Managed Mission Assurance

Concept, Methodology and Runtime Support

Partha Pal, Kurt Rohloff, Michael Atighetchi, Rick Schantz
BBN Technologies
Cambridge, MA 02138
{ppal, krohloff,matighet, rschantz }@bbn.com

***Abstract*—We interpret "mission assurance" to mean the guarantee that Mission Essential Functionality (MEF) provided by an information system is continued despite partial failures and other accidental or maliciously induced changes in the system or its operating environment. MEF can be threatened when the Quality of Service (QoS) delivered by the information system drops below useful levels, when the security and Information Assurance (IA) of the system is compromised, or both. Key enablers of mission assurance under this interpretation are the ability a) to detect when MEF is in jeopardy and b) to remain within "regions of usefulness" in terms of QoS and IA. We describe an approach to managing mission assurance, and the underlying concepts, methodology and runtime support needed to realize the above mentioned enabling capabilities.**

Keywords-mission assurance, cyber-security; quality of service; runtime assessment; runtime adaptation

I. Introduction

Modern organizations, military and civilian alike, are increasingly dependent on distributed network-centric information systems. The information systems provide services and functionality that are essential for the success of the organization's "mission". We interpret "mission assurance" to mean the guarantee that Mission Essential Functionality (MEF) is continued despite partial failures or changes in the system and its operating environment.

MEF can be threatened by diminished Quality of Service (QoS) or comprises in the security or Information Assurance (IA) of the information system, either of which can be caused by accidents or malicious cyber attacks. The most vexing threats, from a mission operation point of view, are cyber-attacks. Cyber-attacks can be quick or long lived, can come from insiders or outsiders, are difficult to model because they follow few physical laws, if any, and can result in service unavailability, resource exhaustion, information leakage and corruption. Given the ease with which adversaries can mount highly asymmetric cyber-attacks against more powerful opponents, cyber-attacks are a critical threat to national security. Furthermore, cyber attacks targeting mission support systems are expected to rise sharply compared to the usual level of background attacks on generic networked systems.

Because mission-critical information systems will need to operate in contested environments (both physical and cyber), "mission assurance" – the guarantee that the MEF is continued despite the compromises and outages that are inevitable in a contested environment — must be treated as an engineering and operational goal. This motivates us to be able to identify when mission assurance is threatened, and manage the threat. However, current technology does not provide adequate support for realizing these goals. For instance—how does one know when a mission is in jeopardy before the information system fails to perform a mission critical task? By the time a failure manifests, it is often too late for any reaction to maintain mission continuity. Furthermore, although some forms of QoS degradation is visible (e.g., longer response time) to the users, it is often left to the user to decide when to take corrective actions. This often leads to slow and incorrect reactions. It is even worse for security compromises, because effects of such compromises may not be visible to the mission stakeholders who use the information system. Modern information systems record a lot of "security incidents", but their impact on the mission is often unknown. At the same time, users often need to install new software or change some configurations to achieve their mission goal. Under the current state of practice, users are forced into a difficult tradeoff: either they are prevented from making these changes at the cost of mission efficiency, possibly failure, in order to remain secure; or they are forced to accept unknown mission risks as a consequence of their actions. A better solution is clearly needed.

This paper is an early report of our efforts to develop a framework where mission-oriented QoS and IA requirements can be captured, continuously assessed and managed at run time by effectively trading off service delivery for information assurance or vice versa. The goal of the continuous assessment and QoS-IA tradeoff is to always remain within "regions of usefulness" based on stakeholder specified QoS and IA requirements. The main contribution of this paper is the formalization of the notion of "managed mission assurance". This paper also introduces a requirements engineering based framework to capture mission-oriented QoS and IA requirements and tradeoff policies in support of managed mission assurance.

The remainder of this paper is organized as follows. In the next section a brief survey of related work is provided. In Section III we explain the concept of "managed mission assurance". Section IV focuses on QoS-IA tradeoff for managed mission assurance. The proof of concept we are developing to demonstrate the envisioned "managed mission assurance" concept is briefly described in Section V. Section VI concludes the paper with a discussion.

Approved for Public Release; Distribution Unlimited: 88ABW-2010-2136, dated 19 April 2010

II. RELATED WORK

For many missions (such as space operations or air traffic control), the responsibility of mission assurance often falls on a special-purpose and custom made “information system”. However, the focus of this work is on missions where an information system is at the heart of the mission operation. In most cases, such information systems include enterprise system management (ESM) capabilities, often centrally monitored from a command center or a Network Operation Center (NOC). ESMs and NOCs represent a specialized class of software that is relevant for the present work. A number of technology platforms are used by system administrators today to monitor and prioritize reported events, and to establish regulatory compliance. Centralized graphic representation of real-time status information and generation of reports that are understandable by management are key functionalities of those products. For example, the PROMETHEUS [1] system of the U.S. Navy’s Cyber Defense Operations Command (NCDOC) automates the monitoring of a large network. It accesses various log files including system log, Web log, e-mail log, firewall log and router logs from the entire network, and prepares and stores the aggregate data for analysis and reporting. It uses Novell’s Sentinel platform [2] to present and prioritize security events in a centralized dashboard. Akamai’s information security management system [3] is another example. Each server in Akamai’s EdgePlatform is watched by a set of “watchdog” components that provide system-level monitoring for security events and anomalies as usage, performance, process counts. These components report through a distributed database system that provides alerting and reporting to Akamai’s Network Operations Command Center (NOCC). Additional automated systems within the NOCC analyze and report on system-wide conditions and trends, and prepare information for inspection by NOCC staff. Apart from custom-made solutions like the ones described above, commercial products like IBM’s Tivoli Security Information and Event Manager [4] and Symantec’s Enterprise Security Manager [5] offer general purpose security event management. These products offer a security operations dashboard that facilitates attack recognition, prioritization and incident management, and enables construction of compliance arguments to various industry security standards. The data collection, parsing and aggregation capabilities offered by the ESMs can be leveraged in our approach. In fact, our proof of concept prototype integrates with representative ESMs to take advantage of their data collection and reporting capability.

A second thread of related work involves the analysis and reasoning of collected information in light of plans and workflows that are relevant for the mission. The Strengthen, Prepare, Detect, React to Mitigate the Insider Threat, DARPA/AFRL (SPDR) project [6] under DARPA's Self Regenerative Systems (SRS) program is an example. The SPDR work focused on recognizing attack plans from data collected from specialized network level mechanisms called Detect Response Embedded Devices (DREDs). In contrast, our goal is to detect deviations from the mission plan and workflow based on data collected by existing sensors.

Specifications and guidelines such as NIST SP 800-30 [7] and DoD mission-assurance policies (e.g., those being explored by the Mission Assurance and Network Operations PEO [8]) represent the third category of work related to our approach. These policies and guidelines provide a blueprint for human experts to assess and mitigate mission risks. Our approach is not inconsistent with any of these, but our focus is more on online risk management that trades off QoS with IA and vice versa.

III. MANAGED MISSION ASSURANCE

As mentioned earlier, continuity of MEF is threatened when security of the system is undermined or when the system fails to deliver the desired levels of service. Managing mission assurance therefore involves timely detection of the system’s departure from acceptable levels of service delivery and information assurance, and nudging it back to acceptable levels either automatically or through human intervention. Although QoS management has been demonstrated earlier [9], and considerable progress has been made in our ability to develop systems so that autonomic and human-assisted defensive responses can be mounted reactively and proactively [10], measuring information assurance—i.e., quantitative evaluation of security has been problematic. Most security assessment approaches are qualitative, and focus on processes than the system in operation. Therefore, in our approach to managed mission assurance, while we strive to reuse and leverage existing QoS and survivability mechanisms, we had to take a different tack on assessing information assurance (IA).

Our approach to managed mission assurance is driven by the understanding that a mission has multiple stakeholders, each may have different required levels of IA and QoS, and each stakeholder’s requirements may vary during the mission.

Informed by our experience in managed QoS and system survivability, we define “assessment” to mean estimating the level of QoS or IA *delivered* by the system at any given point against a *required* level. This aligns our notion of IA assessment with our notion of QoS assessment, and also eliminates the need for an absolute quantification of IA.

Of course, neither IA nor QoS is monolithic. Usually they are expressed in terms of a number of attributes. Some attributes such as confidentiality and integrity are associated with IA, while attributes like timeliness and fidelity are associated with QoS. There are some attributes such as availability that are used to describe both QoS and IA. In a mission, a stakeholder may not have requirements for all attributes, and not all attributes may be important for him all the time. In our approach, stakeholder specified requirements collectively define what we treat as MEF.

But expressing QoS or IA requirements and assessing them require additional information about the attributes. Attributes like “confidentiality” by itself does not mean much, unless it is bound to a spatial context such as a service or a network link. Stakeholder role and responsibility may change during the mission. Requirements and assessment must be cognizant of the time varying aspect of the requirements. Therefore, we claim that management of mission assurance takes place in a multi-dimensional space where various stakeholders can specify their requirements over time. The requirements are expressed as ordered levels (e.g., level1< level2<…levelk) for a specific attribute (e.g., Confidentiality) for a specific spatial scope (e.g., a link between client and service). We also claim that the assessment should be continuous, i.e., performed in an ongoing manner, while the mission is executing.

To facilitate requirements capture in this multi-dimensional space and subsequent assessment against the captured requirements, we propose a new software engineering role, namely the assurance engineers. We also make the stakeholders shoulder some of the responsibility for meaningful assessment by having them specify what level of QoS and IA they must have in order to successfully execute their mission roles. Since stakeholders are not QoS or security experts, we only ask for a high level specification—the key information we seek from the stakeholders are the ordered levels $L_{A, S}$ for different periods in the mission where A represents a specific attribute such as Confidentiality, and S represents the spatial scope i.e., the functions and assets they are interested in. These requirements are elicited from the stakeholders by the assurance engineers, who also instrument the system for collecting measurements and run time assessment. This often involves integrating the runtime assessment support with existing enterprise system management facilities.

Runtime IA assessment is essentially a mapping from measurements obtained from the system at a given time within a mission to the space of $L_{A,S}$ for various stakeholders. To complement the already developed QoS assessment capability, we proposed an organization of measurements relevant for IA assessment into a number of metric classes and a companion assessment methodology. This is described in [11]. In this paper we just note that the two classes of measurements, DEF STAT and RES STAT, representing status of the defense mechanisms and status of system resources respectively, that are easily available in most modern systems provide a generic way to estimate the level of IA delivered by the system at any given point within the mission.

Combining the proposed IA assessment capability with QoS management and survivability architectures enables effective management of mission assurance. To see this, note that the survivability architectures empower the system to monitor the system for security incidents and to mount defensive responses as needed. Similarly, QoS management offers a way of monitoring the QoS delivered and manipulating system resources and application behavior to maintain the delivered QoS at acceptable levels. The IA assessment capability then monitors the delivered levels of IA and alerts the stakeholders, including the system administrators responsible for managing the system's defenses, when their requirements are not met. In an ideal case, autonomic and human administered QoS and survivability management function would lead to satisfaction of QoS and IA requirements of all stakeholders all the time. But this ideal condition is hardly achieved because stakeholders' requirements, even the QoS and IA requirements of a single stakeholder often compete with each other. Moreover, service delivery and defense mechanisms make rely on the same system resources making security and service delivery a zero-sum game. Therefore, meaningful tradeoff between IA and QoS when the system cannot deliver both at their respective required levels becomes an important aspect of effective mission assurance management. Section IV provides more details.

IV. QoS-IA Tradeoff

In many operational contexts, information systems cannot deliver both high levels of service and information assurance. Deployed systems operate under real-world constraints with limited resources to simultaneously service requests and provide security. Consequently, mission stakeholders often need to make tradeoffs between the Quality of Service (QoS) and the Information Assurance (IA) delivered by the system to maintain MEF. We argue that the stakeholders need to be explicit not only about their requirements, but also about their tradeoff preferences in order to maintain mission assurance.

The IA and QoS requirements specified as $L_{A, S}$ may be conflicting with each other- some of these conflicts may be visible at the high level requirements itself, e.g., multiple stakeholders may be interested in the same spatial scope (S) of the system and may have conflicting requirement about the same QoS or IA attribute (A). Some conflicts may not be identified without further analysis, e.g., a stakeholder needs both confidentiality and fast completion time for a given spatial scope S, but the time needed to encrypt and decrypt will make the delay unacceptable. Some others may not be known until run time, e.g., there is not enough CPU resource to allocate the tasks from multiple stakeholders and the defense mechanisms. A framework supporting analysis of specified requirements as well as runtime reasoning about tradeoffs is therefore needed.

The taxonomy of QoS and IA metrics [11] we use in our tradeoff framework allows some metrics to be associated with multiple attributes. Figure 1 shows a partial view of the relations between our metric classes and QoS and IA attributes. Underlying the attributes and metrics are causal structures that relate changes in system configurations to observable metrics that affect the attributes and, ultimately, user perceptions of QoS and IA. In a sense these causal structures dictate what can be observed, measured and controlled, and how changes in the controllable aspects of the system impact the attributes. The causal structures form the basis for managing the tradeoff between QoS and IA. By manipulating the causal structures we can adjust the measurements and hence the assessments. The motivation behind this formulation of the attributes, metrics and causal structures stems from the deficiencies in past attempts at measuring security or assurance of the system as a whole. These approaches, focused on organizational maturity, development methodology, and various forms of penetration and stress testing were largely subjective and are typically done in isolation from ongoing missions. The overall score for the system obtained from such an evaluation process is somewhat meaningless for mission stakeholders and does not help in making runtime decisions when IA and QoS requirements cannot both be met. Subjective assessment is an inescapable reality of assessing something like assurance or security that is inherently related to users' perception. But our contribution is that we decompose the overall IA assessment problem into evaluating observable and directly measurable metrics against specified requirements similar to what is commonly done with QoS. One can argue that our decomposition is subjective, but the assessment is not. We claim that if the security engineer consistently applies the decomposition, our approach leads to more effective assessment and meaningful tradeoffs between information assurance and the quality of service.

Causal structures play an important role in our framework. As an example of how we anticipate manipulating the causal structures in order to facilitate QoS-IA tradeoff, consider the policy that governs a firewall guarding a network enclave that participates in a mission. The firewall policy, customized for that firewall for that mission, is an example of a causal structure. The firewall policy connects throughput (a metric that can be measured) and availability (an attribute that can be used to described QoS or IA) because changes in the firewall policy will directly impact the throughput of service requests to the enclave, and if some requests are now blocked, availability from the requester's point of view will also diminish. Stricter checking boosts IA attributes like confidentiality and integrity, but degrades QoS attributes like availability.

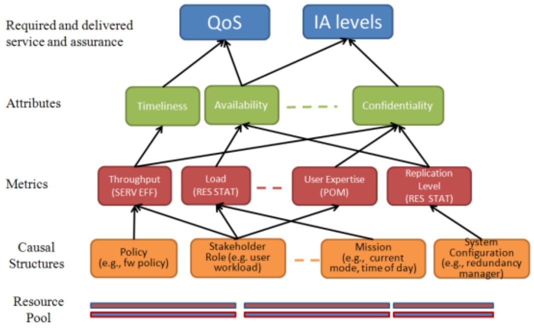

Required and delivered service and assurance
QoS
IA levels
Attributes
Timeliness
Availability
Confidentiality
Metrics
Throughput (SERV EFF)
Load (RES STAT)
User Expertise (POM)
Replication Level (RES STAT)
Causal Structures
Policy (e.g., fw policy)
Stakeholder Role (e.g. user workload)
Mission (e.g., current mode, time of day)
System Configuration (e.g., redundancy manager)
Resource Pool

Figure 1: Attributes, metrics and causal Structures

The notion of causal structures is based on the understanding that information systems are composed of a limited set of functional building blocks such as hosts, routers, operating systems and application software, whose roles and responsibilities within the information system can be described in generic and high level terms. We consider three classes of building blocks:

- Hardware such as routers and compute nodes.
- Software such as operating systems and applications.
- Defense mechanisms, representing hardware or software security mechanisms such as firewalls and virus scanners.

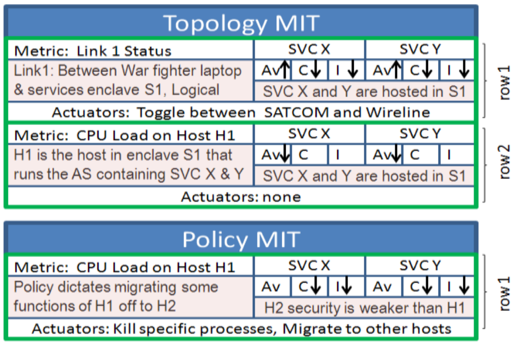
The building blocks collectively represent the information systems infrastructure the mission depends on.

Causal structures are the physical and logical relationships among the specific building block instances in an information system. Clearly, the causal structures in a given system are context sensitive, i.e., dependent on the specific mission and system at hand. However, we argue that these causal structures are, in effect, context specific customization of a small number of generic ones. The causal structures we have identified by examining real and simulated mission systems so far can be described in terms of four generic causal structures.

- Topology and physical dependency: Physical connection and location of building block instances with respect to each other within the information system influences the outcome of manipulation of building blocks and resources. The simplest illustration is resource sharing. The defense mechanism responsible for encrypting responses produced by a service, and the service responsible for actually producing the responses may run as part of the same process e.g., as in Application Server(AS) container. By increasing larger encryption keys to strengthen confidentiality will therefore increase the response time of the AS.
- Logical dependency: Even without any direct physical linkage, building block instances can be dependent on each other because of transitive interactions or shard dependencies. Such dependencies dictate the impact of manipulation of building blocks and resources on measurable metrics just like physical dependencies.
- Policy: Distinct from logical dependency, which is derived from functionality, policy represents a logical dependency that is administratively imposed. The case of firewall policy impacting throughput mentioned earlier is an example. Other examples include replication policy, access control policy etc. each of which forces a certain kind of interaction among the building blocks.
- Mission role: The tasks mission stakeholders need to perform at various times within the mission also impact the outcome of manipulation of building blocks and system resources. For example, if there is only one user being serviced, raising the level of encryption for that service does not impact the response time of other users.

However, this list is not exhaustive and may evolve as our work matures.

The generic causal structures actually provide the assurance engineers a starting point to identify the actual causal structures at play in the system at hand. By examining the topology, logical dependency, applicable policies and mission roles, the assurance engineer can identify the context specific ways in which causal structures can be manipulated, and how such manipulation will impact the metrics, and in turn evaluation of QoS and IA attributes. The information obtained from the analyses of a class of causal structures is captured in what we call the Metric Influence Table (MIT) for that class (e.g., the Topology MIT or the Policy MIT). An MIT is a table where each row corresponds to a specific metric that can be measured in the system at hand. Each such row is decorated with the subset of actuators that can be used to manipulate the causal structures in that class. Each row further indicates how upward or downward movement of the metric influences the QoS or IA attribute (A) of specific spatial scopes (S) based on the dependencies induced by the causal structures of this class. For example, consider how the specific metrics are shown to be



Topology MIT

Metric: Link 1 Status	SVC X			SVC Y			
Link1: Between War fighter laptop & services enclave S1, Logical	Av↑	C↓	I ↓	Av↑	C↓	I ↓	row 1
	SVC X and Y are hosted in S1						
Actuators: Toggle between SATCOM and Wireline							
Metric: CPU Load on Host H1	SVC X			SVC Y			
H1 is the host in enclave S1 that runs the AS containing SVC X & Y	Av↓	C	I	Av↓	C	I	row 2
	SVC X and Y are hosted in S1						
Actuators: none							

Policy MIT

Metric: CPU Load on Host H1	SVC X			SVC Y			
Policy dictates migrating some functions of H1 off to H2	Av	C↓	I↓	Av	C↓	I ↓	row 1
	H2 security is weaker than H1						
Actuators: Kill specific processes, Migrate to other hosts							

Figure 2: Exampls of metric influence table content

directionally related (metric and attribute values move in the same or opposite direction) to the combination of (spatial scope, attribute) in Figure 2. The Topology MIT shows that, by using the topology information of the system at hand, we can deduce that if Link1 is up or have more capacity, the availability (Av) of services X and Y to the war fighter increases. However, if the link is down, the confidentiality (C) and integrity (I) of the services increase because no one, including the attackers, from outside can communicate with the services enclave. On the other hand, specific policies applicable to the system require that H1's responsibility be offloaded to H2 when CPU load on H1 is high. Because H2's security control is less strict than H1, this action will actually diminish the confidentiality (C) and integrity (I) of the services.

However, it is important to note that each MIT provides only *partial* information about the actuators that can be manipulated to change the metric as well as how the metric influences the attributes. An individual metric may appear in multiple MITs, and to obtain the fill picture one needs to union the rows corresponding to that metric from all MITs. The union results in a tabular representation *T*, customized for the system at hand containing the following information for each measurable metric: a) the actuators that can be manipulated to change the measurement, and b) how the movement of the measurement influences the attribute levels. The table *T* provides us a structure to reason about how manipulation of causal structures can adjust the levels of stakeholder-relevant attributes. Construction of MITs and row-wise union of the MITs to obtain *T* are key steps to capture the qualitative and directional relationships between metrics and attributes.

A natural question at this point is consistency of the directional relationship between metrics and attribute levels, and whether directional relationship without any quantitative measure of the movements (i.e., rate of change) is useful.

The consistency issue arises when the union of rows corresponding to a metric from multiple MITs indicates conflicting directionality (one MIT indicates metric and attribute move in the same direction, while the other indicates that they move in the opposite direction). Although we do not have a formal proof yet, our insight is that true dependency between metrics and attributes cannot be unpredictable (i.e., under some condition higher metric value means better, and in other case lower). So, if inconsistency is detected, either our analysis is wrong, or we have undiscovered dependencies. If inconsistency remains for a metric, our tradeoff disfavors manipulations that impact "inconsistent" metrics over manipulations that impact "consistent" metrics.

To understand the utility of the directional relationships in the absence of any quantitative rate information consider an impasse in meeting both the IA and QoS requirements for a specific spatial scope. It is easy to see that the table *T* provides a way to drill down and identify which actuators need to be manipulated. The challenge is to decide which attribute to sacrifice, for which stakeholder, and once that is decided, which actuators to manipulate, to what degree and in which sequence. Continuing with our theme of sharing the responsibility, we address the first challenge by requiring the stakeholders to specify their tradeoff preferences. Such specifications may be with respect to a single stakeholder, i.e., a war fighter indicating under what condition he can accept lesser levels of IA in favor of better QoS. It can also span multiple stakeholders, i.e., commander's requirements may take higher precedence than system administrators'. We are developing a simple language to capture such preferences and runtime support to analyze them.

For the second challenge, we note that *T* will have multiple metrics related to a single attribute, and similarly, a single metric may be related to multiple attributes. Moreover the actuators may be associated with multiple metrics, and furthermore, *T* provides only directional information and no quantification of the results of actuator manipulation. While this has the risk of unintended interference and thrashing (continuous readjustment triggering each other), this situation is not uncommon in control systems. Apart from various pre-programmed rules to prioritize actuator selection, we are exploring an iterative feedback based algorithm where actuators are manipulated in small increments and subsequent manipulations depend on the outcome of the previous ones. To estimate the order and increments, one possibility is to use Monte Carlo simulation of possible changes to causal structures and then estimating the relative changes in attributes.

V. Proof of Concept

We are developing a proof of concept prototype for demonstrating our notion of continuous mission oriented assessment of IA and QoS first, and eventually QoS-IA tradeoffs leading to managed mission assurance. In this prototype, we are leveraging existing capabilities as much as possible. For instance, modern system management mechanisms already collect various measurements—our prototype interfaces with existing system management mechanisms to obtain such measurements. Similarly, our prototype does not attempt to develop control mechanisms for mounting effective defensive responses—rather, it assumes that survivability management and human defenders are responsible for that, and considers the impact of such defensive responses in its assessment instead. For actuating the tradeoffs, our framework relies on existing QoS and survivability management mechanisms, but also accommodates direct interfacing with resource management, defense mechanisms and stakeholders directly as needed.

In our prototype, the top level entity responsible for assessment and tradeoff decisions is known as the Blackboard. The blackboard is an aggregation point for system measurements and observations; collecting information from a single host or set of hosts on the network (see Figure 3). Typically, each stakeholder has a dedicated blackboard where assessments for that stakeholder takes place (for example, a dismounted soldier will have a blackboard running on his laptop) but one or more blackboards could also be assigned to a whole stakeholder class (for example, depending on scale, there may be one or multiple blackboards shared by all administrators manning a network operation center). Multiple blackboards operate in a peer to peer relationship maintaining a loose synchrony. This peering relationship implies that information in one blackboard, including results of assessment computations and collected systems states local to the blackboard, can be shared with other blackboards.Figure 3

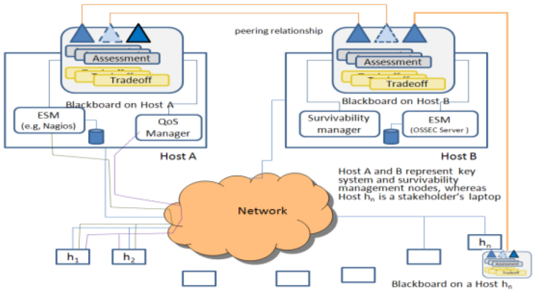


Figure 3: Runtime support for managed mission assurance

describes the configuration of our prototype. The mission support system consists of hosts connected in a network. Hosts A and B are locations of significant system and survivability management functions such as QoS managers, Nagios and OSSEC servers etc., and runs the blackboards. In addition, other hosts like a stakeholder's laptop indicated by the host h_n also run blackboards.

Blackboards are loaded with assessment and tradeoff objects—these encapsulate the stakeholder requirements and preferences. Measurements and observations are reported to the blackboard continuously. The blackboards also keep track of mission progress by subscribing to mission events. Therefore, at any point the blackboards are able to determine, based on the reported measurements whether the system is delivering the required levels of IA and QoS. If there is an impasse, i.e., the required levels $L_{A1, S1}$ and $L_{A2, S2}$ for one or more stakeholders are not met and the attribute- spatial scope pair (A1, S1) is causally related to (A2,S2), the tradeoff analysis kicks in.

We have demonstrated continuous assessment based on observed metrics against stakeholder specified requirements in this prototype. We are currently in the process of adding the compile and runtime support for tradeoff analyses.

VI. Conclusion

In this paper we introduce the concept of "managed mission assurance", and describe methodologies and runtime support required to manage mission assurance as a tradeoff between conflicting QoS and IA requirements. Mission assurance is defined as continuation of mission essential functionality with adequate levels of IA and QoS. Managing mission assurance is critical for mission success in a contested cyber space.

Our approach is based on runtime assessment of the level of QoS and IA delivered by the system, and addressing the impasses when both IA and QoS requirements cannot be met by trading one in favor of the other. To facilitate the runtime assessment and tradeoff analyses without having to solve the hitherto unsolvable "quantitative evaluation of security", we adopted a requirements engineering and risk analyses based approach that is qualitative, and forces the mission stakeholders to share some responsibility.

Qualitative reasoning is a key aspect of our approach. This is most prominent in both the assessment (against stakeholder specified *levels*) and tradeoff analyses (the *directional* dependency and iterative actuation). Previous QoS approaches have been more quantitative but this approach is inadvisable in the IA domain due to the general lack of quantitative grounding neither for effective evaluation nor for the end users. Specifically, stakeholders can rarely, if ever be expected to quantitatively describe their tradeoff preferences or IA goals.

This work is in its early stage. However, as described, we have started to develop a proof of concept to demonstrate managed mission assurance. We fully expect that the initial methodology and algorithms described in this paper will evolve as we progress and experiment with the proof of concept.

Acknowledgement

This work is funded by AFRL under the Contract No. FA 8750-08-C-0196. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

References

[1] U.S.Navy Cyber Defense Operations Command, obtained from: http://www.novell.com/rc/docrepository/public/32/basedocument.2009-02-03.6639189304/U_S_Navy_Cyber_Defense_Operations_Command_Case_Study_en.pdf

[2] Active Event Monitoring for Improved Security and Compliance Management: http://www.novell.com/rc/docrepository/public/37/basedocument.2007-08-07.6773785086/4622035PRINT_en.pdf

[3] Securing the Cloud: http://www-sanmateo.akamai.com/dl/whitepapers/Akamai_ISMS.pdf?campaign_id=$(campaignID)

[4] Retrieved from Tivoli Security Information and Event Manager: ftp://ftp.software.ibm.com/software/tivoli/solutionsheets/TIS10415-USEN-01.pdf

[5] Symantec Enterprise Security Manager: http://eval.symantec.com/mktginfo/enterprise/fact_sheets/ent-factsheet_enterprise_security_manager_6.5_06-2005.en-us.pdf

[6] Steven A. Harp, Richard C. O'Brien, Charles N. Payne, Johnathan Gohde, John Maraist J. Thomas Haigh, "Trapping Malicious Insiders in the SPDR Web," In *Hawaii International Conference on Systems Sciences (HICSS-42)*, Honolulu, HI, 2009.

[7] NIST SP 800 30, available at : http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf

[8] DISA Mission Assurance and Network Operatins Program Executive Office Homepage: http://www.disa.mil/peo-ma/

[9] Joseph Loyall and Richard Schantz. Dynamic QoS Management in Distributed Real-time Embedded Systems. In Handbook of Real-Time and Embedded Systems, Insup Lee, Joe Leung, Sang Son (Eds), 2008.

[10] Jay Lala (Editor). Foundations of Intrusion Tolerant Systems. IEEE 2003.

[11] Partha Pal and Patrick Hurley. Assessing and Managing Quality of Information Assurance. In NATO Symposium on Information Assurance and Cyber Defense. Tekirova, Turkey, 2010