SYM-AM-18-090

PROCEEDINGS
OF THE
FIFTEENTH ANNUAL ACQUISITION RESEARCH SYMPOSIUM

THURSDAY SESSIONS
VOLUME II

Acquisition Research:
Creating Synergy for Informed Change

May 9–10, 2018

March 30, 2018

Prepared for the Naval Postgraduate School, Monterey, CA 93943.

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL

Applying Cause-Effect Mapping to Assess Cybersecurity Vulnerabilities in Model-Centric Acquisition Program Environments

Jack Reid—is a graduate student with the Systems Engineering Advancement Research Initiative (SEAri) at the Massachusetts Institute of Technology. Reid is earning master's degrees in both Aeronautics & Astronautics and Technology & Policy. His research interests concern the design and management of complex sociotechnical systems, particularly with regard to the anticipation of emergent and cascading behavior. He received a BS in Mechanical Engineering and a BA in Philosophy from Texas A&M University and has experience with RAND Corporation and Sandia National Laboratories. [jackreid@mit.edu]

Donna H. Rhodes—is a principal research scientist at the Massachusetts Institute of Technology, and director of the Systems Engineering Advancement Research Initiative (SEAri). Dr. Rhodes conducts research on innovative approaches and methods for architecting complex systems and enterprises, designing for uncertain futures, and human-model interaction. Previously, she held senior management positions at IBM, Lockheed Martin, and Lucent. Dr. Rhodes is a Past President and Fellow of the International Council on Systems Engineering (INCOSE), and INCOSE Founders Award recipient. She received her PhD in Systems Science from T. J. Watson School of Engineering at Binghamton University. [rhodes@mit.edu]

Abstract

Digital engineering approaches are increasingly used in acquisition of systems, changing the current paradigm from documentation-centric to model-centric. Not only are these systems highly vulnerable to cyber threats, so too are their enabling environment and digital assets. While good practices have emerged to support the shift to model-centric program acquisition, such programs experience perturbations over their life cycles that introduce new vulnerabilities that may lead to cascading failures. Cybersecurity vulnerabilities are of particular concern given digital transformation and increasing threat actors, making vulnerability assessment essential throughout acquisition program life cycles. This paper discusses ongoing research that seeks to provide program managers with the means to identify cybersecurity vulnerabilities within model-centric programs (along with other model-related vulnerabilities) and determine where interventions can most effectively be taken. The research builds on recent work in developing a reference model for model-centric program vulnerability assessment that uses the Cause-Effect Mapping (CEM) analytic technique. This research investigates cybersecurity specifically, using CEM and other dynamic analysis approaches, including a prototype for proactive assessment of cybersecurity and evaluation of potential interventions.

Introduction

Digital transformation changes how systems are acquired and developed through the use of model-centric engineering practices and toolsets. While offering great benefit, new challenges arise from both technological and socio-cultural dimensions. This drives the need to examine and address vulnerabilities not only for products and systems, but also for the model-centric environments necessary for their acquisition and development. Recent research has investigated the use of Cause-Effect Mapping (CEM) as a mechanism for better enabling program managers and system engineers to anticipate and respond to programmatic vulnerabilities as related to model-centric environments. A Reference CEM for model-centric enterprises resulting from the work shows promise for considering the cascading vulnerabilities and potential intervention options. In ongoing research, additional investigation aims to refine the Reference CEM and analytic approach for cybersecurity-focused program vulnerability assessment.

Motivation

Modern society has many needs and problems that can only be addressed through large-scale socio-technical engineering programs (e.g., defense systems, multi-modal transportation systems, energy delivery system of systems, health-care management systems). The use of model-centric approaches, modeling and simulation, and “digital twins” is increasingly used in acquisition of such systems, changing the current paradigm from documentation-centric to model-centric. Not only are these systems highly vulnerable to cyber threats, so too are their enabling environment and digital assets. While good practices have emerged to support the shift to model-centric program acquisition, such programs experience perturbations over their life cycles that introduce new vulnerabilities that may lead to cascading failures. For instance, perturbations may be caused by policy change (leading to IP disagreements), economic factors (leading to training cuts), or disruptive technology (leading to outdated infrastructure). Early detection and intervention of vulnerabilities can mitigate disruptions and failures. The research seeks to contribute to the vulnerability assessment state of practice for acquisition programs, both public and private, that increasingly depend on digital assets and model-centric environments.

Background

The following subsections describe model-centric engineering, cyber-security vulnerability assessment, and cause-effect mapping. A companion paper (Reid & Rhodes, 2018) provides additional background information.

Model-Centric Engineering

Acquisition program management is grounded in management science and a sound set of practices evolved over decades; however, new challenges arise as acquisition becomes increasingly model-centric. Baldwin and Lucero (2016) state, “The DoD sees value in adopting digital engineering design and model-centric practices, enabling a shift from the linear, document centric acquisition and engineering process toward a dynamic digital, model-centric ecosystem.”

Model-Centric Engineering (MCE) has been defined as “an overarching digital engineering approach that integrates different model types with simulations, surrogates, systems and components at different levels of abstraction and fidelity across disciplines throughout the lifecycle” (Blackburn et al., 2017). MCE involves using integrated models across disciplines, subsystems, life-cycle stages, and analyst groups. It uses models as the “source of truth” to reduce document handoff and allow for more continuous evaluation. This reduces communication time and rework in response to requirement changes. Most discussions of MCE focus on engineering practices and methods to overcome implementation difficulties. In any system, however, engineering is only a piece of the problem. Numerous human factors, business, and organizational issues exist. Current program managers have significant experience with modern engineering processes and use this experience to identify and mitigate vulnerabilities. No such experience exists with MCE, however. This fact, coupled with the increased integration of models, means that emergent uncertainties (policy change, budget cuts, disruptive technologies, threats, changing demographics, etc.) and related programmatic decisions (e.g., staff cuts, reduced training hours) may lead to cascading vulnerabilities within MCE programs, potentially jeopardizing program success. New tools are needed to enable program managers to identify model-centric program vulnerabilities and determine where interventions can most effectively be taken.

Cybersecurity Vulnerability Assessment

MCE, with its focus on digitization, integration, and collaboration, has the potential to increase the cybersecurity vulnerability of an enterprise. A *vulnerability* is the means by which the hazard might disrupt the system, thus it is through the vulnerability that the system is susceptible to the hazard. Vulnerabilities are best expressed as the causal series of events connecting a hazard to system failure. This is a generalization of common, field-specific usages of the term. MITRE's Common Vulnerabilities and Exposures (CVE) database defines a vulnerability as "a weakness in the computational logic (e.g., code) found in software and some hardware components (e.g., firmware) that, when exploited, results in a negative impact to confidentiality, integrity, OR availability" (The MITRE Corporation, 2015). In this definition, the same components can be seen: some structural means or "weakness" that can result in system disruption or "negative impact" if a hazard is present or the vulnerability is "exploited." For example, the infamous Spectre security vulnerability is described by CVE as "systems with microprocessors utilizing speculative execution and branch prediction may allow unauthorized disclosure of information to an attacker with local user access via a side-channel analysis" (The MITRE Corporation, 2017). This is a neat summary of the hazard (an attacker), the means (side-channel analysis using speculative execution and branch prediction), and the disruption (unauthorized disclosure of information).

Risk and vulnerability assessment methods have not failed to adapt to novel cybersecurity concerns. The aforementioned CVE database has been public since 1999. Quality assurance testing (essentially the verification and validation of software) has been around since the beginning of commercial software. Software penetration testing (where security experts intentionally seek to break a software product) has been the industry norm for more than a decade (Arkin, Stender, & McGraw, 2005). Black-box mutational fuzzing and concolic execution are being used to automatically test for certain types of software vulnerabilities (Schwarz, 2018). Formal verification tools, initially limited to pure software domains such as cryptography (Meadows, 1994), has been rapidly advancing and finding applications in hardware (Kern & Greenstreet, 1999) and business processes (Morimoto, 2008), as well as fields that straddle the software-hardware-environment boundaries (Kamali et al., 2016). The methods listed here just scratch the surface of approaches security researchers and engineers are taking to identify and resolve such technical cybersecurity vulnerabilities.

Beyond these specific testing methods, assessment frameworks have progressed as well. System-Theoretic Process Analysis (STPA) has adjusted, adapted, and been applied to handle cybersecurity vulnerabilities associated with additive manufacturing (Pope & Yampolskiy, 2016), Internet of Things (Pope, 2017), Air Operations (Young, 2013), and Mission Operations (Young & Porada, 2017). More recently, there have also been efforts to combine compiler technology with STPA to automatically detect vulnerabilities in software-controlled systems (Pope, 2018).

While cybersecurity vulnerabilities in operational systems remain alarmingly common, from the trivial (Hanselman, 2012) to the critical (Gressin, 2017), there is some evidence that software is becoming more secure, at least in terms of defects per line of equivalent source code (Pope, 2017). In many cases, however, the acquisition or development process itself needs to be protected from outside threats and endogenous failures. Be it military information or technology-related trade secrets, there is real value in attempting to penetrate much earlier in the life cycle in order to either steal secrets (Hanna, Smythe, & Martin, 2018; Raymond, 2017) or to disrupt production (Statt, 2018).

Defense acquisition programs have already instituted a variety of means of ensuring the security of their work. Some of these means were originally instituted to address other forms of threats but have turned out to be effective in addressing cybersecurity as well. These methods include relying on the security clearance process, the use of Sensitive Compartmented Information Facilities (SCIFs), restrictions on the use of media storage devices, separate networks such as SIPRNet and NIPRNet that are isolated or semi-isolated from the internet, and general compartmentalization of critical information. Some (non-U.S.) defense agencies have gone so far as to revert to using typewriters where able in order to avoid security breaches and leaks (Irvine & Parfitt, 2013).

Unfortunately, many of these historically successful methods are in conflict with the more straightforward implementations of many components of an MCE environment. For example, the use of SCIFs has been quite successful in preventing unauthorized access to data. The typical use of a SCIF in the design process, where a small number of engineers work on a task isolated from the outside world, is not directly compatible with an MCE environment structured around model integration and collaboration across teams and locations. While this problem has been previously considered and ways to mitigate this conflict have been proposed (e.g., Reid & Rhodes, 2016), no silver bullet to resolving these tensions exists and it is likely that the increased use of MCE will result in both the exacerbation of current vulnerabilities and the creation of new ones. Furthermore, most means of assessing such vulnerabilities are aimed at assisting software and systems engineers to identify and remove cybersecurity vulnerabilities from the end system. New methods for enabling project and program managers to perform cybersecurity assessments of their enterprise and engineering environment are needed.

Cause-Effect Mapping

Cause-Effect Mapping is a vulnerability assessment tool that consists of a mapping of causal chains that connect an exogenous hazard to a system degradation or failure, termed a *terminal event*. Each chain represents a specific vulnerability, sometimes called a *vulnerability chain* in order to emphasize that vulnerabilities are not discrete events. Terminal events are broadly defined and include any form of value loss. Interventions are actions that eliminate or mitigate a vulnerability, and are represented on the map as points that break the causal chain. An example CEM (that lacks interventions) can be seen in Figure 1.

The hazards are external to the perspective of the defined user, and are thus sometimes called *external triggers*. An *intermediary event* is any unintended state change of a system’s form or operations that could jeopardize value delivery of the program.

A CEM is not created for a system, but for a specific class of decision-maker. The hazards (referred to as “spontaneous events” in Figure 1) are exogenous from the point of view of the decision-maker that the CEM was made for. In this way, CEM avoids the “blaming someone else” problem by making all hazards exogenous. The decision-maker only has control over the intermediary events. While she may not be at fault for any of the vulnerabilities, it is still her responsibility to address them.

CEM is fundamentally a qualitative analysis method, though it can be readily adapted into a quantitative form by adding probabilities of transition to each intermediary. CEM provides immediate insight into which parts of the system warrant more detailed modeling using other methods.

The basic steps to create a new CEM are not application specific and are as follows:

1. The stakeholder herself lists potential hazards posed to the program.
2. She then traces the consequences of each of these hazards through the intermediary events to the final terminal events.
3. The process is then done in reverse: She looks at the terminal events, adds in any that are still missing, and works backwards on how they might come about.
4. She then examines the causal connections between each intermediary event to see if there are any additional connections not previously noticed.
5. Finally, she consults lessons learned databases, case studies, and other experts to generate additional hazards, intermediary events, causal connections, and interventions, as well as to verify existing ones.

Any of these steps can take place either formally, using automated tools to enumerate possible vulnerabilities, or informally, relying upon the stakeholder's own experience.

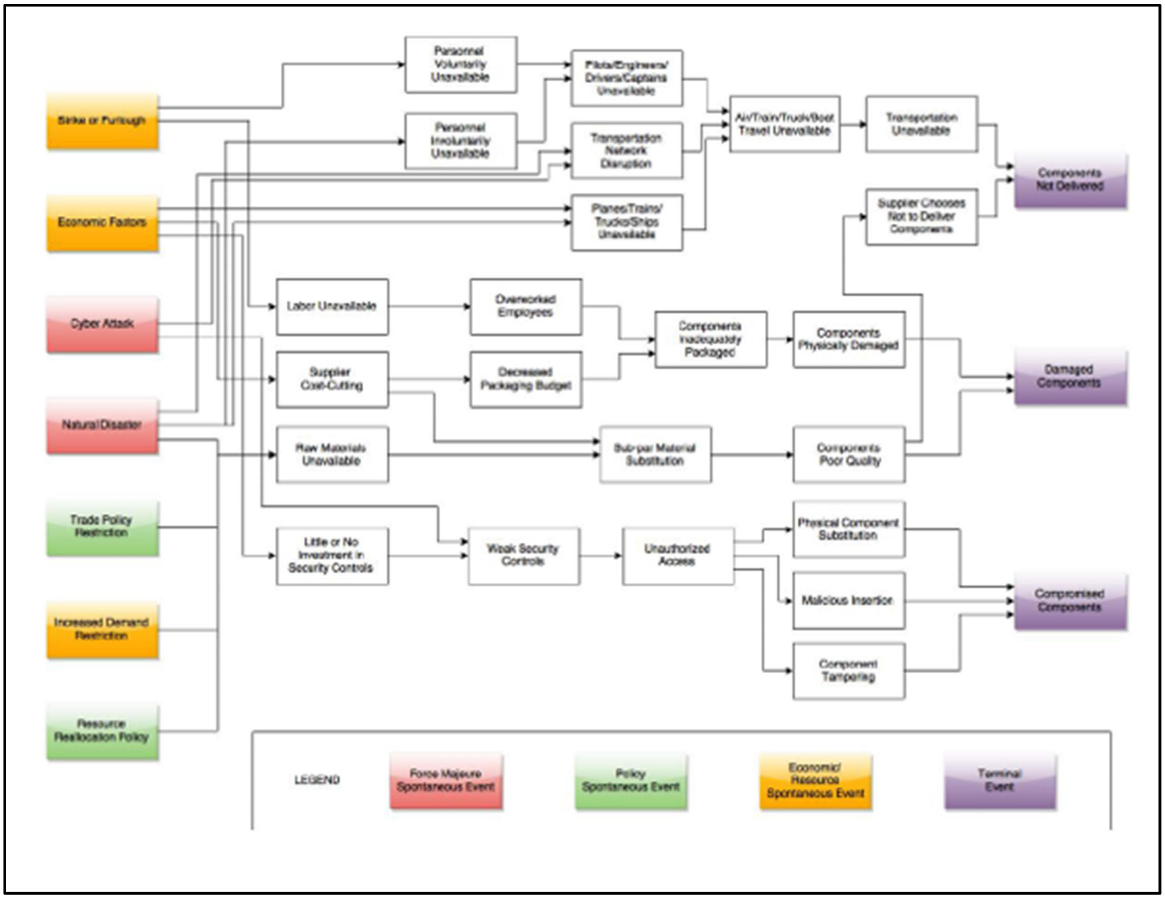

Strike or Furlough
Economic Factors
Cyber Attack
Natural Disaster
Trade Policy Restriction
Increased Demand Restriction
Resource Reallocation Policy
Personnel Voluntarily Unavailable
Personnel Involuntarily Unavailable
Pilots/Engineers/ Drivers/Captains Unavailable
Transportation Network Disruption
Planes/Trains/ Trucks/Ships Unavailable
Air/Train/Truck/Boat Travel Unavailable
Transportation Unavailable
Supplier Chooses Not to Deliver Components
Components Not Delivered
Labor Unavailable
Overworked Employees
Supplier Cost-Cutting
Decreased Packaging Budget
Components Inadequately Packaged
Components Physically Damaged
Damaged Components
Raw Materials Unavailable
Sub-par Material Substitution
Components Poor Quality
Little or No Investment in Security Controls
Weak Security Controls
Unauthorized Access
Physical Component Substitution
Malicious Insertion
Component Tampering
Compromised Components
LEGEND
Force Majeure Spontaneous Event
Policy Spontaneous Event
Economic/ Resource Spontaneous Event
Terminal Event

Figure 1. Example CEM of a supply chain
(Rovito & Rhodes, 2016)

CEM has previously been applied in a case study of a Maritime Security System of Systems (Mekdeci et al., 2012) and to a supply chain case (Rovito & Rhodes, 2016). More recently, an earlier phase of this research developed a Reference CEM for use by program managers to assess enterprise-level vulnerabilities in the MCE environment (Reid &

Rhodes, 2018). This work, which was based upon literature reviews, interviews with experts, and other sources, sought to provide program managers with an entry point for considering such vulnerabilities. Additionally, the steps to create a CEM for one's own program were outlined and some potential use cases discussed. These use cases are as follows:

(A) By a Program Manager: Assessing potential future vulnerabilities and plan possible interventions

(B) By a Program Manager: Determining specific vulnerabilities to address in response to the presence of a specific hazard

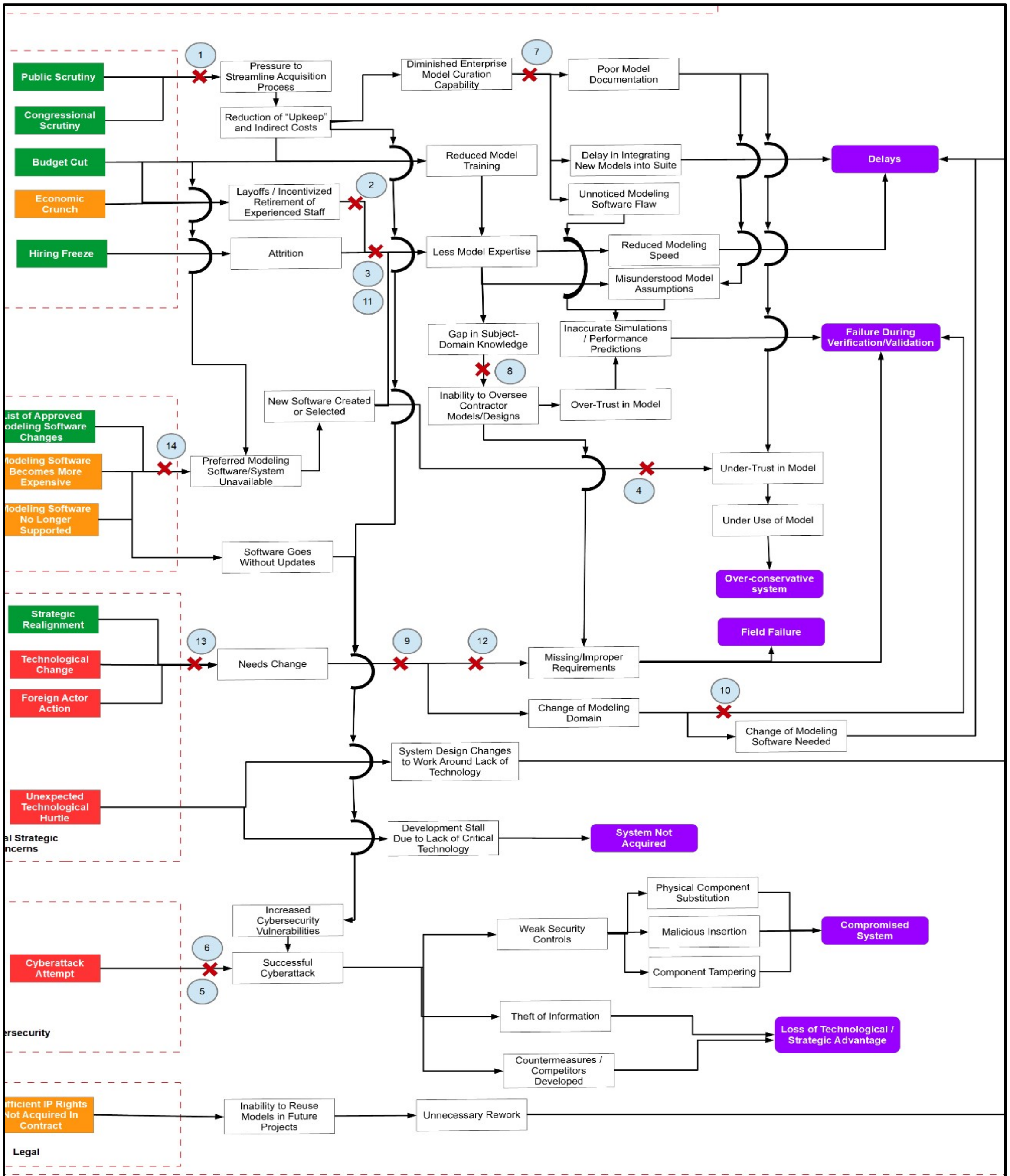
(C) By the Program Organization: Changing program processes to mitigate or eliminate vulnerabilities

(D) By Researchers: Organizing and classifying vulnerabilities into various categories or types

Most users of CEM tend to find it most useful in identifying high priority intervention points and new vulnerabilities. Other benefits of note include increased understanding of the causal path and the interrelationships between vulnerabilities. While the resultant reference CEM was quite detailed in some respects, such as both vulnerabilities and interventions involving model curation, it was less well developed in others, notably cybersecurity, as can be seen in Figure 2.

Use (A) is most relevant for novice program managers or program managers using MCE for the first time. A senior program manager or team of program managers creates a CEM for their organization's program process. This CEM can then be provided to the novice for study and reference. The program manager can then learn what can go wrong and how to intervene. In this case, the CEM could be tied to a Lesson's Learned database, such as NASA's Lessons Learned Information System (NASA Office of the Chief Engineer, 1994). This enables concrete examples and consequences to be linked to each vulnerability. One of the important factors here is that the CEM does not just present potential interventions, but it also places them in the appropriate part of the causal sequence. This enables the program manager to not only know how to intervene, but at what point.

In Use (D), CEM is used to organize and classify vulnerability chains. Two obvious classifiers are terminal events and hazards. Which is used to organize a CEM depends on whether the user wants to examine the causal chains forward or backwards. Beyond this, however, more complicated classifiers are possible. As can be seen in Figure 2, external triggers that result in similar vulnerability chains are grouped together. By "similar," we mean that these vulnerability chains either involve many of the same intermediary events or that they involve the same part of the program. For instance, most of the intermediary events involving model curation and trust are located close to one another in the center-top of the figure. Once these groupings have been identified, they can be considered together, such as the "Belt-tightening" grouping, and common means of intervention considered.


Public Scrutiny
Congressional Scrutiny
Budget Cut
Economic Crunch
Hiring Freeze
1
Pressure to Streamline Acquisition Process
Reduction of "Upkeep" and Indirect Costs
Diminished Enterprise Model Curation Capability
7
Poor Model Documentation
Reduced Model Training
Delay in Integrating New Models into Suite
Delays
Layoffs / Incentivized Retirement of Experienced Staff
2
Unnoticed Modeling Software Flaw
Attrition
3
11
Less Model Expertise
Reduced Modeling Speed
Misunderstood Model Assumptions
Gap in Subject-Domain Knowledge
8
Inaccurate Simulations / Performance Predictions
Failure During Verification/Validation
New Software Created or Selected
Inability to Oversee Contractor Models/Designs
Over-Trust in Model
ist of Approved odeling Software Changes
14
odeling Software Becomes More Expensive
Preferred Modeling Software/System Unavailable
4
Under-Trust in Model
Under Use of Model
odeling Software No Longer Supported
Software Goes Without Updates
Over-conservative system
Strategic Realignment
Field Failure
13
9
12
Technological Change
Needs Change
Missing/Improper Requirements
Foreign Actor Action
10
Change of Modeling Domain
Change of Modeling Software Needed
System Design Changes to Work Around Lack of Technology
Unexpected Technological Hurtle
al Strategic ncerns
Development Stall Due to Lack of Critical Technology
System Not Acquired
Physical Component Substitution
Increased Cybersecurity Vulnerabilities
Weak Security Controls
Malicious Insertion
Compromised System
6
Cyberattack Attempt
Successful Cyberattack
Component Tampering
5
Theft of Information
rsecurity
Loss of Technological / Strategic Advantage
Countermeasures / Competitors Developed
fficient IP Rights Not Acquired In Contract
Inability to Reuse Models in Future Projects
Unnecessary Rework
Legal

Intervention Points

e Internal Assessment and a PR Strategy
e various non-monetary benefits (such as allowing pets at work
0 schedule) to encourage employees to stay
to share resources and employees with other programs
mployees with prior experience with the new software
ly conduct security training and monitor compliance
e critical data
e documentation and curation processes within the program

8) Institute handover periods to benefit from contractor expertise
9) Reevaluate training regime and needed fields of expertise
10) Increase the amount of testing conducted
11) Increase use of contractors/consultants (including previous employees) to maintain expertise level
12) Reevaluate requirements with the client and other stakeholders
13) Design for modularity to minimize impact on system
14) Negotiate with client to see if they are able to pay for the software

Figure 2. Preliminary Reference CEM for Model-Centric Vulnerabilities With Example Intervention Points

Strengthening Cybersecurity Aspects of a CEM for Model-Centric Programs

As was discussed in the previous section, the MCE Reference CEM shown in Figure 2 was generated using literature reviews and interviews with experts, among other sources. The cybersecurity portion of it was adapted, mostly unchanged, from previous work on supply chains (Rovito & Rhodes, 2016). Cybersecurity is a rising international concern and is of particular relevance with the increasing digitization associated with MCE environments. As a result, further development of that portion of the Reference CEM was desired.

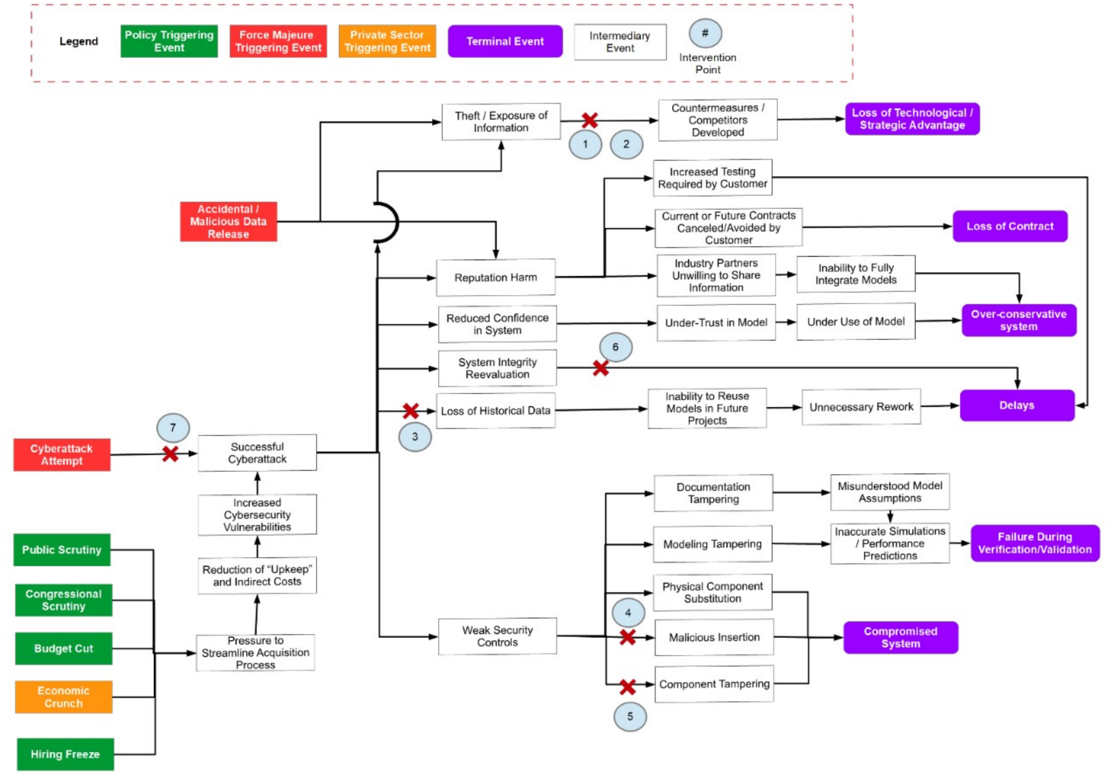
To accomplish this, an ongoing series of interviews with systems engineers and program managers from a variety of fields, including defense, aerospace, manufacturing, and semiconductors, is being conducted. These interviews have sought to provide insight into the following questions, in the context of a model-centric enterprise:

1. To what extent are program managers aware of programmatic vulnerabilities?
2. How do program managers conceptualize these vulnerabilities?
3. How do program managers respond to these vulnerabilities?
4. What vulnerabilities are present in MCE programs?
5. What cybersecurity vulnerabilities does MCE pose?

The first four questions were the primary focus of that previous phase of research. In this phase of the research, the focus is on the fifth question as a means of expanding the cybersecurity component of the Reference CEM shown in Figure 2. When it came to the topic of cybersecurity vulnerabilities in general, the interviewees commonly raised the following issues:

- Cybersecurity needs to be thoroughly considered much earlier than it commonly is, preferably in the proposal generation stage.
- Program managers and systems engineers are sometimes intimidated by cybersecurity issues and thus seek to pass them onto specialists later in the acquisition process.
- MBSE and MCE toolset developers and proponents have not done a thorough enough job of considering programmatic cybersecurity vulnerabilities, though the tools are typically quite effective at designing for cybersecurity in end systems.
- Despite all of the above, according to the interviewees, traditional programmatic cybersecurity defensive practices tends to quite effective. This is due primarily to the conservative approaches most defense-related engineering groups use, as discussed in the Cybersecurity Vulnerability Assessment section. The increased use of MCE, particularly for multi-site collaboration, could change this.

The above points, many of which were commonly stated by the same expert, are clearly nuanced and complicated, with both points of success and failure. These points, along with more specific comments from the interviewees, resulted in an expanded cybersecurity CEM that can be seen in Figure 3. Note that in its full form, this would still be a part of the general Reference CEM shown in Figure 2. Here it is shown isolated for clarity.


Legend
Policy Triggering Event
Force Majeure Triggering Event
Private Sector Triggering Event
Terminal Event
Intermediary Event
#
Intervention Point
Theft / Exposure of Information
Countermeasures / Competitors Developed
Loss of Technological / Strategic Advantage
1
2
Increased Testing Required by Customer
Accidental / Malicious Data Release
Current or Future Contracts Canceled/Avoided by Customer
Loss of Contract
Reputation Harm
Industry Partners Unwilling to Share Information
Inability to Fully Integrate Models
Reduced Confidence in System
Under-Trust in Model
Under Use of Model
Over-conservative system
6
System Integrity Reevaluation
Loss of Historical Data
Inability to Reuse Models in Future Projects
Unnecessary Rework
Delays
3
7
Cyberattack Attempt
Successful Cyberattack
Increased Cybersecurity Vulnerabilities
Reduction of "Upkeep" and Indirect Costs
Pressure to Streamline Acquisition Process
Public Scrutiny
Congressional Scrutiny
Budget Cut
Economic Crunch
Hiring Freeze
Documentation Tampering
Misunderstood Model Assumptions
Modeling Tampering
Inaccurate Simulations / Performance Predictions
Failure During Verification/Validation
Physical Component Substitution
4
Weak Security Controls
Malicious Insertion
Compromised System
Component Tampering
5

Intervention Points

1) Compartmentalize sensitive information
2) Obfuscate sensitive data with false or misleading information
3) Isolated but readily accessible back-ups of data
4) Reviews/Comparisons of models between lifecycle stages
5) Multiple, independent simulations or component checkers
6) Isolated, independent backup equipment that can be switched to while primary equipment is being evaluated
7) Conduct regular "red-team" / penetration test exercises

Figure 3. Reference Cybersecurity CEM (Preliminary)

Discussion

Some of the vulnerabilities and interventions shown in Figure 3 are not unique to MCE environments. Some of the vulnerabilities will simply be exacerbated by the increased use of MCE environments and processes. Some of the interventions will require new, creative means of implementing. For instance, Intervention Point #1 in Figure 3 is "Compartmentalize sensitive information." Clearly this is already done with the use of SCIFs and the Need-To-Know (NTK) information framework. However, such methods may not be feasible if the benefits of model integration and collaboration offered by MCE are desired. Instead, new methods must be developed. An example of one such possibility is the Federal Drug Administration's (FDA's) Sentinel Initiative, which involves querying a distributed system and receiving anonymized, aggregate data back (Office of Surveillance and Epidemiology, 2010). Such a system may allow modeling software to communicate across domains and locations, while still ensuring that even if one location is breached, only some information is exposed.

This Reference CEM does omit vulnerabilities and interventions that are entirely unchanged, however. For example, practices like the security clearance system and restricting the usual of digital storage media will remain effective interventions that are not significantly impacted by MCE environments.

One set of vulnerabilities that came up repeatedly in both the interviews and was observed in the class activity dataset were those that passed through the reputation harm intermediate event, as shown in Figure 4. Despite the frequency that the potential for this vulnerability was raised, few interventions were proposed for post-breach. This suggests that program managers and systems engineers could use more training in how to respond to breaches, particularly prominent ones, instead of just how to prevent them. While in the private sector there is evidence suggesting that the reputation harm incurred by a prominent breach does not significantly impact the firm (Lange & Burger, 2017), contractors to the government are known to suffer significant financial penalties due to breaches, even when such a breach is unrelated to their government duties (Braun, 2014; Overly, 2017). In a defense acquisition environment, there is thus significant incentive to having program managers (and the enterprise as a whole) well prepared to respond to major breaches.

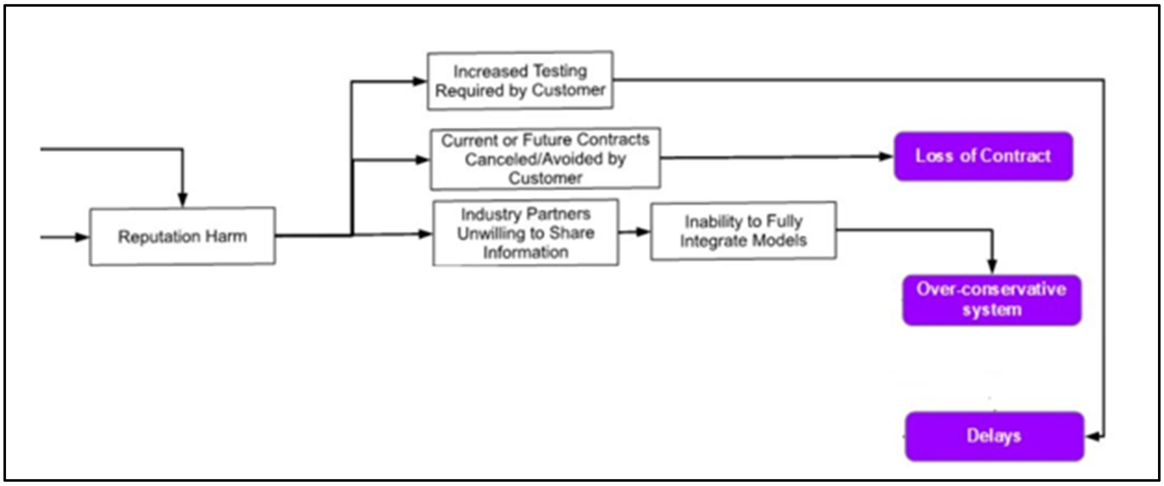

Increased Testing Required by Customer
Current or Future Contracts Canceled/Avoided by Customer
Loss of Contract
Reputation Harm
Industry Partners Unwilling to Share Information
Inability to Fully Integrate Models
Over-conservative system
Delays

Figure 4. Reputation Harm Vulnerabilities, Section of Figure 3

CEM is intended to supplement, rather than replace, existing vulnerability assessment methods, particularly when it comes to cybersecurity. In this way, it can help fulfill the requirements set by NIST's Risk Management Framework (RMF; Ross et al., 2016) and the DoD's Defense Federal Acquisition Regulation Supplement (DFARS; Manufacturing Extension Partnership, 2017b). These regulations have shifted how government contractors handle cybersecurity. Previously, one-time assessments were completed and defensive practices instituted. Now the process is more dynamic. Contractors have to continuously assess threats and develop countermeasures as they arise, both with regards to the end-system and to the enterprise. CEM can potentially assist in this by serving as a reference that can be revisited as new threats arise.

Future Directions

As the research progresses, three directions of future research are being pursued. The first is to conduct a second round of interviews with other stakeholders in the acquisition process. The second is to evolve an interactive version of the Reference CEM. The third is to compare vulnerabilities present in MCE environments with those present in other, comparable fields.

Future Interviews

The interviews thus far have been with program managers and system engineers (the people who "live in" in the MCE environment). As this research proceeds, a future round of interviews with MBSE and MCE toolmakers and leaders of enterprise model-centric environments is planned. Several of the interviewees expressed an interest in increased enterprise-level ownership of MCE environments. Additionally, a few expressed concern about the degree of security in MCE toolsets. Thus it is worth talking to such individuals about their perspectives on vulnerabilities in MCE environments.

Interactive Tool

An interactive version of the CEM, which enables easy sorting and adding vulnerabilities, is desired. This would make the method more accessible, similar to how NIST's Cybersecurity Assessment Tool (Manufacturing Extension Partnership, 2017a) makes the RMF (Ross et al., 2016) more approachable to small manufacturers. Additionally, it could serve as a platform for future usability testing of CEM in MCE programs. In future research, an interactive demonstration prototype will be generated to synthesize the research outcomes and show how these can be used in practice.

Healthcare Industry Comparison

There is some indication that program managers may be well served by observing fields that are somewhat analogous to defense acquisition in order to derive helpful metaphors (Karas, Moore, & Parrott, 2008) or lessons learned (German & Rhodes, 2016).

The healthcare industry shows promise for such an analogy to cybersecurity in MCE environments. The healthcare industry deals with sensitive information, computer equipment, and high pressure environments. All of these are present at numerous stages of operation. Patient records have to be transferred from one system to another and be available to medical practitioners. Researchers need to be able to query systems in order to provide improved medical treatment but cannot violate individuals' privacy. They must do all this and more while under constant threat of cyberattack, as recent events have shown (Ryckaert, 2018; Woollaston, 2017; Zetter, 2016).

Engineers and researchers have made significant headway in making medical devices more interoperable with one another, particularly when it comes to sharing data securely (Goldman, 2014). Increasingly, model-based methods are being used to assess and design medical systems (Pajic et al., 2014). As was related in the Discussion section, the FDA's Sentinel Initiative seeks to enable active querying of medical data while preserving individual privacy.

All of these endeavors are strikingly similar to the challenges currently faced in defense acquisition. This suggests that there may be benefit in conducting a systematic comparison of the two fields. The healthcare industry, along with other fields, will be examined for potential metaphors and lessons learned that are applicable to understanding vulnerabilities in MCE environments.

Conclusions

Acquisition programs increasingly use model-centric approaches, generating and using digital assets throughout the life cycle. Recent advancements support new model-centric practices, yet uncertainties can lead to model-related vulnerabilities jeopardizing program success. Extending recent research (Reid & Rhodes, 2018) on vulnerability assessment of model-centric programs to cybersecurity, anticipated results are empirically-grounded cybersecurity vulnerabilities related to model-centric acquisition programs, and a prototype using a CEM reference model with dynamic analytic tools.

References

Arkin, B., Stender, S., & McGraw, G. (2005). Software penetration testing. *IEEE Security and Privacy, 3*(1), 84–87. https://doi.org/10.1109/MSP.2005.23

Baldwin, K. J., & Lucero, S. D. (2016). Defense system complexity: Engineering challenges and opportunities. *The ITEA Journal of Test and Evaluation, 37*(1), 10–16.

Blackburn, M., Verma, D., Dillon-Merrill, R., Blake, R., Bone, M., Chell, B., … Evangelista, E. (2017). *Transforming systems engineering through model-centric engineering. Hoboken, NJ: Systems Engineering Research Center*. Retrieved from http://www.sercuarc.org/wp-content/uploads/2014/05/A013_SERC-RT-168_Technical-Report-SERC-2017-TR-110.pdf

Braun, S. (2014, September 10). OPM plans to terminate contracts with USIS. *Federal News Radio.* Retrieved from https://federalnewsradio.com/management/2014/09/opm-plans-to-terminate-contracts-with-usis/

German, E. S., & Rhodes, D. H. (2016). Human-model interactivity: What can be learned from the experience of pilots with the glass cockpit? In *Conference on Systems Engineering Research*. Huntsville, AL.

Goldman, J. M. (2014, November). Solving the interoperability challenge. *IEEE Pulse.* Retrieved from https://pulse.embs.org/november-2014/solving-interoperability-challenge/

Gressin, S. (2017). The Equifax data breach: What to do. Retrieved March 27, 2018, from https://www.consumer.ftc.gov/blog/2017/09/equifax-data-breach-what-do

Hanna, J., Smythe, C., & Martin, C. (2018, January 24). China's Sinovel convicted in U.S. of stealing trade secrets. *Bloomberg*. Retrieved from https://www.bloomberg.com/news/articles/2018-01-24/chinese-firm-sinovel-convicted-in-u-s-of-trade-secret-theft

Hanselman, S. (2012). Everything's broken and nobody's upset. Retrieved from https://www.hanselman.com/blog/EverythingsBrokenAndNobodysUpset.aspx

Irvine, C., & Parfitt, T. (2013, July 11). Kremlin returns to typewriters to avoid computer leaks. *The Telegraph*. Retrieved from https://www.telegraph.co.uk/news/worldnews/europe/russia/10173645/Kremlin-returns-to-typewriters-to-avoid-computer-leaks.html

Kamali, M., Dennis, L. A., McAree, O., Fisher, M., & Veres, S. M. (2016). Formal verification of autonomous vehicle platooning. *Science of Computer Programming, 1,* 1–19. https://doi.org/10.1016/j.scico.2017.05.006

Karas, T. H., Moore, J. H., & Parrott, L. K. (2008). Metaphors for cyber security. *Sandia Report*. Albuquerque, NM: Sandia National Laboratories. Retrieved from http://evolutionofcomputing.org/Multicellular/Cyberfest Report.pdf

Kern, C., & Greenstreet, M. R. (1999). Formal verification in hardware design: *A survey. ACM Transactions on Design Automation of Electronic Systems, 4*(2), 123–193. https://doi.org/10.1145/307988.307989

Lange, R., & Burger, E. W. (2017). Long-term market implications of data breaches, not. *Journal of Information Privacy and Security, 13*(4). https://doi.org/10.1080/15536548.2017.1394070

Manufacturing Extension Partnership. (2017a). Cyber risk management. Retrieved March 29, 2018, from https://www.nist.gov/mep/cyber-risk-management

Manufacturing Extension Partnership. (2017b). DFARS cybersecurity requirements. Retrieved March 29, 2018, from https://www.nist.gov/mep/cybersecurity-resources-manufacturers/dfars800-171-compliance

Meadows, C. A. (1994). Formal verification of cryptographic protocols: A survey. *In International Conference on the Theory and Application of Cryptology* (pp. 133–150). Springer, Berlin, Heidelberg. https://doi.org/10.1007/BFb0000430

The MITRE Corporation. (2015). Terminology. Retrieved February 20, 2018, from https://cve.mitre.org/about/terminology.html

The MITRE Corporation. (2017). CVE-2017-5753. Retrieved February 20, 2018, from https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-5753

Morimoto, S. (2008). A survey of formal verification for business process modeling (pp. 514–522). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-69387-1_58

NASA Office of the Chief Engineer. (1994). NASA Public Lessons Learned System. Retrieved July 13, 2017, from https://llis.nasa.gov/

Office of Surveillance and Epidemiology (Ed.). (2010). *The Sentinel Initiative*.

Overly, S. (2017, October). IRS temporarily suspends contract with Equifax. *Politico*. Retrieved from https://www.politico.com/story/2017/10/12/irs-equifax-contract-suspended-243732

Pajic, M., Mangharam, R., Sokolsky, O., Arney, D., & Goldman, J. (2014). Model-driven safety analysis of closed-loop medical systems. *IEEE Transactions on Industrial Informatics, 10*(1), 3–16. https://doi.org/10.1109/TII.2012.2226594

Pope, G. (2017). A hazard analysis technique for the internet of things (IoT) and mobile. In *STAMP Workshop*. Cambridge, MA.

Pope, G. (2018). Combining STPA with compiler technology to identify vulnerabilities and hazards in software-controlled systems. In *STAMP Workshop*. Cambridge, MA.

Pope, G., & Yampolskiy, M. (2016). A hazard analysis technique for additive manufacturing. In *Better Software East Conference*. Orlando, FL. Retrieved from https://arxiv.org/ftp/arxiv/papers/1706/1706.00497.pdf

Raymond, N. (2017, August 31). U.S. charges Chinese-Canadian citizen with trade secret theft. *Reuters2.* Retrieved from https://ca.reuters.com/article/topNews/idCAKCN1BB2K8-OCATP

Reid, J. B., & Rhodes, D. H. (2016). Digital system models : An investigation of the non-technical challenges and research needs. In *Conference on Systems Engineering Research.* Huntsville, AL.

Reid, J. B., & Rhodes, D. H. (2018). Assessing vulnerabilities in model-centric acquisition programs using cause-effect mapping. In *15th Annual Acquisition Research Symposium.* Monterey, CA: Naval Postgraduate School.

Ross, R., Dempsey, K., Pillitteri, V. Y., Jacobs, J., & Goren, N. (2016). Risk management. Retrieved March 29, 2018, from https://csrc.nist.gov/projects/risk-management/risk-management-framework-(RMF)-Overview

Rovito, S. M., & Rhodes, D. H. (2016). Enabling better supply chain decisions through a generic model utilizing cause-effect mapping. In *Proceedings of the 2016 Annual IEEE Sytems Conference*. IEEE.

Ryckaert, V. (2018). Hackers held patient data ransom, so Greenfield hospital system paid $50,000. *The Indianapolis Star*. Retrieved from https://www.indystar.com/story/news/crime/2018/01/17/hancock-health-paid-50-000-hackers-who-encrypted-patient-files/1040079001/

Schwarz, E. (2018). Automating vulnerability discovery in critical applications. Retrieved from https://www.sei.cmu.edu/research-capabilities/all-work/display.cfm?customel_datapageid_4050=6487

Statt, N. (2018, March). Boeing production plant hit with WannaCry ransomware attack. *The Verge*. Retrieved from https://www.theverge.com/2018/3/28/17174540/boeing-wannacry-ransomware-attack-production-plant-charleston-south-carolina

Woollaston, V. (2017, May). The NHS trusts and hospitals affected by the Wannacry cyberattack. *Wired*. Retrieved from http://www.wired.co.uk/article/nhs-trusts-affected-by-cyber-attack

Young, W. E. (2013). A system saftey approach to assuring air operations against cyber disruptions. In *STAMP Workshop*. Cambridge, MA.

Young, W. E., & Porada, R. (2017). System-theoretic process analysis for security (STPA-SEC): Cyber security and STPA. In *STAMP Workshop*. Cambridge, MA.

Zetter, K. (2016, March). Why hospitals are the perfect targets for ransomware. *Wired*. Retrieved from https://www.wired.com/2016/03/ransomware-why-hospitals-are-the-perfect-targets/

Acknowledgment

This material is based upon work by the Naval Postgraduate School Acquisition Research Programs under Grant No. N00244-17-1-0011.

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL
555 DYER ROAD, INGERSOLL HALL
MONTEREY, CA 93943

www.acquisitionresearch.net