From Auto-adaptive to Survivable and Self-Regenerative Systems – Successes, Challenges, and Future

Michael Atighetchi
Information and Knowledge Technologies Business Unit
BBN Technologies
Cambridge, MA
matigheti@bbn.com

Partha Pal
Information and Knowledge Technologies Business Unit
BBN Technologies
Cambridge, MA
ppal@bbn.com

Abstract—This paper charts the course of adaptive behavior in intrusion tolerance, starting from pre-programmed and user-controlled reactive adaptation to highly sophisticated autonomic and cognitively driven adaptation. The goal of intrusion-tolerance is to provide mission continuity even under conditions of sustained cyber attacks. We describe key themes of our previous work in adaptive cyber defense and introduction of autonomic response capabilities and discuss quantifiable progress as well as challenges that warrant further research. We also discuss the potential impact of new trends in distributed systems, e.g., service-oriented architecture and cloud computing, on future survivable systems, and point out new opportunities for developing sophisticated auto-adaptive capabilities for increased survivability.

Keywords: Intrusion Tolerance, Information Assurance, Survivability, Cognitive Algorithms, Autonomous Computing

I. INTRODUCTION

Defending an information system against cyber-attacks is an arms race that is inherently asymmetric and favors the adversary. Adversaries only need to find a single exploit, whereas defenders need to prevent as many of these exploits from succeeding as possible. At the same time, adversaries are able to find more opportunities for attack as information systems become increasingly more complex and interconnected, opening up new flaws and vulnerabilities in the underlying software components, infrastructure, and interconnections.

Approaches for mitigating this asymmetric threat have changed over the last two decades from a static model of security that tries to build failure free systems to a more dynamic and adaptive view of defense that assumes that failures are inevitable and the key for surviving attacks lies in the way failures and their consequences are handled. First generation security mechanisms were solely based on protection mechanisms, including encryption and physical network separation. The main problems with static protection include its binary and static nature and the fact that straight isolation became increasingly impractical as systems began to be more connected and users started to demand access to information across isolation boundaries. The second generation of security mechanisms placed strong emphasis on signature-based detection and root cause attack analysis, so that these could be used as a trigger for remedial actions by human experts. While detection and reactive reconfiguration is appropriate for availability attacks, constructing signatures for detecting privilege escalations or confidentiality breaches has proven quite difficult. In addition, human-scale response time has proven ineffective or inapplicable in many system contexts, requiring rapid detection or relaying on some systems to be compromised so that others can be protected. The goal of third generation approaches is to tolerate attacks even if they cannot be detected in a reliable way. Approaches range from effects-based techniques for attack detection to proactive reconfiguration and byzantine-fault tolerant replication. Although third generation systems generally provide means for graceful degradation, they will eventually run out of resources in the presence of determined attackers and fail to provide service. Recent research has started to look at self regeneration capabilities to replenish lost resources and learn better cyber-defense strategies for improving survivability over time.

Recent improvements in technology areas such as service-oriented architecture, cloud computing, virtualization, and semantic linking have enabled more dynamic ways of system construction and resource management. Security and survivability architectures and associated designs and mechanisms need to keep up with the increased agility, dynamism, and fluidity of commoditized resources. In our view, cognitive approaches for cyber event interpretation and action selection, learning, and emergent properties all fall into scope as building blocks that, if assembled correctly, will lead to realizing the self-healing, self-managing, self-improving (collectively called self-* ) properties of survivable systems.

This paper describes our work in intrusion tolerance over the last decade and describes a steady progression of increased...
agility, situational awareness, and autonomic defensive adaptation being built into information systems. We describe the key technical enablers together with our efforts in validating the resulting benefits. Despite good progress, several hard challenges still remain relevant in today’s environments and warrant further research. In addition, trends in recent distributed middleware technologies have made certain security concerns easier to address, e.g., via appropriate use of redundant resources, while raising the difficulty significantly in others areas, e.g., identity and privacy management and data confidentiality.

The organization of this paper is as follows: We present our work in section II, and describe hard challenges for providing cognitive defense components in section III. Section IV describes new trends in distributed systems and new opportunities for building systems with autonomous defense capabilities that can maintain a high level of survivability and continue to operate through attacks. Section V concludes this paper by summarizing our position and recommending new research directions for cognitive survivability.

II. PAST WORK IN INTRUSION TOLERANCE

Much of our work in this area stems from the early work on the Quality Objects (QuO) [1] adaptive middleware for implementing quality of service in distributed object platforms. We have extended and used our middleware capabilities in multiple different contexts such as intrusion tolerance, distributed real-time embedded (DRE) systems, and network management. While Figure 1 displays a time table of the intrusion tolerance projects executed by BBN, the rest of this section describes the logical progression in the form of

a) Creating a systematic approach for defense-enabling applications
b) Creating reusable, autonomic dynamic responses
c) Adding unpredictability to dynamic responses to make initial compromise more difficult
d) Tolerating arbitrary corruption once components have been compromised
e) Integrating artifacts into a survivability architecture
f) Automating cyber security decision making
g) Validating claims in a) to f)

Figure 1 - A decade of research in intrusion tolerance

A. Defense Enabling

One of our earliest insights was to treat intrusion tolerance or survivability as an intrinsic property of applications and, in turn, of distributed systems that consist of interoperating applications. Consequently, we developed a technique to integrate security and survivability mechanisms with applications such that the applications can participate in their own defense. This software engineering process, called defense-enabling, has been the center-piece of our work in survivability and intrusion tolerance. Defense-enabling integrates defenses with functionality of the defended application for a given threat and resource environment. We use a middleware-based integration approach that facilitates encapsulation of defenses into generic components that can be (re)used by multiple defense-enabled applications. The first step in defense enabling is to identify which potential attacks are most fatal for the application and rank them by likelihood and cost to the attacker. For example, attackers may be able to easily kill processes but it might be significantly more difficult to cause arbitrarily corrupt behavior. The next step is to develop defense strategies, focusing on the set of likely attacks that impact the system most first. The final step is to realize the strategies in middleware layer, which provides a clean separation between the application’s functionality and the additional defensive capabilities. This separation allows independent development of defenses separate from application development, and facilitates reuse.

B. Autonomic Dynamic Response

The experience of defense-enabling multiple applications led us to encapsulate a number of reusable defenses strategies and tactics in reusable packages [1].

Figure 2 - Manually created defense strategies

Figure 2 displays the set of strategies we developed along two dimensions. The y axis shows the abstraction layer ranging from mechanisms over tactics and sub-strategies to an overall defense strategy. Mechanisms provide individual capabilities provided by commonly available tools, such as firewalls or anti-virus software. Tactics are reactive defenses that use the capabilities of a small number of mechanisms. Typically, a tactic combines a sensor with an actuator mechanism to adapt to local situational changes. The basic objective of a defense strategy is to increase an application’s survivability through automated coordinated defensive behavior. The overall strategy is to significantly improve the first line of defense by managing the shortcomings and failures of the protection it provides. The x axis shows the increasing amount of coordination required, ranging from local isolated deployments of instances, over distributed deployments with specialized protocols between instances, to a system-wide scope involving coordination...
between multiple protocols. As shown by the dashed red line in Figure 2, hierarchical decomposition and binding of an overall defense strategy into a set of relevant sub-strategies, tactics, and mechanisms are possible. This notion has interesting similarities to orchestration and choreography in service-oriented architectures (SOAs), which we will discuss in section IV. The following are representative examples of tactics illustrating the scope of defenses developed in the APOD project:

Squelching insider floods: This tactic monitors traffic flow and adaptively activates rate limiting upon observing packet floods. The tactic continuously monitors outgoing traffic via Iptables and calculates throughput metrics such as packets per second and bits per second. It then compares mean values for observed and expected parameters at regular intervals; if the observed outgoing traffic is significantly higher than expected, the tactic changes routing configurations at the boundary controllers to limit outgoing traffic.

Restoring corrupted files: Upon sensing integrity violations of critical files via file integrity tools, this tactic restores corrupted files from a non-corruptible backup medium (such as a CD-ROM).

Shutting down hosts: However hard a defensive system tries to keep an attacker out, the attacker might still find some way to take over one of the hosts and use it as a launching point for further attacks. In such cases, an adaptive system could fall back on the tactic of halting suspected hosts. However, blindly shutting down hosts based on information obtained from a local intrusion detection system (IDS) information can leave the system vulnerable to attacks that cause self-inflicted denial of service. Therefore, this tactic should not be deployed in isolation but rather as an integrated part of an attack-containment strategy.

C. Use of Unpredictability

Defensive responses that are deterministic eventually become ineffective because, given sufficient time to observe the system and its defenses in action, the adversary will be able to predict defensive responses and plan accordingly. For instance, knowledge about where new replicas are spawned in a replication system enables attackers to plan and script complicated multi-step attack scenarios that maximize attack effectiveness at a low cost. In the ITUA project [3], we studied the use of unpredictability in adaptive responses at various OSI layers to increase the level of difficulty for creating scripted attacks. We injected uncertainty on the network layer by utilizing a NAT-based port and address hopping scheme that dynamically changed endpoint information at regular intervals and changing low-level socket behavior caused by deny rules in firewalls, e.g., ICMP reject vs. silent dropping of packets. At the application layer, we implemented replication managers that implement Byzantine fault-tolerance (more details in the next section) protocols to kill corrupted replicas and start new replicas following a probabilistic approach that imposes uncertainty on attackers.

D. Byzantine Fault Tolerance

Straightforward use of redundancy can tolerate crash failures. But a sophisticated adversary can cause more damage by corrupting the application that lead to Byzantine failures. To address this shortcoming, we developed a replication architecture under the ITUA project that can tolerate arbitrary (Byzantine) corruption of replicas. This framework, shown in Figure 3, transparently replicates existing application objects through gateways, actively manages replicas by killing corrupted replicas and starting new ones, and incorporates voting algorithms that can tolerate arbitrary corruption of replicas. Hosts are organized into security domains, which provide containment and diversity boundaries. Each host runs an ITUA Manager process, which uses the gateway to implement a specialized voting protocol for determining replica lifecycle actions. Replicas and application clients are connected via the ITUA gateway as well and implement the Byzantine fault-tolerant replication and connection groups.

Figure 3 - The ITUA Architecture

E. Survivability Architecture

After developing a number of defense mechanisms and using them in support of various defensive strategies and tactics, we attempted to see whether a survivable DoD-relevant information system can be developed, using currently available defense mechanisms and security technologies.

As part of the DARPA OASIS Dem/Val and BBN’s DPASA project [4], we designed and evaluated a high water mark survivability architecture that provides strong guarantees for attacker tolerance and survival, intrusion detection and situational awareness, and design flexibility. The main goals of the resulting architecture were to

- Create a very high barrier to entry for an attacker, both for initial intrusion and moving to other parts of the system after gaining initial access.
- Maximize the likelihood of a sensor being tripped by attacker activity by monitoring all parts of the system at all layers, perhaps in a redundant and overlapping manner.
- Support dynamic reconfiguration of the system, with or without human intervention, to recover from damages caused by attacks or cope as long as possible with damages that cannot be repaired.

Redundancy plays an important role in this strategy. It provides tolerance by masking individual failures or providing
alternative options to continue operating. However, it also offers the adversary more places to attack or steal data. We designed the survivability architecture in a way that combines redundancy with diversity to ensure that an attacker needs to find two significantly different attacks to cause a failure in the system. The high-level strategy also favors aggressive monitoring, but aggressive monitoring means generation and transportation of more low-level alerts. This, combined with the alert reduction and meaningful alarm requirements, necessitates a sophisticated correlation mechanism that includes both near-sensor and centralized correlation to aggregate, filter, and rank low-level alerts.

The architecture was based on a number of sound survivability design principles, including

- Single point of failure protection through increased redundancy for critical components (see redundant core quads in Figure 4)
- Creation of physical barriers through zones (shown in Figure 4) to make privilege escalation more difficult
- Controlled use of diversity to include some diversity in each access path to key assets
- Creation of a robust basis for defense-in-depth by implementing base algorithms as trusted hardware
- Enforcement of containment regions to limit spread of attacks across boundaries.
- Adaptive responses that quickly mount actions that have localized impact or can be easily reversed while deferring to humans for more coordinated and dangerous activities.
- Minimalism to eliminate unnecessary features
- Automated configuration generation from specs to limit inconsistencies

![Figure 4 - DPASA quads and zones](image)

**F. Cognitive Approaches to Survivability Management**

The state-of-the-art survivable system, built and evaluated in the OASIS Dem/Val program, showed excellent resiliency in containing and thwarting attacks, even when the adversary was given considerable access and system privileges, described in more detail in Section G. Expert human involvement was needed, however, to interpret alerts and incident reports and to make decisions about defensive responses. Human involvement in this manner is expensive, and motivates the essence of autonomic control envisioned in the next generation of self-regenerative survivable systems.

The goal of the CSISM project [5] was to develop automated reasoning mechanisms that when incorporated in the survivability architecture will minimize the role of human experts and pave the way for truly self-regenerative survivable systems.

![Figure 5 - Elements of CSISM multi-layer reasoning architecture](image)

**G. Validation through Experimentation**

Experimental validation of claims made by the various technologies outlined in Sections A, through F, has been a central aspect of our work. This has lead to a number of formal independent red team exercises, indicated by dotted vertical lines in Figure 1, together with concrete results measured by key survivability metrics and many lessons learned.

In the APOD red team experiments, we offense-enabled a publish/subscribe system and subjected it to a sophisticated adversary. The adaptive defenses successfully slowed down attack progress, with a resulting minimal time of denial of 19 minutes as compared to immediate denial within seconds for the undefended system.

As part of the evaluation for the ITUA project, we offense-enabled parts of Boeing’s EIST publish/subscribe system [6] and showed tolerance against sophisticated corruption of replicas and managers and graceful degradation in the presence of multiple failures.

Evaluation of the DPASA survivability architecture was performed in two formal red team exercises, involving two
sophisticated and mostly unconstrained red teams with full knowledge about the architecture and code. Analysis of the results showed that the approach to designing survivability into an information management system taken by the BBN Technologies team was effective in making it hard for an attacker to intrude into the system and cause significant damage. During the second experiment, the starting point was a single compromised host under complete control of the red team, and the goal was to evaluate the effectiveness of the architecture in tolerating and containing the spread of attacks. The Red Team considered a wide range of attack possibilities to violate confidentiality, integrity, and availability of the system. Many of these possibilities were nullified by the design of the system, leaving a set of ~24 attack ideas, of which 19 were fully developed and executed. The Red Team was able to stop the mission from completion within the stipulated time in only 4 of the 19 runs. The mission was successfully completed in the remaining 15 runs effectively tolerating, recovering from, or adapting to damages inflicted by the attacks. The survivable system did not experience any compromise in confidentiality and integrity even with high level of access inside the system and pre-positioned attack code. The architecture provided good attack notification and localization, and provided appropriate knobs and levers to expert operators to manually deal with attacks.

The CSISM red team exercise demonstrated that the goal of automating expert-level cyber-defense decision-making was achieved. The tests conducted during the exercise did not involve any human operator performing cyber-defense activities. Based on the red team’s report, CSISM detected 88% of all attacks, and of those detected, generated effective immediate responses in 86% of cases. This implies effective response in ~75% of all attacks. For timeliness, CSISM espoused focusing on the response time of the reasoning engines, i.e., the time taken by the reasoning engines to process input information. We measured response time of local responses in the order of milliseconds, whereas the response time of more deliberative and system-wide reasoning was in the order of seconds. The CSISM reasoning engines demonstrated continued effort to survive, even if the initial response was ineffective within a run. However, CSISM’s ability to improve over time across multiple runs is currently limited and was not reliably demonstrated (i.e., the learning runs during the exercise failed to converge, whereas other runs did).

III. CHALLENGES

Our work in survivability and intrusion tolerance has demonstrated that it is possible to build autonomic intrusion tolerant systems that hold up to sophisticated and unrestricted attacks by dynamically responding to provide graceful degradation, failure isolation, and self-healing. However, in implementing example systems and subjecting them to attacks, we encountered a number of technical challenges, solutions to which we believe are necessary to advance the current state of intrusion tolerance to the next level of sophistication. This section describes a targeted selection of challenges and potential ways forward, organized by the categories of a) cognitive approaches, b) biologically-inspired selection functions, and c) evaluation of and utility functions for survivability.

A. Challenges in using Cognitive Approaches

The CSISM project demonstrated the value of cognitive approaches in automating system-wide action selection in noisy environments with incomplete knowledge. The focus in the project was on cognitive algorithms in the form of coherence theory and associated proof algorithms that generate correct hypotheses about the system state in terms of accusation against components. Most of the action selection was implemented through manually created if-then-else rules over selected hypotheses.

We realized early on that instead of manually creating the action selection function, we could implement a framework for learning this function and optimizing for certain requirements, such as minimizing timeliness or cost. However, we encountered a number of significant challenges for which solutions were outside the scope of CSISM.

Learning in adversarial environments: To cope with unforeseen events, it is beneficial to include a learning component into our survivability middleware that is able to learn from mistakes as well as successes. In addition to standard problems of learning systems, such as the quality and quantity of the training set, learning in the cyber defense domain needs to account for the fact that intelligent adversaries are part of equation. Attackers may trick the learning system into learning the value of unimportant actions while holding back the real attack actions (and associated effects) to the very end. In CSISM, although the learner had access to all the information over all runs, it was unclear how much and especially which parts needed to be carried across various runs and whether it was safe for the learner to tweak the reasoning mechanism online—especially when there is no certainty about the occurrences or ordering of attack repetitions. Furthermore, learners rely on the notion of a quality signal, which indicates whether the system is offering good/acceptable service or not; and more importantly indicates to the learner when the service quality becomes unacceptable. Typical quality signals are multivariate, and incorporate a variety of views including system availability, costs to users, and damage. No such formalism exists in the context of cyber-security decision-making and in fact the notion of quality of cyber security or information assurance is highly subjective.

Simulation-based experimentation: To support validation of intrusion tolerance we created a high-level simulator that can model abstract concepts (e.g., hosts, networks, clients, and servers) and associated protocols between them (e.g., TCP flows, pub/sub interactions). We also developed a library of common attack effects that can be injected into the simulator, including host crashes, network partitions, file corruptions etc. The main idea was to use a simulator instance as an abstract executable model of the real environment. To support learning, we added a snapshot capability and supported flexible quality functions over the system state. Given such a model, the online reasoning engine could perform what-if analysis by injecting an effect that corresponds to a recently detected attack and perform a pruned forward-chaining search to find the optimal sequence of defense actions that mitigates the attack. Besides
B. Biologically-inspired selection functions

Encoding “ethical” invariants: Adaptive strategies can be used by intelligent adversaries and turned against the system they are trying to defend. We encountered one notable instance during the APOD red team exercises, where the red team tried to subvert the selection function for shutting down corrupted hosts and trick the APOD system into isolating inappropriate hosts to cause self-inflicted denial of service. Introduction of stronger coordination in the formation of the selection function fixed this particular instance of the problem, but we expect similar problems to resurface in the future. We envision formulating and enforcing “do no harm” invariants analogous to safety invariants on mechanisms that mount defensive responses, especially the cognitive mechanisms responsible for cyber-defense decision making. No such framework exists in cyber defense, but biological organisms seem to exhibit this as an intrinsic trait.

Parameterized redundancy and diversity distributions: Redundancy is an important aspect in providing robustness and intrusion tolerance, but it comes at a high price. Diversity takes the same argument to an even more extreme level, despite the recent progress in artificially introduced diversity. A system that scales down its use of redundant and diverse resources in normal mode and scales up when under attack is clearly desirable from a cost conservation point of view. However, systems that rely heavily on such a strategy are vulnerable to stealthy attacks that only become visible when it is too late and no redundant resources are available any more. Although we had good success with 4-fold redundancy and graceful degradation in the DPASA survivability architecture, the initial cost of 4 times increase in hardware components have proven prohibitive in most environments. New research is needed to dynamically adjust redundancy and diversity distributions in a reliable and trustworthy way. Biological systems may provide inspiration in the form of dynamic management of diversity distributions of attributes (genes, behaviors) across populations.

C. Effective Evaluation of Survivability

Evaluating individual security mechanisms and tools in isolation is significantly different from evaluating system security, especially in a quantified way. In fact existing methods of evaluating system security are often focused on software processes and organizational practices. The accepted standard for system-wide evaluation is variations of red teaming such red team experiments, adversarial and penetration testing, collaborative red teaming, white boarding etc. Other prominent methodologies in use include model based probabilistic validation and logical argumentation. They all have their pros and cons. For example—the outcome of red team-based evaluations depend significantly on the expertise of the red team, Red team evaluation is costly and time consuming, it is impractical to conduct many red team exercises, and therefore results from a red team exercise should not be taken as the be all and end all evaluation results of a system. Similarly, model based evaluation suffers from the problem all model-based approaches suffer—it is only as good as the model is. If the model is too abstracted from the real system, the results are not realistic. On the other hand, building a high-fidelity model can be as expensive as building the system itself.

Having experienced several attempts to evaluate survivability, we can outline a number of ideas that we think will be useful. First, evaluation of survivability should not depend only on experimental techniques or scores obtained from experimental measurements. Quantitative measures can also be obtained from non-experimental techniques and they should also be used in constructing the assurance case. For example, the number of attacks or attack classes eliminated, or number of defense mechanisms that the adversary must overcome in order to achieve for each of a given set of attacker objectives can be estimated by logical analysis and white boarding. These numbers also provide very useful data points for rating an individual survivable system as well as comparing one with others.

Second, intervals, such as “time until certain event” seem to be useful for quantifying survivability. The event in question could be a Red Team achievement, a scenario milestone, or the point at which the system degrades beyond being usable. A good experiment design should include at least one such “time to” metric, in addition to other applicable time-based (i.e., round-trip time, uptime, completion time of an operation) or count-based (i.e., number of alerts, number of successful or failed events) or rate-based (i.e., rate of successful transactions) metrics.

Third, easily measurable and intuitive metrics tend to be mission or application specific. However, making them too specific can be problematic. For instance, a successful publication in a redundancy-based publish-subscribe protocol may involve multiple retries. In this case, counting low-level publish/subscribe messages instead of the externally visible event of successful publication would be inappropriate. Counting preservation of confidentiality (C), integrity (I), and availability (A) in information objects separately exposed some of these issues in OASIS Dem/Val. Very high C and I scores were possible with a low A score, seemingly indicating some success in “retaining critical functionality” despite a failure of the mission. Evaluating the effectiveness of specific defense mechanisms is useful and a number of the 20 focused exercises in the 2nd phase evaluation of Oasis Dem/Val were of this nature. However, we argue that high-level metrics and requirements should be formulated in terms of the system’s external behavior.

Finally, the OASIS Dem/Val program demonstrated a way to conduct adversarial tests where the Red Team flavor can be retained in lightweight and focused exercises. This approach was quite useful in assessing the defended system and we believe represents another building block for making more comprehensive assurance cases.

IV. NEW TRENDS AND OPPORTUNITIES

Recent advances in distributed computing have extended the Internet from a medium to connect multiple computers together to a global platform supporting flexible, interoperable,
and dynamic interactions between clients and services. This new platform faces some of the challenges with respect to intrusion tolerance we previously outlined, but at the same time provides potential for making significant progress in some technical areas easier while exacerbating the need and difficulty in others.

SOA [7] and Cloud Computing [8] provide means to dynamically manage redundant computation resources in a structured way. Cloud Computing can reduce a lot of provisioning issues and enable “on-click” dynamic provisioning of computing power and storage. The SOA concept implies that software building blocks, including security mechanisms, can now be thought of as services, potentially developed independently, to be connected to a service bus. Combining SOA and cloud has the potential to make intrusion tolerant architectures affordable, reusable, available, and dynamically adjustable to different environments and risk profiles.

However, indiscriminate migration to SOA and cloud computing can be dangerous. In addition to compute power, storage, or connectivity, the cloud must also offer a level of trust and protection. While distributing the processing of critical information across cloud nodes offers opportunity to load balance, improving availability, and to perform voting, improving integrity, it may significantly weaken confidentiality of mission critical data. Standard approaches for data confidentiality are based on cryptographic encryption (in transit or at rest) and delegation of service execution to innermost domains. Although encryption techniques remain applicable in the cloud, the data needs to get decrypted for a service to make use of it, and since the service is located somewhere in the cloud, this may open up the possibility of data leaks. New research in parallel algorithms and distributed split-data processing (analogous to split-key cryptography) could provide confidentiality guarantees because no single service in the cloud will have access to the all mission critical information units. This is clearly one area we are interested in pursuing. In addition, we see a strong need for advanced autonomic control algorithms that dynamically adjust redundancy and diversity distributions based on mission requirements and environmental parameters.

Moving to a SOA also means composing applications through a set of service interactions, which may break underlying assumptions made by individual services and introduce unforeseen vulnerabilities. Safe composition of an intrusion tolerant system of systems will require orchestration processes that preserve safety properties of components and flag inconsistencies in the composed systems. While languages like the Business Process Execution Language (BPEL) [9] offer ways to describe abstract executable business processes and workflows and their mapping to services, they currently only focus on representing publicly observable functional behavior of abstract processes in a standardized fashion and do not address systemic cross-cutting properties, such as robustness, security, or trust. This finding is similar to the argument made in [10] on the lack of resource management, exception handling, process variation, and data flow integration. Applying algorithms that operate on mission models and perform orchestration to maintain mission survivability is an exciting new research area that may change the way intrusion tolerant systems of systems are developed in the future. Biological systems may provide starting points for construction of complex yet robust systems of systems, but careful analysis is key to avoiding the problem of metaphoric reuse without making improvements.

V. CONCLUSION

In this paper we described our continued progression toward realizing the vision of better managed and agile distributed systems. Such systems can adapt their configurations, resource usage, even functional behavior to accommodate changes in their operating environments, including those that can be caused by a malicious adversary. We started with the understanding that defenses must be organically ingrained with the system, as opposed to an afterthought that introduces defensive mechanisms in the network and the hosts without much consideration of the end-to-end systemic behavior or the applications’ needs and with the realization that along with prevention and detection, dynamism, in the form of adaptive defensive response is essential for survival. In our work we have experimented with security mechanisms and fault-tolerance techniques such as redundancy, diversity and associated protocols, developed survivability architectures with multiple levels of defenses and strong containment and isolation properties, and also a range of adaptive behavior spanning from knee-jerk local responses (with ms time scale) to more deliberative responses that need to analyze a number of contingencies and potential outcomes. As we begin to introduce new software development methodologies like cloud computing, on one hand there will be more opportunities to do better analysis of intrusion detection data, provisioning more compute or memory resources on demand leading to more intelligent system behavior and newer defensive responses. The smarter system management and reactive-deliberative-introspective response capabilities that we can do today are already inspired by biology. In the future, the fine grained redundancy of digital data and processing capabilities spread in the cloud provides the basis for developing more biologically-inspired/cognitive survival mechanisms. Right now, we are able to contain and gracefully degrade in the face of a sophisticated adversary, in the future we aim to regain the lost capabilities, regenerate (or re-provision) lost resources from the cloud, and most importantly refine the defensive logic introspectively using machine learning techniques to learn to improve the system’s defenses from past mistakes and successes. On the other hand, SOAs and cloud computing cognitive autonomic defenses introduce new vulnerabilities around confidentiality, privacy, and trust. Once again, biology may provide the starting point in terms of addressing these-- in particular, self-regenerative systems will need to be aware of and address the "auto-immune" issues where the system mistakenly (either by itself or by way of some fault) disables its defenses or attacks itself. The current software landscape and state of survivability research is at a point where investigation into these issues is not only possible but also becoming more important to undertake.

REFERENCES


[9] OASIS Standard WS-BPEL 2.0