Annex A – COMPOSED SYSTEMS

This section provides an overview of the use of the composed system model as a potential mechanism by which the Common Criteria approach can be leveraged to be of use in performing TRA. This model is used to combine many CC evaluated components into a composed system, which more accurately reflects the scope of a TRA engagement. This approach is generally referred to in the CC as a system-level evaluation. Examples of how assurance levels can be derived for a composed system are provided and the challenges of using this approach are described.

A.1 COMPOSED SYSTEMS

A logical view on these complementary, yet dissimilar, approaches to IT security as it pertains to risk management is to determine how the strengths of each can be combined into a more complete solution and, simultaneously, reduce the effort to conduct risk analysis by allowing one effort to build on the other. This approach would entail combining the component-based, requirements-oriented quantitative view of the Common Criteria with the holistic approach espoused by the TRA methodologies. One method to perform such an amalgamation of approaches is to view a system as a sum of components, what is referred to in IT circles as a composed system.

The concept of composed systems dates back to early days of IT architecture development and has been employed in various guises. Most notably in software development circles, the ***composite design pattern***[1] is frequently used to allow clients to treat objects or compositions of objects uniformly in a hierarchical representation. A composed system may be designed with the purpose of driving specific properties for a collection of constituent components, or the composite properties of a collection of components may be viewed in a systemic fashion. From this specific context, it is possible to apply a TRA methodology to CC evaluated component, by examining the composition of such components as an equivalent and uniform entity.

A composed system will view all constituent components as a single system and strive to optimize the system as a whole. Within the context of integrating the CC and TRA approaches, the constituent components are CC evaluated products, and optimization is the reduction of risk to an acceptable level across the entire composed system. There are immediate concerns with such an approach:

1) **Lack of Common Authority**: From a system management perspective, simply choosing to view a collection of components as a system does not mean that there will be agreement among the respective authorities for these components as to the methods by which risk will be reduced across the system as a whole. This difference of opinion among key stakeholders is primarily a project management issue and is often encountered in the TRA process. Nevertheless, there is reason to expect that this difficulty will be greater under this model, given that various stakeholders will have more investment (time, prestige) in maintaining the assurance level for their own component;

2) **Local versus Global Optimization**: A concern in any composed system approach is that the optimization for the system as a whole will be achieved at the expense of the optimization of some individual components. This concern may be less significant since a fundamental assumption of the composition and risk analysis approach should be that no changes to the environment can jeopardize the assurance level of any given component. Within the bounds of this constraint, however, the changes to the system can be recommended so as to minimize the

[1] Eric Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns, Addison-Wesley Professional Computing Seriesm, 1995, pg. 163.

risk level without compromising the assurance level of the continuant parts. It is, in fact, a generally accepted principle that a system cannot be optimized by optimizing its component parts, but only through the optimization of the solution itself. This principle provides further evidence that system-level management is needed to utilize the composed system model properly; and

3) **Redundancy**: Similar to the previous concern, the fact that individual elements of the composed system cannot be altered in such a way that the assurance level of these components is invalidated, it is likely that the composed system will contain security safeguard redundancies. The mere presence of redundant safeguards in a risk analysis environment is often raised as vulnerability. One can speculate that a redundant safeguard brings vagueness to the manner in which information assets are protected. Also, there is a concern that security policy decisions may be applied inconsistently if there are multiple paths to access system/information assets. Redundant safeguards also have a limiting effect on supporting system security activities such as auditing and maintenance.

However, the most fundamental concern is how CC evaluated components can be aggregated into a composed system in such a manner that it is possible to make a statement about the security level of the system itself. This discussion is limited exclusively to the notion of how an "assurance level" (or equivalent) can be assigned to the composed system. It is recognized that there will have to be architectural standards by which these components can be combined to form the composed system.

In the paper on ***Composable Trusted Systems***[2], Lee [et al.] suggests two methods for generating an assurance level for a composed system. Each method is described below.

A.2 VERTICAL ASSURANCE

This approach for composability generates a cumulative assurance value from the lowest level component to the highest (systemic) level. Each level of the composition would be treated as separate and independent in terms of it ability to meet the required level of assurance. At each composite level, it is possible to determine if the policy requirements are met by the implementation details at the next level. If policy requirements are met at each level within the composed system, a ***chain of belief*** is formed which provides assurance that the composed system is in compliance with policy requirements at a systemic level.

In presenting this approach, Lee [et al.] provided a mechanism for forming a metric to calculate assurance and expressed, at each level, a probability of loss of assurance for a given policy statement. These level-based policies were then aggregated to express a Cumulative Probability for Loss of Assurance at the systemic level. It was recognized that there were several limitations to this approach including:

1) The difficultly in expressing policy adherence in terms of probabilities;
2) The loss of information relating to the fact that adherence is expressed as a binary success/failure, rather than providing insight into the degree of failure; and
3) Once expressed as a cumulative loss of assurance, any policy violation cannot be traced back to a specific policy level objective.

A.3 STRUCTURAL ASSURANCE

This method of composing assurance for a composed system takes a simpler view. Essentially, all properties of a composed system are assumed to be a union of the properties of the constituent parts. From this description, there are two scenarios which can be described in terms of how components will interact:

[2] E.S. Lee, B.W. Thompson, P.I.P. Boulton, R.E. Soper, Composable Trusted Systems, Technical Report CSRI-272, May 31, 1992.

1) Components are isolated so there is no interaction defined in the specification; and
2) Interactions are planned, designed and implemented as part of the specification.

The first scenario implies that deviations from the expected behaviour of a given component will not affect the behaviour or assurance level of other component functions or properties within the composed system. In this way, each component can be treated as having independent properties and the properties can be accumulated and analyzed independently. This view is contrary to several fields of systemic research including complexity theory and patterns of emergent behaviour.

When analyzing a composed system using the structured assurance method it is necessary to take into account the degree to which a failed component or function will interfere with its neighbouring components. An analogy can be made with a gearing mechanism where a slightly misadjusted gear may have a much more significant impact on the gears to which it is linked, potentially resulting in a complete systemic failure. In his paper, Lee [et al.] provides a more specific example of memory corruption in software modules resulting in a complete systemic fault. This concern can be ameliorated through the use of barriers, included as part of the specification, to effectively isolate components and reduce interference.

The second scenario, in which interactions are well defined in the specification using controls to keep the interaction within the bounds set by the policy specification, is not exempt from the structured analysis of intra-component interaction. However, the overall analysis is more complex in that the designed interactions must be scrutinized to ensure that they are in compliance with the defined policy objectives (e.g. channel integrity). Techniques such as ***sensitivity analysis*** are recommended for this form of evaluation in which the input from a component must be treated as suspect. Automation tools exist which can assist with this analysis, both with software and hardware components.

In essence this technique requires an impact analysis between each component in the composed system to determine the sensitivity of the system to catastrophic failure due to inherent limitations in the composed system itself. The level of effort to perform such an analysis grows significantly as more components are introduced to the system. It should be noted that when two composed systems are themselves to be merged into a higher order composite system, the structured approach would require that the composed system be first decomposed and an analysis performed on each of the lowest level components. This requirement is due to the fact that there may not be inter-system interference in either of the composed systems, but interference, at a component level, may exist when the composed systems are combined.

A.4 A NOTE ON COMPLEXITY THEORY

"A system that is not explicitly described by the behaviour of the components of the system, and is therefore unexpected to a designer or observer can be deemed to have emergent properties."

The goal of the CC is to ensure that a system has been designed, implemented and documented according to policy and specifications. However, an alternate view on the process is that an evaluator is trying to disprove the negative hypothesis; that the system has not been implemented according to the specifications. With this assumption, it can be extrapolated that the system is going to display unexpected behaviour. From this statement, one can add that such a system will have emergent properties.

In the context of a composed system, the combining of two such systems (or components) will create not only a more complicated system in the general sense of the word, but also a complex system. Behaviours and patterns emerge in complex systems as a result of the patterns of relationship between the components. There are generally accepted properties of complex systems that define the impact emergent behaviour will have in a composed system including the following:

1) There are rarely simple cause and effect relationships between elements. A small deviation in a single component may have a large impact at the system level through the interaction with neighbouring components;
2) Emergent behaviour often encompasses feedback (or dampening) potentially amplifying the initial deviation into a system level fault; and
3) Decomposition of the emergent behaviour is extremely difficult based on observation of the behaviour itself. Debugging a system that is displaying emergent properties is extremely difficult since the actions at the system level cannot be explained and the components on their own cannot display the behaviour.