14th ICCRTS: C2 and Agility

"Mission Assurance in a Distributed Environment"

Topic 7: C2 Assessment Tools and Metrics

Authors

Chad DeStefano

Thomas Clark

Point of Contact:

Chad DeStefano

Air Force Research Laboratory Information Directorate

Cyber C2 Branch (RISF)

525 Brooks Road, Rome, NY 13441-4505

315-330-4286

Chad.DeStefano@rl.af.mil

Mission Assurance in a Distributed Environment

Abstract

The increased use of machine assistance has opened new doors for adversaries to thwart coalition planning processes and systems using the next generation of sophisticated cyber exploits. These new attack vectors, aimed directly at our core mission-essential systems, will not close during normal operations and will require Command and Control (C2) planners to operate in contested environments regardless of contingencies. Operating in this increasingly hostile environment requires confidence in our infrastructure beyond what traditional information assurance addresses - we require the wherewithal to sustain sufficient planning capability to fulfill vital objectives in the face of adversity. This *mission assurance* concept will allow the critical components of the planning process to be sustained instead of degrading so severely that machine assistance is fully lost. The groundwork for achieving mission assurance will be discussed in this paper and we will examine a distributed planning program in detail as an initial exemplar for applying a new breed of mission assurance techniques.

Contents

1 Introduction

If detecting threats in cyberspace were as easy as spotting poorly written emails sent as obvious phishing expeditions from bogus accounts, we could simply ignore them and move on, assured we avoided any potential pitfalls. Despite the sheer volume of messages we may have to wade through, there is some small comfort knowing that phishing is not a very sophisticated threat because everyone seems to be doing it. The real threats to our status quo are those computer programs or decision logic deployed by highly coordinated teams of programmers in complete stealth over prolonged periods of time that can impact decisions or outcomes in subtle or not-so-subtle ways.

Recent newsworthy examples of highly motivated adversaries and coordinated threats in the kinetic world are: nation-state sponsored experiments that temporarily blinded sovereign satellites using lasers, the actual recorded destruction of a satellite in space, and GPS jamming during Operation Iraqi Freedom (Martin, 21). Understanding blue force reliance on technology in the non-kinetic domain will prepare us for the inevitable onslaught on our military networks such as the outages observed during the Gary McKinnon incident (Wilson, 16).

1.1 Problem Statement

To understand the escalating danger, consider a few additional examples. A single hacker of intermediate skill level could deface a website or conduct a small-scale distributed denial of service attack on a few nodes; however, the act itself will be noticed as soon as a carriage return is hit. There are more covert attacks that may take time to discover and require the cooperation of several individuals or organizations to piece together, such as: identity theft, a one-time, unauthorized access to a database, or an intruder that ensures proper operation of his own exploit by applying patches to keep other malware out.

These examples give way to a new class of threats emerging that may be impossible to detect because they are designed to influence in subtle ways (or none at all), over time, until a threshold is reached and critical components of an organization's mission are swiftly undermined. A worse scenario is not achieved by sudden death, but *death by a thousand cuts*, whereby the victim is totally oblivious to the exploit and the attacker accomplishes all its goals in absolute silence. In the military, an example of the former is to slightly alter a target's coordinates in order to achieve maximum collateral damage for immediate political gain. An example of the latter is persistent and calculated decrements of inventory in a database, coupled with embezzlement, until an enemy builds up his own arsenal while simultaneously decreasing his enemy's capability. In these last two examples, the primary attribute that sets them apart from other types of attacks is that friendly objectives are compromised, forcing organizations to shift emphasis on maintaining *mission assurance* in the face of attacks.

To further complicate matters for the military, many of its information processing systems operate over computer networks at geographically dispersed locations. Of those systems, it is the scheduling or campaign planning systems that are the most important and the ones that could yield the greatest gain for an adversary. The problem becomes particularly acute in a distributed system composed of cooperating agents or processes (Birman) that participate in the planning workflow such as those in the Distributed Episodic Exploratory Planning (DEEP) architecture. In DEEP, the very architecture that is flexible enough to support robust planning sessions and increase survivability, is the same Achilles heel that could foster rogue agents that adversely impact plans by seemingly trusted participants. As will be shown, compromised DEEP agents could undermine basic plan components in several roles: as core planning agents or as support agents that adapt or critique plan elements.

1.2 Future C2 Requirements

To meet future challenges, the U.S. Air Force (USAF) is moving toward a model of continuous air operations not bound by the traditional 24-hour Air Tasking Order (ATO) cycle. Meeting these objectives will require a highly synchronized, distributed planning, replanning, and execution capability. As a potential way ahead, the USAF released an innovative paper entitled "C2 Enabling Concepts" depicting what a potential future C2 environment could be. Four key concepts emerged from this vision of a future Air Operations Center (AOC):

- Distributed/Reachback planning
- Redundant/Backup planning
- Continuous planning
- Flexible, scalable, tailorable C2

The USAF is transforming its systems methodically to meet these advanced warfighting concepts, and in doing so, may leave itself more vulnerable to compromise as it spreads out functionality across geographically distributed nodes in a network. As a basic research topic area, DEEP addresses many of these operational capabilities as a core technical competency, particularly distributed, continuous and adaptable planning; and also presents a token technical architecture to develop new concepts. We will leverage DEEP's science and technology roots and its prototype implementation to present important aspects of a typical military objective and how we can take the next steps to ensure successful operation in a contested cyberspace environment.

1.3 Mission Assurance

In the space domain, mission assurance encompasses everything from improving risk management, safety, and reliability to decreasing the cost of overall mission success (Office of Safety and Mission Assurance). With so much at stake in terms of finances, research, time, and lives, space mission success is paramount. This is no different for

military operations during the planning stages where trust in data, systems and the agents that act within those systems is vital, especially if agility is a primary system attribute.

For general military purposes, mission assurance is defined as the ability to achieve the most important objectives of a mission using any means necessary, including non-automated backup processes and manual procedures if required. As a minimum, system availability must be high while operating in a contested environment. In this context, a *contested environment* is defined as conducting operations on the network where an adversary's cyber tools have been deployed and are actively working to thwart standard operations using stealth, deception and/or manipulation.

If we look at the attacks on the Estonian web infrastructure (2007 cyberattacks on Estonia) and assume these attacks worked as the news portrayed them, Estonia may have maintained normal operations if they were able to quickly cordon off affected routers before attracting worldwide attention. If they had been able to revert to alternate routing tables, backup systems or even rely on fallback mechanisms on mirrored sites, they essentially would've been able to "fight through" the unprovoked attack.

The tenets of information assurance: confidentiality, integrity, authenticity, availability and non-repudiation, are prerequisites for achieving mission assurance; however they are not sufficient. The tenets of information assurance are defined as follows.

- Attribution – holding a user accountable for their actions
- Authentication – making sure only privileged users are accessing appropriate information
- Availability – ensuring information and services are available to users when they need it within specific time constraints
- Confidentiality – making sure information that is destined for an individual or group is only seen by those parties
- Integrity – information kept unmodified by unintended sources

In building a case for mission assurance, availability must also be defined as a function of prioritized mission tasks mapped to network capabilities so that degraded states can be specified and measured. Trust must be built on top of the remaining four tenets so that those contributing to mission success will be given increased responsibility and those detracting can be continually isolated. In addition, mission workflow should be formally specified as business processes so individual threads can be identified and their importance to mission objectives analyzed. Achieving mission assurance requires that we ask the following questions:

- What tasks can the operation do without?
- What are the minimum resources the operation needs to run?
- How does the operation handle graceful or violent degradation?
- What are the most important objectives in the operation?
- How does the operation handle uncertainty?
- How does the operation protect data?

- How does the operation handle intrusion at centralized nodes and at the outer edge of the network?

There are many challenges not only establishing some modicum of mission assurance techniques but also in just defining "it." In addition, there are nearly an infinite number of exploits and even self-imposed and overly restrictive security procedures that could undermine mission assurance (Defense Science Board Task Force, 5-6). However, those topics will not be addressed in this paper, and we will instead concentrate on defining the next level of criteria beyond information assurance and apply it to a distributed planning agent system.

1.4 Objectives of the Paper

The goal of this paper is to illustrate mission assurance concepts within a distributed application participating in simulated real world operations in a contested environment. The major focus will be an examination of a research prototype system under development as an in-house project at the Air Force Research Laboratory (AFRL) called the Distributed Episodic Exploratory Planning (DEEP). DEEP has sufficient depth as a robust agent infrastructure to explore multiple pressure points within its architecture as well as enough breadth to cover a variety of operational use cases. This paper presents concepts of how mission assurance is being applied within a well-scoped research project. The concepts, along with metrics for testing them will be presented as well as a generalized approach for applying mission assurance in distributed planning systems.

2 Distributed Episodic Exploratory Planning (DEEP)

The long-term goal of the DEEP project is to develop an in-house prototype system for distributed, mixed-initiative planning that improves decision-making for mission planners (DeStefano and Lachevet). It applies analogical reasoning over an experience base to support Network Centric Operations (NCO).

2.1 Objective of the DEEP Project

Alberts and Hayes (2007) advocate bold new approaches beyond current organizational process, focusing on what is possible for NCO. High priority basic research topics recommended as areas to systematically explore are:

1. Taxonomy for planning and plans;
2. Quality metrics for planning and plans;
3. Factors that influence planning quality;
4. Factors that influence plan quality;
5. Impact of planning and plan quality on operations;
6. Methods and tools for planning; and

7. Plan visualization

Pursuant to achieving DEEP's vision, essentially all the above topics needed to be represented in the program's two primary objectives:

- Provide a mixed-initiative planning environment where human expertise is captured, developed, adapted and integrated with machine-generated output to augment human intuition and creativity.
- Support distributed planners in multiple cooperating command centers to conduct distributed and collaborative planning.

The DEEP architecture was explicitly designed to support the tenets of NCO in a true distributed manner. Because DEEP is not based on any current C2 system, we are able to explore concepts such as combining planning and execution to support dynamic re-planning, machine-mediated self-synchronization of distributed planners, and experiment with the impact of trust in an NCO environment.

2.2 DEEP Overview

DEEP is a components-based architecture comprised of the following modules:

- Distributed Blackboard for multi-agent, non-deterministic, opportunistic reasoning
- Case-Based Reasoning system to capture experiences (successes and/or failures)
- Episodic Memory for powerful analogical reasoning
- Multi-Agent System for mixed initiative planning
- ARPI Core Plan Representation for human-to-machine common dialog
- Constructive Simulation for exploration of possible future states

The DEEP architecture includes a messaging system, various knowledge objects, a shared data storage system, and a number of agents. For convenience, we will describe the pieces in the architecture in the order in which they might be typically used. One should bear in mind, however, that in this type of mixed-initiative system, there will rarely be a clean path from the initial planning problem to the final solution.

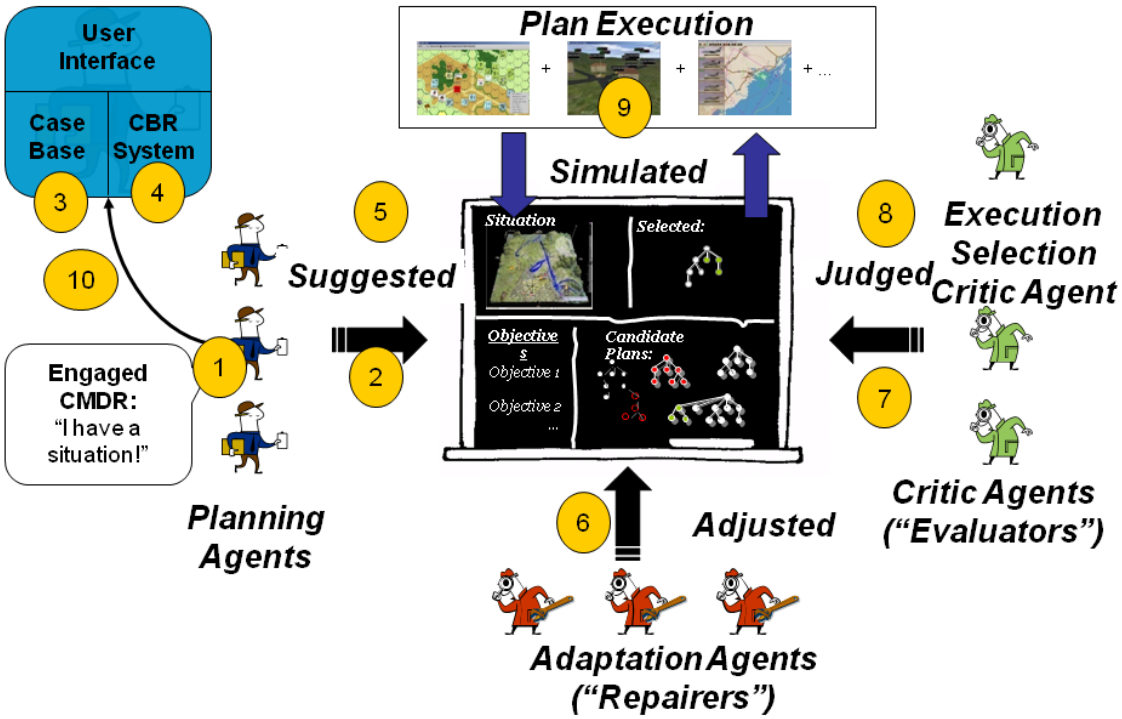

Figure 1 - DEEP Architecture

The starting point for entry into the architecture in Figure 1 occurs when a commander describes a new mission using a *planning agent* (1). The planning agent allows the commander to input information into the system which defines their *current objectives*. These objectives, along with other information, such as resources, locations, and time constraints, are collectively known as the *situation*. This situation is then placed on the shared blackboard (2). The blackboard would in turn notify all registered components of the existence of a new situation. Using the given situation, the other planning agents, with their associated case bases and cased-based reasoning capabilities, would each search their repositories for relevant past experiences (3). These results are then modified to fit the current situation (4) and are posted to the blackboard as *candidate plans* (5). Once the candidate plans are on the blackboard, they are adapted by specialized *adaptation agents* to further refine these plans to meet the current situation (6). These plans are now ready to be critiqued by the *critic agents*.

Critic agents concurrently scrutinize the candidate plans and score them based on their individual expertise (7). Once the plans are scored, the *execution selection critic* gathers the adapted plans along with their scores, determines their overall scores, and selects a number of top rated plans to be executed (8). The top rated plans are now executed (currently in a simulated environment) (9). Once a plan completes execution, the results are combined with the plan and assimilated back into the original planning agent's case base (10).

Although we have described this planning and execution as a single flow through the system, in reality, few plans will execute without changes. The DEEP architecture

supports the modification of currently executing plans through feedback of partial results of plan execution into the blackboard. This allows the plans to be run through the adaptation and critique processes as many times as needed.

2.2.1 Component Interaction

As a network-centric application, there is a strong emphasis in DEEP on how knowledge is passed and how that knowledge is encapsulated. Before discussing the major structural subsystems, we present a discussion on how the components interact and on the knowledge objects that the systems process. The next two sections discuss the main type of knowledge object used in DEEP and in the DEEP messaging system.

2.2.2 Common Plan Representation

The various DEEP components all use a common knowledge representation to facilitate their interactions. This creates a common understanding between and among various communities of interest, including those outside of the Air Force for both military and civilian agencies. Thus, DEEP was designed to support plans for joint, coalition, and civilian operations as well handle plans at different abstraction levels (i.e., strategic, tactical, or operational). Planning for heterogeneous operations also means that the plan representation has to be able to consider the semantics of terms used in the plan, ensuring agreement among all participants. Finally, because DEEP is a mixed-initiative environment, the chosen plan representation must be easily machine-readable as well as presentable to a user.

In DEEP, the common plan representation (CPR) is used to represent individual experiences, or *cases*, which are composed of a plan, events, and one or more outcomes (Pease). The attributes of the plan are used by the cased-based reasoning system to determine the similarity of past cases with the current situation. Execution (currently through simulation) of the plan populates the events and outcome sections (Ford).

2.2.3 System Messaging

CPR is the foundation for the DEEP architecture and used by all components, thus a formalized messaging model is required for the interactions within the systems. The systems that interact with one another include various types of agents along with the system blackboard. To accomplish this, a formalized messaging scheme based on inter-agent communication is required with a defined structure so that new systems are able to understand incoming messages as well as transmit their own.

In the current DEEP architecture, the communication protocol used is the publish-subscribe communication paradigm through interactions with the blackboard. At a high level, systems subscribe to the blackboard and are notified when new information is

added. Because of the push to create a functional proof-of-concept architecture, a simple taxonomy is currently in place to determine notification and message types until a more formalized communications protocol is established. The blackboard mediates all messages using its defined messaging scheme and connectivity medium. To prohibit the distributed planning aspect of DEEP deteriorating to "chat-room" type collaboration, an artificial barrier has been placed on human-to-human direct planning. Therefore, agents of any kind (human or software) do not communicate with each other directly, but instead use the blackboard as a hub of communication.

As an example, let us assume a simple setup of two planning agents (A and B), one blackboard, one critic agent, and one simulator. All systems are registered with the blackboard for communication. Planning Agent A becomes an *engaged* agent when its user inputs a new situation through the agent's user interface. Planning Agent A in turn places this new problem on the blackboard. Once posted, the blackboard notifies registered systems with a message indicating the type of object (e.g., new problem situation) by broadcasting these messages. The notified systems have to decide if the message is relevant to them. This happens for all communication, so when Planning Agent A posts a new situation, *all* registered systems are notified, including critics who cannot do anything with a situation.

2.3 Mission Assurance in DEEP

The following sections examine DEEP processes and component interaction against the established set of information assurance tenets as a first pass and then identify areas that could be enhanced within a mission assurance framework. Each of the primary components and their potential vulnerabilities are examined along with various resolution approaches.

2.3.1 DEEP Workflow

A key DEEP feature is a planning workflow model, which provides an understanding of operational paths, activity dependencies, and component interaction. The workflow model will allow users to identify a critical path through the system, defining mandatory and optional plan elements and system components. This process identifies mission critical functions and allows designers to allocate finite system resources. The workflow also defines the process by which state transitions occur in DEEP and have initially been created using Microsoft PowerPoint objects but could easily be translated into Unified Modeling Language (UML) variants.

DEEP's distributed case-based reasoning system and blackboard are foundational to other aspects of the system. All agents and simulations are dependent on having a working case retrieval system and a blackboard in which to collaborate; however, this does not preclude agents from utilizing external utilities and applications. Finally, agent processes

are also critical for adaptation and plan evaluation, case-based retrieval, and information sharing.

2.3.2 Agent Control Center

The Agent Control Center (ACC) is the software currently under development that controls automated agents and provides situational awareness to other DEEP components. The ACC would be responsible for monitoring and controlling all agents executing in the architecture. It would monitor agent traffic and look for odd behavior such as unknown software attempting communication with DEEP components and out-of-band network traffic being transported to unknown locations. The ACC would have the capability to ascertain the status of agents, identify network bandwidth limitations, manage centralized state, and dynamically shutdown and restart agents on different platforms. It could also conduct trend analyses and allow for human interaction for better control. It would use standard information assurance techniques and attempt to spot corrupted agents before damage is inflicted on plan elements. The ACC itself could operate in a contested environment, but it would also need appropriate authentication and access control.

2.3.3 DEEP Information

Corruption or manipulation of core system data can cause deceit among agents as a minimum or complete application stoppage in some situations. Problems associated with prolonged deceit can compound over time and has the potential to create mistrust among human and computer agents. Rogue agents in DEEP could modify readily available data with the intent to deceive using subtle or obvious techniques. An example could be to make minute changes to target locations so that strikes against them would be off course. These changes could also be accomplished after critic agent evaluations so the normal DEEP processing cycle would not detect them, forcing another means of assuring data integrity. Encryption and checksum techniques could be used to maintain data integrity from deliberate or inadvertent manipulation during transmission, but could also thwart reconnaissance from stealthy rootkits or other malware during local execution and data storage.

Even if the data is safe from prying eyes and potential manipulation, it is still not necessarily *protected*. The ACC will continually monitor agent network traffic and component execution with the intent on keeping core and control information leakage to a minimum. The ACC will authenticate, register and control access among DEEP participants and will not allow transmission outside that range.

2.3.4 DEEP Agents

By design, DEEP has built-in protective measures that could prevent certain types of unacceptable actions. An example is a corrupt planning agent inputting bogus plan

segments that would be challenged throughout the planning process by all the critique agents and simulation mechanisms. Another example is a plan segment retrieved from the case base by a corrupt agent will not be allowed to make an official plan change due to unacceptable scores by legitimate agents. Though corrupt agents may not be able to force their will on plan outcomes, there is a clear indication that the agent's case base is not an accurate representation of a legitimate agent and should be isolated. A solution to this is for the ACC to randomly conduct tests by running the scenario against a redundant case base to compare results. If drastically different or frequently erroneous results are returned for a planning agent, these could be clues that the planning agent is corrupt. The ACC can neutralize the corrupt agent and restart a healthy agent in its place on another node.

However, corrupt adaptation agents and planning agents can, with permission, make changes to the plan during runtime, essentially making all prior accomplishments worthless without persistent storage. DEEP designers have considered using a database or some other storage mechanism but would require considerable architectural changes. Even with persistent state, how does the ACC know the exact occurrence of a plan change from an uncorrupt to corrupt state?

2.3.5 Network

Network attacks, such as distributed denial of service attacks, would present serious problems for DEEP agent processing. Using redundancy for knowledge sources and data along with the ACC having the ability to dynamically reconfigure the network will allow previously unavailable processes to be made available. A user understanding the DEEP workflow will be able to take an alternative course of action when a known part of the workflow is not functioning.

2.3.6 Human Factors

DEEP relies on close human intervention during mixed-initiative planning as the human and machine collaborate together to create the plan. In the autonomous agent examples, there is a general concern for code intrusion, but for human interaction there is the issue of a misguided user or imposter accessing data or manipulating system processes. Using authentication, DEEP will allow only permitted users to operate and interact with the system. Authentication will also provide the ability to attribute actions to a particular user and users can have their actions mapped by the control center for trend analysis, which could help spot potential threats or oddities occurring in the system.

2.3.7 Trust

A complex research topic for DEEP is trust, because if trust can be quantified, then agent corruption can most likely be picked up quickly before damage is done. So how do we

implement trust? Agents could possibly be scored higher as they make rational and intentional decisions to provide something positive to the process - sort of similar to scoring a book seller on a website higher with good customer service. If an agent starts scoring lower or acts unpredictably, it can be restarted somewhere else with the current state to continue operating where it left off.

Another option for trust is using redundant agents in the system and, assuming they are deterministic, one could evaluate result differences using a voting system or a *wisdom of the crowds* approach. If an agent consistently goes against the norm of the group, it almost certainly has an issue if it should be giving the same answer in the deterministic system. If the particular agents are stochastic, a different approach will have to be used because the answers are expected to be different with each execution.

2.3.8 General Procedures

Following proper information assurance principles and operating on a secure foundation are essential for mission assurance. There are many variables to consider, such as following proper software engineering principles, operating on trusted and reliable software and hardware platforms, and installing scheduled and emergency system patches. Other practices include conducting exercises of red versus blue team attacks in a simulated operational environment and detecting malicious code tests using a suite of software vulnerability tools.

2.3.9 DEEP Experiment

At this stage of our in-house activity, DEEP leverages an autonomous agent architecture in place executing against several use cases; however, there is little to no information assurance encapsulated within the architecture. We plan to rapidly rectify this situation with a series of experiments over the next several months and will describe our initial test configuration and some questions experimentation will attempt to answer.

2.3.9.1 Rogue Agent Detection

Either the ACC or other agents must be able to detect intrusions or changes to the system caused by rogue agents. Under this test case, core DEEP processes would be running and a new agent would be randomly introduced to the system using several unique techniques. The DEEP system would have to detect the presence of this “rogue” agent and conduct corrective measures under the following circumstances:

- In one case the rogue agent would broadcast using a method dissimilar to the formal messaging used by DEEP in an out-of-band system call.
- A second case would have the agent gathering information and sending it out to unknown sources.

- A third case would be the agent modifies information on the blackboard with the full intent on corrupting an on-going planning process.

What is addressed in these cases is the presence of an adversarial agent that is looking to harm the planning process.

2.3.9.2 System Repair

System repair would begin after detection and data manipulation. Sample test cases are:

- Information passing through the system has been modified forcing a previous state to be retrieved.
- Knowledge structures (databases, files) have been altered and need to be repaired.
- How fast can DEEP dynamically re-shape its infrastructure while moving agents by taking them down and bringing them back up elsewhere?

2.3.9.3 Human Factors

Another area that bears analysis is dealing with the human in the loop as it pertains to agent interaction. In this case an agent with mixed-initiative ability would be connected to the system and the human would be presented with an agent they could log into and conduct operations. The human would have to authenticate into the system and be challenged by privileges as to what they sections of the plan they were allowed to access. This user should also be attributable for all actions occurring during the experiment.

2.3.9.4 Network Manipulation

DEEP has a lot of network data and information passing through its subcomponents, but the message traffic is completely open. As part of the experiment, network transmissions will be taken down randomly, causing a dynamic shift to occur for a test of that function. Another case will be to have the information relayed to a bogus node or have a legitimate node receive the data but have it be inaccessible for long enough to make it unusable. In the months leading up to the conference, the author will implement many of these ideas and will report initial findings.

All these details will be designed, developed, and tested before running the experiments. The system will be configured and executed in a contested environment to see how it responds. As each individual metric is refined, we will adhere to the following overarching themes:

- Can DEEP maintain core functionality?
- Can DEEP rollback state properly?

- Can the ACC conduct its role for dynamic agent transfer or does the ACC become too vulnerable itself and need more redundancy?
- Can DEEP repair itself from intentional corruption?

These are some of the questions that will have to be determined for success, but what about other quantifiable metrics?

2.3.9.5 Hypothesis / Metrics / Assessment

The hypothesis of the mission assurance work is whether DEEP will be able conduct normal systems operations in a contested environment replete with hostile and deceptive agents. There are many smaller variables to be tested under the mission assurance work, which includes:

- Must be able to transfer and restart agents on secure, trusted machines.
- Must be able to maintain agent state on move or restart. It may be that all state has to be lost so a specific metric is not enforced.
- Any transfer of components or replanning has to be faster than a full system restart.
- ACC must detect compromised agents.
- ACC must detect information theft.
- Components are attributable to their actions.

An assessment on the cost of an intrusion will be a measurement as well. One major question will be posed: what kind of an impact in mathematical complexity does an intrusion have depending on when it occurs?

Operation in a contested environment may not be the only metric, others include: (1) determining the quality of plan that can be produced with missing components or using human replacements (2) in preliminary assessment, developer testing will be conducted to determine if the plans are too similar to the raw cases from the case base, (3) similarity measurements can be taken to determine the quality of case base retrieval. Finally, future testing can be done using subject matter experts to measure the quality of the plan and further refine metrics.

3 Future Work

Follow on work to this paper includes fulfilling implementation of the concepts as well as testing them. Designing proper experiments and creating the optimal metrics is work that needs to be done.

Future work includes issues such as experimenting on other sections of the DEEP architecture, reviewing other planning techniques for their inherit robustness to cyber

attacks, integrating other AFRL technologies with DEEP, examining other threats, and multi-agent control.

4 Conclusion

The DOD and many industries are migrating to net-centric applications to conduct their core business strategies by putting "power to the edge," resulting in increased agility to meet ever-increasing requirements. Using a distributed approach to planning that pushes information into the hands of people and processes best suited for making problem-solving decisions is positive, but vulnerable by virtue of forcing participants to connect and reach back for virtually every individual operation. To conduct distributed coalition operations requires a reliable and trustworthy network and applications and services that can survive and thrive on the network. Mission assurance is needed to make all of this possible and is crucial to having distributed and agile C2. Coalition operations need to understand this and move to ensure all deployed systems are satisfactory in terms of information assurance and need to place emphasis on building systems and networks that are reliable, self-sustainable, and trustworthy.

Developing mission assurance capabilities is not an option as we move forward, but a requirement to operate in a contested landscape. DEEP is just one example of a research project planning system that could be vastly improved to counter adversity, but these concepts need further research and testing to broaden them out to a wide range of systems. We have shown how to apply an initial set of mission assurance principles to DEEP and the kinds of impacts a contested environment can have on DEEP and systems in general. This is important because it enumerates the possibilities and actions that can be taken against your systems so as to devise a proper defensive capability.

5 References

Alberts, D., & Hayes, E. (2007). *Planning: Complex Endeavors.* Command and Control Research Program.

Birman, Kenneth P. Reliable Distributed Systems. Springer: Springer Science and Business Media, Inc., 2005.

Braun, G., et al. (2006). *AFFOR Command and Control Enabling Concept-Change. 2.* AF/A5XS Internal Report.

Defense Science Board Task Force. "Mission Impact of Foreign Influence on DoD Software." CROSSTALK 21.5 (2008): 5-8. 24 Mar. 2009 <http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA487065&Location=U2&doc=GetTRDoc.pdf>.

DeStefano, C., Lachevet, K., & Carozzoni, J. (2008). Distributed Planning in a Mixed-Initiative Environment. *Proceedings of the 13th International Command and Control Research and Technology Symposium.*

Ford, A., & Carozzoni, J. (2007). Creating and Capturing Expertise in Mixed-Initiative Planning. *Proceedings of the 12th International Command and Control Research and Technology Symposium.*

Martin, Lorraine M., Ms. “Preparing for Conflict in Space: A New Perspective of the Joint Fight.” High Frontier 4.2 (2008): 21-23. 24 Mar. 2009 <http://www.afspc.af.mil/shared/media/document/AFD-080226-050.pdf>.

“Office of Safety and Mission Assurance.” NASA Office of Safety and Mission Assurance. 28 Jan. 2009. NASA. 24 Mar. 2009 <http://www.hq.nasa.gov/office/codeq/>.

Pease, R. Adam. “Core Plan Representation.” Version 4 November 6, 1998

Wilson, C. (2007). Network Centric Operations: Background and Oversight Issues for Congress. CRS Report for Congress.

“2007 cyberattacks on Estonia.” Wikipedia. 23 Mar. 2009. 24 Mar. 2009 <http://en.wikipedia.org/wiki/Cyberattacks_on_Estonia_2007>.