REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
28-12-2015	Final Report	28-Jul-2010 - 27-Jul-2014

4. TITLE AND SUBTITLE
Final Report: A Multi-scale Cognitive Approach to Intrusion Detection and Response

5a. CONTRACT NUMBER
W911NF-10-1-0352

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS
David Benjamin

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
Pace University
1 Pace Plaza

New York, NY 10038 -1598

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
57379-CS.1

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
The goal of this research is to create an architecture for multi-scale analysis of emergent behavior for network security. Our system will analyze network behaviors ranging from entire system behavior down to the packet level, treating treat attackers' behavior as a complex nonlinear behavioral system. The significance of this project is that it represents a completely new direction in intrusion detection research. Previous work has focused on analysis of individual alerts and sensor readings, rather than on analysis of the dynamics of global patterns of alerts and sensors. A major subgoal of this work is to evaluate data mining methods in cybersecurity. There is a large body of

15. SUBJECT TERMS
intrusion detection, cognition, learning

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			David Benjamin
UU	UU	UU	UU		19b. TELEPHONE NUMBER 212-346-1012

Report Title

Final Report: A Multi-scale Cognitive Approach to Intrusion Detection and Response

ABSTRACT

The goal of this research is to create an architecture for multi-scale analysis of emergent behavior for network security. Our system will analyze network behaviors ranging from entire system behavior down to the packet level, treating treat attackers' behavior as a complex nonlinear behavioral system. The significance of this project is that it represents a completely new direction in intrusion detection research. Previous work has focused on analysis of individual alerts and sensor readings, rather than on analysis of the dynamics of global patterns of alerts and sensors. A major subgoal of this work is to evaluate data mining methods in cybersecurity. There is a large body of published work, but little has been migrated into products. We need to find which methods work best and why the others fail.

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received Paper

TOTAL:

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received Paper

TOTAL:

Number of Papers published in non peer-reviewed journals:

(c) Presentations

Number of Presentations: 0.00

Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received Paper

TOTAL:

Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received Paper

TOTAL:

Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):

(d) Manuscripts

Received Paper

TOTAL:

Number of Manuscripts:

Books

Received Book

TOTAL:

Received	Book Chapter

TOTAL:

Patents Submitted

Patents Awarded

Awards

Graduate Students

NAME	PERCENT_SUPPORTED	Discipline
Hamid Sheikhghanbari	1.00	
Maryam Ansari	1.00	
Nikhil Fernandes	0.50	
Taranjyot Multani	1.00	
FTE Equivalent:	**3.50**	
Total Number:	**4**	

Names of Post Doctorates

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Faculty Supported

NAME	PERCENT_SUPPORTED	National Academy Member
David Benjamin	0.25	
FTE Equivalent:	**0.25**	
Total Number:	**1**	

Names of Under Graduate students supported

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period:	0.00
The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:.....	0.00
The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:.....	0.00
Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):.....	0.00
Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:.....	0.00
The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense	0.00
The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:	0.00

Names of Personnel receiving masters degrees

NAME	
Hamid Sheikhghanbari	
Maryam Ansari	
Nilhil Fernandes	
Taranjyot Multani	
Total Number:	**4**

Names of personnel receiving PHDs

NAME
Total Number:

Names of other research staff

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Sub Contractors (DD882)

Inventions (DD882)

Scientific Progress

Statement of the problem studied

Our overall approach is to investigate the application of analytical methods from the behavioral sciences to cybersecurity, especially advanced nonlinear methods. We view attackers' behavior together with the behavior of the computer network as a complex system; currently this dynamics is not well understood. Existing intrusion detection tools based on statistical analysis of traffic do not address this at all. Our system's central data structure is a hierarchical graph whose nodes are the beliefs, hypotheses and observations about behaviors in the computer network. The nodes at each level contain the information relevant to that level of the network; higher-level nodes in this graph are decomposed into subgraphs. The evolution of the structure of this graph over time describes the behavior of the network. The data gathered from the network sensors create a behavior space, which can be treated as a time series. Although this data can often appear to be random noise, there can exist hidden patterns that specify its dynamics. One of the main goals of this project is to apply and evaluate nonlinear analytical methods such as recurrence quantification analysis, fractal analysis and nonlinear dimensionality reduction to the evolution of this graph.

This type of dynamic analysis is necessary because the basic challenge is to identify patterns that are changing over time as the attackers' behavior changes. Furthermore, in real systems these patterns are buried in enormous amounts of noise; we must be able to recreate this situation in a controlled environment and search for relevant information.

This approach requires gathering multiple datasets on the behavior of large networks, then analyzing and mining this data for useful patterns. The success of this project depends on obtaining datasets that contain malicious activity from an intruder and datasets that contain no such activity, and finding recurrent patterns that distinguish them. A large number of datasets is required, with varying
network topologies and usage patterns.

We have created a number of different virtual networks, varying according to topology, mixture of OS platforms, and configuration of virtual servers. From each of these networks we have created a number of datasets using different mixtures of attacks. We have focused on user-to-root attacks. The results of our analysis are in the Scientific Accomplishments section.

One barrier that we face is testing our methods in a sufficiently realistic setting, consisting of a network of hundreds of nodes including mobile nodes, and testing in a manner that permits network behaviors to be reproduced for analysis and experimentation. In the past year, we have constructed an experimental design tool to explore nonlinear patterns in network behavior, and have performed initial experiments. The central obstacle to the completion of the work has been the lack of a sufficiently powerful computer on which to run large, realistic experiments. In the past year, we wrote and submitted a DURIP proposal for $110,000 to fund the purchase of such a computer, and that proposal was approved. Once we receive this machine, our main focus in the next year will be the mining of nonlinear patterns in the large datasets we can generate.

Another barrier is the lack of previous work with which to compare our approach. Nearly all work using this form of analysis is in the fields of medicine, physics, etc.

A barrier to our analysis of published data mining methods is that the publications do not reveal much detail about how their experiments were done.

A final barrier to our exploration of nonlinear data mining methods is their extreme computational cost. Analyzing even a small portion of our data using these methods can take hours of computation.

One goal of the project is to develop a relatively inexpensive testing platform to support cybersecurity research. Our own cybersecurity research includes work on cognitive cybersecurity agents [1], [2], [3], and on data mining in intrusion detection. We have found that our cybersecurity work has been impeded by the lack of certain resources. Our cognitive agents research has been difficult to evaluate without a good testbed that can be used to create a variety of attacks in a truly realistic setting, and our data mining research has been hampered by the lack of a large and diverse collection of network traffic datasets. We built RBG to support our research and we believe it can be equally useful to other researchers.

A number of simulators have been used to try to simulate networks and attacks [4], [5], [6], [7], [8], [9], [10], [11]. In this approach, networks, traffic and attacks are modeled at a high level of abstraction, usually using discrete event simulation.

A basic difficulty is that a simulator does not produce the whole range of behavior of a real network, but attempts to predict the behavior of the traffic on the network, either by using mathematical formulas or by replaying packet streams. As a result, simulators depend deeply on the assumptions made in programming the simulation, so their results are questionable. Such predictions can be very useful in evaluating network properties such as routing strategies and protocols, but cannot accurately predict how the behavior of a network might change when it is under attack.

In the context of the rapidly decreasing cost of virtualization, we have developed RBG to be a testbed for cybersecurity research that replicates real networks in detail, with virtual machines and virtual users that run real software and generate real traffic. RBG is a real network, just without the physical boxes for the machines and the physical bodies of the users. RBG generates real traffic and supports and measures the effects of real attacks. This provides the advantages of a real network without the associated costs, and can provide datasets from a variety of networks. DeterLab [13] is also evolving from a physical testbed into a more virtual testbed, but apparently without simulated users. Their focus is still more on providing a large set of resources that can be used in many different ways by cybersecurity researchers; our focus is on creating an inexpensive testbed that replicates the behaviors of real networks. We need to move beyond demonstration prototypes and model real systems and real attacks on a large scale.

Summary of the most important results

We obtained a powerful machine, which has 768 cores and 1.25 TB memory. RBG has been implemented on the machine, and runs test networks of 1000 virtual machines faster than real time. This has enabled us to analyze large realistic datasets. That analysis will be published.

The main accomplishment of these two grants was the construction of the RBG testbed. RBG is not a simulator; it creates real networks with virtual machines and virtual users. The software on the machines is real, the traffic is real and the attacks are real. The only things missing are the physical bodies of the machines and the users. RBG is not really comparable to network simulators.

The design of our testbed was guided by the following four goals:

Realism
Flexibility
Economy
Automated Control

The first goal, Realism, was the most important. The value of any testbed is directly dependent on how close its behavior is to that observed in real networks. This requires a testbed to create real traffic, not simulated packet streams, and to enable real attacks to be launched. The need for realism eliminated network simulators from consideration.

Flexibility is needed so that a wide range of experiments can be performed. Realism can be achieved by using a real network, but then flexibility is sacrificed. The need for flexibility led us to choose virtual networks.

Economy was important to us, and not just because we are a relatively small school with limited resources. Existing testbeds are large and expensive [14], which restricts their number and use. We wanted to create a testbed that was much less expensive and that has a small footprint, so that it could be widely adopted.

Finally, we wish to make our testbed relatively easy to use by automating as much of the control as possible. Currently, RBG supports automatic configuration and construction of the virtual network and automatic logging of network data, but setting up and launching attacks still requires some manual intervention.

Overview of the Structure of RBG

RBG consists of three main components:

- The control management system sets up and monitors the experiments. This is located in a reserved virtual machine in the network.

- The experimental platform contains the network infrastructure, including the virtual machines and virtual routers and the ability to generate traffic on them.

- The data collection system creates a database of the network traffic for future analysis.

In each experiment, RBG creates a network of virtual machines. Each client machine is equipped with a virtual user that runs applications to generate real traffic. RBG's attack generator is used to select one or more virtual machines to be attackers and to select which attacks to launch, whom they will attack and when they will be launched. A database records the network traffic together with information about which traffic is generated by attacker machines, for future analysis.

In designing RBG, we decided to keep our testbed initially separate from the Internet, to maximize the controllability of the environment. Although we intend eventually to permit experiments that are connected to the live Internet, we have not done so yet. Experiments requiring simulated access to the Internet are currently handled by constructing a small set of virtual machines that generate the type of Internet traffic to be modeled, e.g. a set of virtual machines implementing a shopping site like Amazon, which can be accessed by the client machines in the testbed network.

RBG's Virtual Network

The virtual network is the foundation of RBG. The first important choice we were faced with in designing RBG was selecting the software to support the virtual machines and network. This choice was very important, as it would constrain most of our other decisions. We had a number of choices, including Xen, VirtualBox, VMware, and other virtualization products. We wished to go open source, but encountered problems when we investigated the open source choices: primarily performance difficulties and also difficulty in setting up and controlling the virtual network. These same problems came up when we tried most of the

commercial alternatives. In the end we chose VMware because it was the fastest and its vCenter control software helps a great deal in creation and control of the network. Their academic site license was inexpensive and made our choice economically feasible. Overall, VMware provided the most satisfactory solution to our four design goals.

To produce realistic network behavior for a wide range of networks, RBG needs to provide support for a number of different hardware and software platforms, including mobile devices. To host RBG's virtual network, we use the VMware ESXi hypervisor. The benefits of using this bare metal hypervisor are apparent when we consider that the elimination of a resource intensive operating system is crucial given the fact that we need to host a large number of virtual nodes in the network. Our virtual networks contain user nodes, IDS nodes and server nodes. User nodes represent the devices that users operate to access network resources.

VMware ESXi has extensive virtual networking capabilities. Virtual machines can be connected logically to each other so that they can send and receive data almost identically to a real world network. To replicate the different types of network topologies, we used virtual switches (VSwitch) that are available in ESXi. This switch serves as a programmable patch panel to set up desired experimental topologies. The ESXi hypervisor allows for a total of 4096 ports, allocated via virtual switches per each host. The ESXi virtual switch supports copying packets to a mirror port. By using promiscuous mode, ESX Server makes a virtual switch port act as a SPAN port or mirror port. This makes it possible to debug using a sniffer or to run monitoring applications. The availability of SPAN ports is an essential part of our requirements for this project. ESXi provides a great deal of flexibility in the configuration of virtual switches.

Our virtual networks consist of both server and client nodes. The client nodes are lightweight nodes that run user applications. Currently, our clients run Linux, Windows 7 and Windows XP. It is straightforward to add more operating systems. Each is a complete distribution running in its own thread. Our clients run email, chat and browser applications including http and ftp.

A typical client node is lightweight, with two clients sharing a single processing core in real time. Each client is configured with 1GB memory, 10 GB disk space, and one 100M Ethernet interface.

The server nodes include web servers, database servers, email servers and chat servers. The servers can be Windows servers, e.g. Win 2003, or Linux servers. Each server contains actual data, e.g. a database, and serves the files to clients as requested. RBG can configure and connect server nodes as requested to fulfill user requests for cloud resources.

vSphere and vCenter provide the basic capabilities enabling virtual servers to be configured and deployed on command. vSphere is a very mature and efficient platform for managing a large and dynamically changing collection of virtual machines. When a client VM issues a request for a server with specified properties, vSphere configures the virtual server with the specified operating system image and returns its IP to the client. In this way, our virtual users can request cloud resources, run programs on them, and release them when completed.

Traffic Generation by Virtual Users

One of the key aspects of RBG is its ability to create fully realistic traffic. The traffic is not simulated in any way, but is generated by running real applications on virtual machines. The virtual machines include various operating system platforms, including Windows 7, Windows 8, and Ubuntu Linux. Each virtual machine is configured with the full set of application software typical for a machine of its type. For example, when we recreate an academic computing environment, the types of users include faculty, administrators and students. Each type of user machine will be configured with the appropriate software. This is achieved in a straightforward way by using VMware to clone a real machine of that type.

In order to obtain full realism in the generated traffic, we use a unified cognitive architecture [12] to implement each virtual user. This cognitive architecture (Soar) is a mature artificial intelligence application that uses goal-directed reasoning to guide its actions. Each virtual machine in RBG contains a copy of this cognitive architecture so that it can replicate the actions of a typical client user in a realistically human way.

The Soar cognitive architecture was originally developed at Carnegie-Mellon University in the early 1980s, and has been developed and expanded over the past thirty years. Soar is based on a theory of problem solving and learning that has been shown to reproduce human behavior on a wide range of tasks, including algorithm design and robot control, and has been used to model the behavior of military pilots [15] and the behavior of the NASA test director [16]. Soar is goal directed, with the ability to automatically create subgoals to accomplish tasks. Our Soar client agents possess goals that are typical of client users, e.g. reading and responding to email, querying a database, and requesting cloud resources to run a program. To accomplish these goals, the Soar agent uses the standard applications on the client machines, exactly as a human user would. The frequencies of the goals used by the Soar agent are based on normal user profiles extracted from real traffic on a Pace University network.

The applications on the client machines include sending email, querying a database, accessing a web server, accessing an FTP server, requesting cloud resources, executing a user program, and chatting with another client. For example, a Windows 7 virtual machine client would actually use Internet Explorer to download a webpage from a webserver. The cognitive architecture

itself is run by the testbed controller via a dedicated port on each virtual machine.

Different classes of users are implemented by different rule sets, so that they run different applications and in varying ways. The server virtual machines in RBG are actual servers, including webservers, database servers, chat servers and email servers. In this way, realistic scenarios of user behavior are created and controlled, and actual network traffic generated for analysis.

Running and Recording Attacks

The central goal of the testbed is to support experiments in which attacks are launched and recorded in a controlled way. In each experimental run, one or more machines are selected to serve as attackers. The RBG controller loads each attacking machine with the exploit software for the experiment and activates it at a selected time. RBG can launch a wide range of exploits, including the exploits in Metasploit, Ettercap, Ophcrack, BeEF and Hydra.

Each type of attack is implemented by an attack script that controls the attacking and attacked machines through a reserved port on each machine. The script coordinates the actions of the machines to cause the desired behavior. For example, for a phishing attack the attacked machine will request a page from the attacker's site at a predetermined time, then will open the malware attachment, installing the attacker's software. Then the attacking machine will connect and carry out the attack, e.g. copying files from the attacked machine. Each of these actions is triggered by the script at the appropriate time.
Scripts can generate multiple attacks of different types, to create a range of attack scenarios. Attacks of different levels of sophistication can be used to represent different classes of attackers, or to represent multiple attackers. The generality of the attack mechanism permits RBG to evaluate the effectiveness of defensive software installed on the network. It also enables the creation of datasets for mining.

During each experiment, network traffic data is captured and stored for analysis. Associated with every virtual switch in our virtual networks is an Intrusion Detection System node connected to the SPAN port on the switch. As all packet traffic flowing through a switch is mirrored to the SPAN port, it is the perfect location to host a sniffer that examines all traffic for suspicious activity. The IDS node captures and dumps all the packet data flowing through its virtual switch in real time and logs the data to a DBMS, which is hosted on its own virtual machine. RBG prevents attacks on the virtual machine that hosts the database, of course, because it is not part of the network being modeled.

Experiments are controlled by a top-level script that automates each test run. This permits a large number of runs to be controlled and analyzed. For example, to test the stability of an intrusion classifier that has been learned for user-to-root attacks, it may be applied to the data from a number of runs of the testbed, with each run varying one or more network properties and applying a variety of user-to-root attacks. The performance of the classifier is plotted against the properties to reveal how the classifier performs as the network changes, indicating whether the classification method is overfitting to the data.

The steps performed in each run of RBG are:

1) Generate a graph with a topology whose parameters match a given distribution
2) Generate a distribution of types of platforms
3) Instantiate virtual machines for each of the nodes in the graph, obeying the distribution of platform types
4) Instantiate virtual machines for each of the nodes in the graph, obeying the distribution of platform types
5) Generate a distribution of types of users
6) Instantiate these users on the virtual machines
7) Select one or more attack types and times
8) Run the virtual network
9) Launch the attacks

Conclusion

Our goal in this research was to build a tool that could provide a realistic recreation of a modern network, with a large number of nodes and providing a range of services. We could then use this environment to help us develop and evaluate cybersecurity algorithms.

RBG achieved this goal. It provides cybersecurity researchers with a powerful and relatively inexpensive testbed that can be used to test security software, generate datasets, and explore the effects of network parameters on security by generating realistic network behavior. RBG provides researchers the ability to create virtual networks of varying sizes and topologies, to populate those networks with machines with different operating systems, to control those machines with simulated users of varying types, and to launch a wide variety of attacks.

We have demonstrated how RBG can be used in data mining research to generate custom-tailored datasets to investigate

properties of attacks and classifiers. In the example we presented, we created a number of distinct virtual networks and generated 260 classifiers by using a range of classification algorithms. We were able to recreate published results and explore variations on them.

Our future work includes using RBG to test the effectiveness of more advanced features in data mining, especially features that depend on the content of packets, and features that measure changes in the attack landscape.

Bibliography

[1] Rubel, P., Pal, P., Atigetchi, M., Benjamin, D. P., and Webber, F., “Anomaly and Specification Based Cognitive Approach for Mission-Level Detection and Response”, Recent Advances in Intrusion Detection, Lecture Notes in Comp. Sci., Vol. 5230/2008, pp. 408-409.
[2] Benjamin, D. P., "A Cognitive Approach to Intrusion Detection", Proceedings of the IEEE Conference on Computational Intelligence for Security and Defense Applications 2007 (CISDA2007), Honolulu, Hawaii, April 2007.
[3] Benjamin, D. P., R. Shankar-Iyer, and A. Perumal, "VMSoar: A Cognitive Agent for Network Security", Proceedings of the SPIE Defense Symposium on Data Mining, Intrusion Detection, Information Assurance and Data Networks Security, Orlando, Florida, March-April 2005.
[4] Kuhl, M. E., et al., “Cyber Attack Modeling and Simulation for Network Security Analysis”, Proceedings of the 2007 Winter Simulation Conference, S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds., 2007.
[5] Costantini, K. C., “Development of a Cyber Attack Simulator for Network Modeling and Cyber Security Analysis”, masters thesis, Rochester Institute of Technology, 2007.
[6] Varga, A., and Hornig, R., An Overview of the OMNeT++ Simulation Environment, , 2008.
[7] Sarraute, C., Miranda, F., Orlicki, J., Simulation of Computer Network Attacks, Argentine Symposium on Computing Technology, 2007.
[8] Pullen, J. M., 2000. The network workbench: network simulation software for academic investigation of Internet concepts. Comput. Netw. 32, 3 (March 2000), 365-378. DOI=10.1016/S1389-1286(99)00133-4 http://dx.doi.org/10.1016/S1389-1286(99)00133-4
[9] Emulab. http://www.emulab.net.
[10] Vahdat, A., Yocum, K., Walsh, K., Mahadevan, P., Kostic, D., Chase, J., and Becker, D. Scalability and accuracy in a large-scale network emulator. In OSDI (2002).
[11] Ehrensberger, J.; Vernez, J.; Robert, S., "Nessi: a python network simulator for fast protocol development," Computer-Aided Modeling, Analysis and Design of Communication Links and Networks, 2006 11th International Workshop on , vol., no., pp.67,71, 0-0 0, doi: 10.1109/CAMAD.2006.1649720
[12] Laird, J.E., Newell, A. and Rosenbloom, P.S., “Soar: An Architecture for General Intelligence”, Artificial Intelligence 33, pp. 1-64, (1987).
[13] Benzel, T., “The Science of Cyber Security Experimentation: The DETER Project”, Proceedings of the 27th Annual Computer Security Applications Conference, pp. 137-148, 2011.
[14] Edgar, T., Carroll, T., and Manz, D., “Challenges of Cybersecurity Research in a Multi-User Cyber- Physical Testbed”, NIST Cybersecurity in Cyber-Physical Systems Workshop, April 23 – 24, 2012. http://csrc.nist.gov/news_events/cps-workshop/cps-workshop_abstract-5_pnnl.pdf
[15] Jones, R. Tambe, M., Laird, J., Rosenbloom, P. (1993). Intelligent automated agents for flight training simulators. In Proceedings of the Third Conference on Computer Generated Forces and Behavioral Representation. University of Central Florida.
[16] Printz, H., “An Analysis of Space Shuttle Countdown Activities: Preliminaries to a Computational Model of the NASA Test Director”, Ph.D. Thesis, CMU-CS-91-138, May 1991.

Technology Transfer