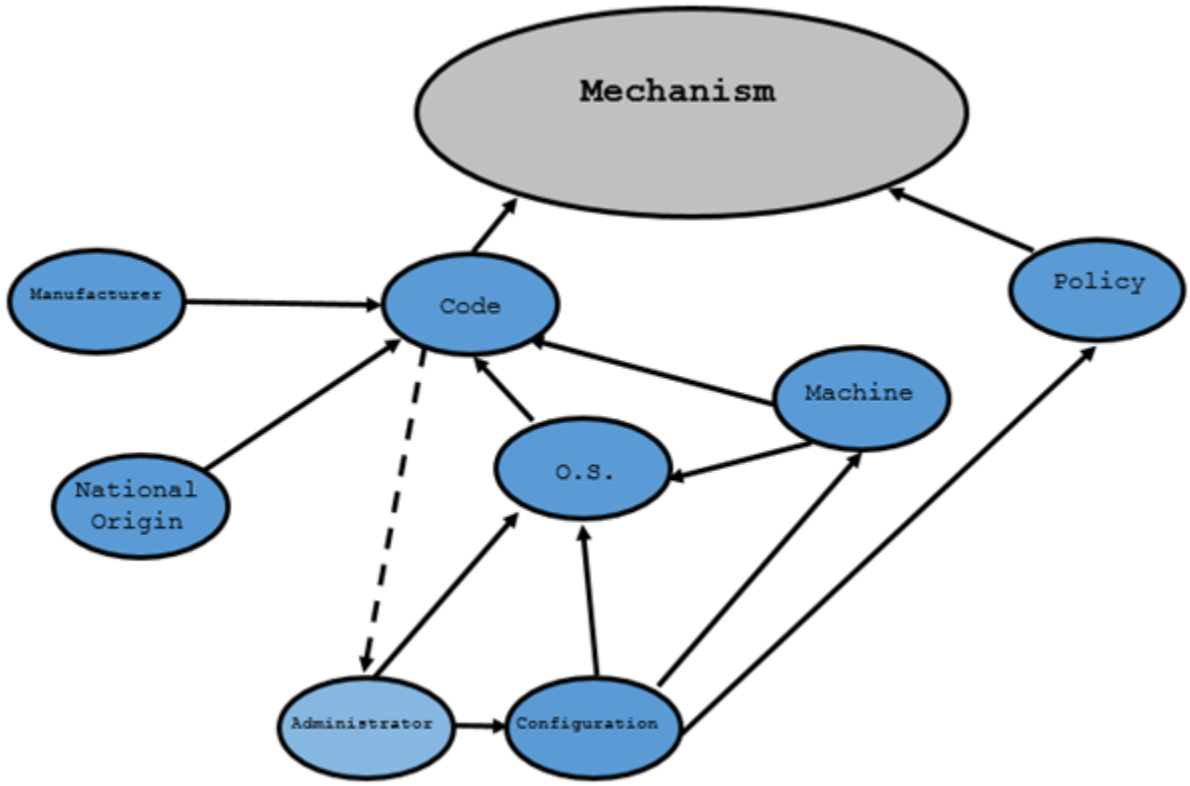

Figure 3.12

In figure 3.12, each of the blue nodes represents a Security Critical Attribute for a given software-based Mechanism. The arrows represent the interactions described in their object structure. The solid arrows denote a “compromise” relation and the dotted arrow denotes a “preempt” relation.

Currently, when determining the risk of the Mechanism, these relations are resolved starting from the node of highest rank. This knowledge (along with the current implementation of the compromise function that replaces the risk of the compromised node if it is less than the compromising node) allows us to demonstrate that the highest risk node is the administrator. This is represented in figure 3.13. In figure 3.14, the compromise relations resolve to set the risk field for the Mechanism.

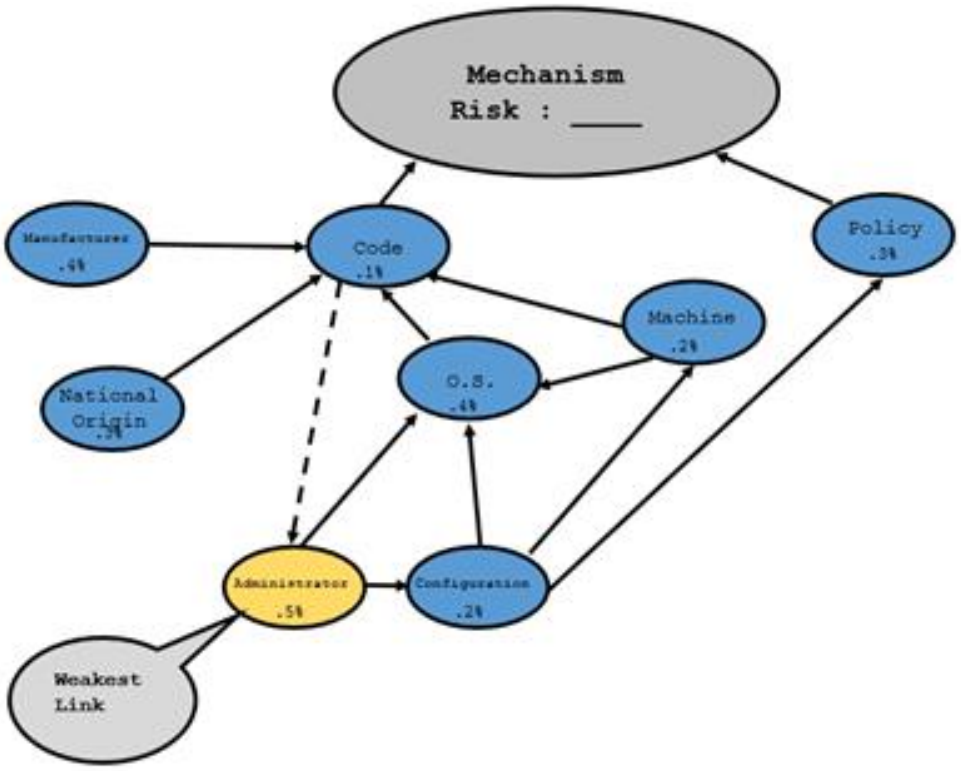

Figure 3.13

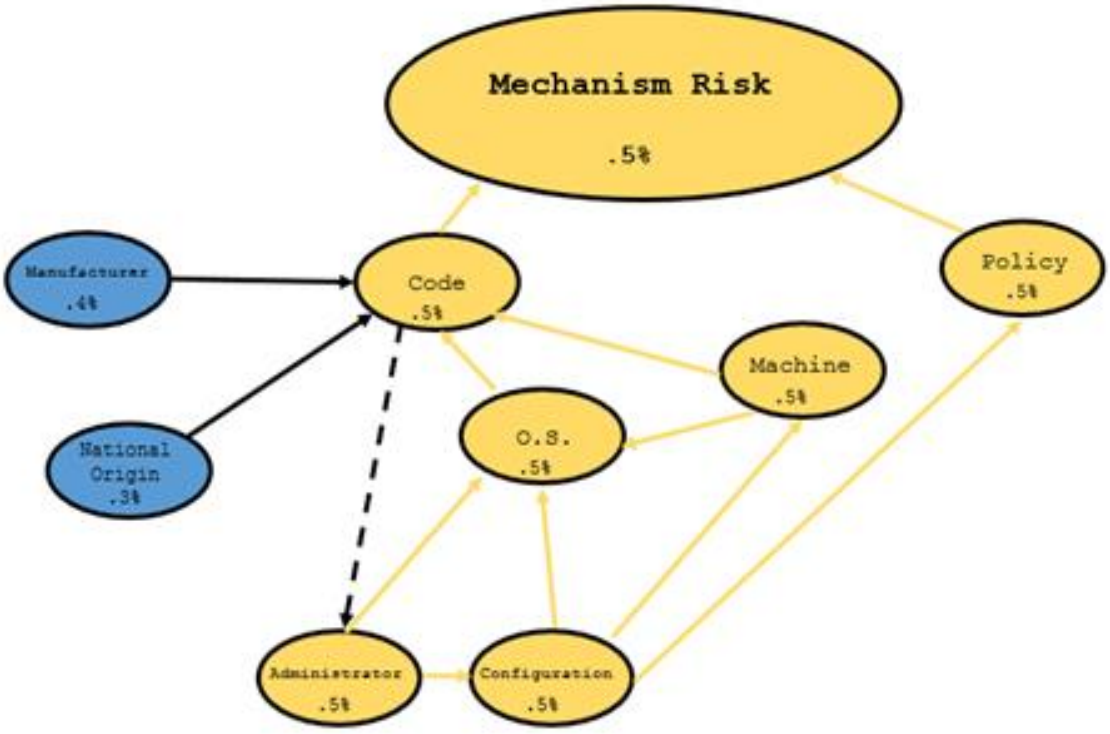

Figure 3.14

3.1.5 Resolving Risk at The Solution Level

When resolving risk at the solution level, we observe that all of the paths through the system assume a stronger attacker model for the independent path. This is the default path. The risk generated is the byproduct of the generated Mechanism risk (representing an independent breakdown). An example of this can be seen in the paths through shared SCA's (see figure 3.15). In this example, the risk is a one-time cost to traverse all layers where the shared SCA is present (breaking independently any additional layers). Specifically, this supports our Weakest Link principle.

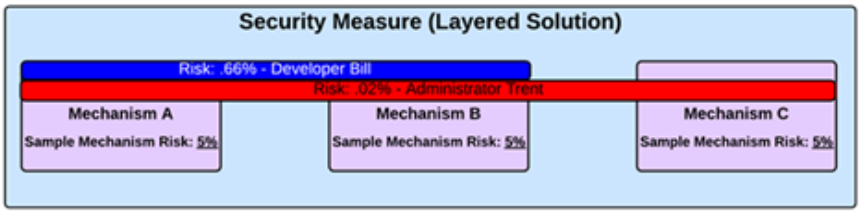

Figure 3.15

3.2 Data Management Plan

All data and artifacts are available via the Purdue University Research Repository (otherwise known as PURR). This includes copies of the problem abstract, soft bid(s), final bid(s), draft proposal, final proposal, progress report, draft final report, and final report. This repository also contains weekly dashboard updates, meetings presentations, and original copies of research materials used.

4. Results and Conclusion

Our research exemplifies a model for the measurement and sharing of residual risk in layered solutions. But more specifically, we wish to conclude this report by stating it is feasible to represent the combination of risk assessments in a simplistic manner. Our model is currently conceptual in design. However, given the correct resources, it could easily be tested in an operational environment. This model also provides a starting point for future research in layered solutions. We will highlight this later in our report.

5. Schedule

October

Week 1: Dashboard Update, Review of Original Abstract, Begin Literature Review on Vulnerabilities and Risk Analysis

Week 2: Dashboard Update, Continued Literature Review on Vulnerabilities, Scoping of Original Abstract

Week 3: Dashboard Update, Literature Review into Ontologies and Taxonomies

Week 4: Dashboard Update, Continued Literature Review into Ontologies and Taxonomies, Literature Review into Unified Modeling Language (UML, Research Question Design (Combining Risk Assessments)

November

Week 5: Dashboard Update, Established Layered Solution Environment (3.0 The Environment of a Layered Solution), Began Establishing Modeling for the Combining of Risk Assessments

Week 6: Progress Report / Presentation - INSuRE

Week 7: Dashboard Update, Obtained and Incorporated Feedback from CERIAS Faculty and Graduate Research Assistants on Modeling for the Combining of Risk Assessments into Draft Final Report

Week 8: (Thanksgiving Break) – Documenting and Formatting of Draft Final Report

December

Week 9/10: Final Report Submission, Final Poster Submission, Final Presentation - INSuRE

After Project/Semester Completion

March 2015: CERIAS Symposium Presentation

6. Budget

Our proposed budget of $3968.00 matches the actual budget necessary to conduct this project. This is represented in Table 6.1. The project was delivered, as proposed, on schedule.

Proposed Budget	Actual Budget
Senior Personnel Costs – $3968.00	Senior Personnel – $3968.00
* No other costs.	*No other costs.
Total Direct and Indirect Costs – $3968.00	**Total Direct and Indirect Costs – $3968.00**

Table 6.1

7. Final Discussion and Future Directions

Due to the limitations of risk analysis (previously stated in our detailed problem description), our model does require that organizations provide this information at The Security Critical Attribute Object. This is not a limitation to the model. Rather, the ideals of modularity are present in this aspect of the model. By allowing organizations to incorporate existing risk analysis practices, our model actually allows these same organizations the opportunity to evaluate risk assessments in layered solutions.

Below are some opportunities that exist for improving this model and layered solution assessment (this list is non-exhaustive).

Birthday Paradox

The Birthday Paradox (or Birthday Problem) concerns the probability that, in a set of *n* randomly chosen people, some pair of them will have the same birthday. This is based on the reality that, statistically, the probability of any two individuals sharing a *birthday* reaches 100% (or 1.0) when the population is represented by 367 individuals (since there are 366 possible birthdays - including February 29). With this in mind, it is feasible to believe that as the population grows to 367 individuals - the likelihood of any two individuals sharing a birthday also increases. For example, in a population of 23 individuals there is a likelihood of 50% (or .50) that a match will exist $(1 - (\frac{364}{365})^{23(23-1)/2})$. With a population of slightly less than 59 individuals, the likelihood (alarmingly) increases to approximately 99% (or .99).

Considering these figures, it is feasible to believe that this phenomena could also exist in cascading vulnerabilities amongst the Mechanisms presented in our model. Examples of cascading failure can be seen in Richard G. Little's work [10].

Evaluation of Risk

It is possible to represent the risk analysis assumptions as more than simple percentages. In theory, Bayesian scores can be utilized for the assessment of risk at The Security Critical Attribute Object proportion of our model. This, of course, would require modification of the scoring metrics to meet the requirements of an organization looking to adopt this approach. However, we do not feel as though it would affect the remainder our model. The organization would simply adopt a qualitative or categorical (vice quantitative or percentage) understanding of risk in layered solutions.

8. Bibliography

[1] Goodbye DIACAP, Hello DIARMF - InfoSec Institute. (2011, November 17). Retrieved October 9, 2014, from http://resources.infosecinstitute.com/goodbye-diacap-hello-diarmf/

[2] Shackleford, D. (2007, November 15). SANS Institute InfoSec Reading Room. Retrieved October 9, 2014, from http://www.sans.org/reading-room/whitepapers/analyst/regulations-standards-encryption-applies-34675

[3] Zio, Enrico, Sansavini, Giovanni, Component Criticality in Failure Cascade Processes of Network Systems, Risk Analysis Vol 31 (2011). Retrieved 9 October 2014 from http://onlinelibrary.wiley.com.ezproxy.lib.purdue.edu/doi/10.1111/j.1539-6924.2011.01584.x/abstract

[4] Ferdous et al., Fault and Event Tree Analyses for Process Systems Risk Analysis: Uncertainty Handling Formulations, Risk Analysis, Vol 31 (2011). Retrieved 9 October 2014 from http://onlinelibrary.wiley.com.ezproxy.lib.purdue.edu/doi/10.1111/j.1539-6924.2010.01475.x/abstract

[5] Sommestad et al., A Probabilistic Relational Model for Security Risk Analysis, Computer Security 29 (2010). Retrieved 9 October 2014 from http://www.sciencedirect.com.ezproxy.lib.purdue.edu/science/article/pii/S0167404810000209

[6] Pat´e-Cornell, Elizabeth, On “Black Swans” and “Perfect Storms”: Risk Analysis and Management When Statistics Are Not Enough, Risk Analysis, Vol 32 (2012). Retrieved 9 October 2014 from http://onlinelibrary.wiley.com.ezproxy.lib.purdue.edu/doi/10.1111/j.1539-6924.2011.01787.x/abstract

[7] Roeper, F., & Ziring, N. (2012). Building Robust Security Solutions Using Layering and Independence.*RSAConference2012*.

[8] Etzioni, A. (2013). Cybersecurity in the Private Sector. Issues in Science and Technology. Retrieved December 1, 2014, from http://issues.org/28-1/etzioni-2/

[9] Commercial Solutions for Classified Program. (2014, March 1). Retrieved December 9, 2014, from https://www.nsa.gov/ia/programs/csfc_program/

[10] Little, R. (2002). Controlling Cascading Failure: Understanding the Vulnerabilities of Interconnected Infrastructures. *Journal of Urban Technology, 9*(1), 109-123.

9. Biographical Sketches of the Team Members

Christopher Martinez

Christopher Martinez was born in Silverdale, Washington, USA in 1989. He received his Bachelor of Science from The University of Washington in 2013.

In 2012, Christopher joined The Center for Education and Research in Information Assurance and Security (CERIAS) at Purdue University. He is studying for his Master of Science in Interdisciplinary Information Security under the Scholarship for Service (SFS) CyberCorps program. He is an active member of the National Eagle Scout Association (NESA), USA Freedom Corps, Upsilon Pi Epsilon, National Society for Hispanic MBAs (NSHMBA), and the Association of Computing Machinery (ACM).

Robert Haverkos

Robert Haverkos received a Bachelor of Science from Embry-Riddle Aeronautical University in 2013. In 2014, Robert joined The Center for Education and Research in Information Assurance and Security (CERIAS) at Purdue University. He is studying for his Master of Science in Interdisciplinary Information Security under the Scholarship for Service (SFS) CyberCorps program. Robert's research interests include information security and information analysis.