Trustworthy Refinement Through Intrusion-Aware Design (TRIAD)

Robert J. Ellison
Andrew P. Moore

October 2002
Revised March 2003

TECHNICAL REPORT
CMU/SEI-2003-TR-002
ESC-TR-2003-002

Carnegie Mellon
Software Engineering Institute

Pittsburgh, PA 15213-3890

Trustworthy Refinement Through Intrusion-Aware Design (TRIAD)

CMU/SEI-2003-TR-002
ESC-TR-2003-002

Robert J. Ellison
Andrew P. Moore

October 2002
Revised March 2003

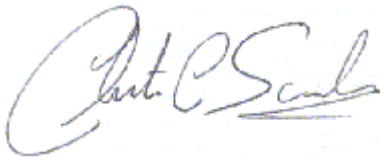
Networked Systems Survivability

This report was prepared for the

SEI Joint Program Office
HQ ESC/DIB
5 Eglin Street
Hanscom AFB, MA 01731-2116

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

FOR THE COMMANDER



Christos Scondras
Chief of Programs, XPK

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number F19628-00-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Table of Contents

Acknowledgements ..... **vii**

Executive Summary ..... **ix**

Abstract ..... **xv**

1 Introduction **1**
1.1 Background 3
1.2 Related Work 5
1.3 Structure of this Report 8

2 TRIAD Overview **9**
2.1 Model Structure 9
2.2 Model Execution 13
2.3 Intrusion-Awareness of TRIAD 19

3 Survivability Strategy Documentation **21**
3.1 Survivability Traceability 22
3.2 Documentation Artifacts 24

4 Survivability Strategy Development **29**
4.1 Threat Identification 30
4.2 Threat Dynamics Analysis 34
4.3 Risk Mitigation 40
4.4 Conceptual Architecture Refinement 46

5 Example: TRIAD Application **49**
5.1 First Iteration 51
5.2 Second Iteration 56
5.3 Final Concept 60

6 Conclusion **63**
6.1 Model Usage 63
6.2 Future Work 65

Appendix: Glossary 67

References 69

List of Figures

Figure 1: TRIAD Process Overview 10

Figure 2: Data Relationships 12

Figure 3: Execution of TRIAD 13

Figure 4: Survivability Strategy Refinement Process Overview 15

Figure 5: Technical Architecture Refinement Process Overview 16

Figure 6: Attack Tree Representation 18

Figure 7: Structured Intrusion Analysis 20

Figure 8: Survivability Tracing from Mission to Conceptual Architecture 23

Figure 9: Example Survivability Tracing Tables 23

Figure 10: Survivability Strategy Refinement Process 30

Figure 11: Simple Influence Diagrams 36

Figure 12: A Feedback Loop for Controlling Vulnerability 39

Figure 13: The Effects of Vulnerability Publication on Internet Vulnerability 40

Figure 14: eBiz Survivability Strategy Development Process 49

Figure 15: eBiz Concept of Operations 51

Figure 16: Online Credit Card Payment Transaction 52

Figure 17: Online Credit Card Payment Repudiation 53

Figure 18: Dynamics of Fraudulent Card Use 55

Figure 19: Extent of Legal Action 55

Figure 20: Composing Influence Diagrams of the First Iteration 56

Figure 21: Initial eBiz Conceptual Architecture 57

Figure 22: Customer Frustration Due to Strengthened Accountability.....59

Figure 23: Composite Influence Diagram 59

Figure 24: Final eBiz Conceptual Architecture.....61

Figure 25: TRIAD in SDM Process (1) As Mini-Spiral or (2) Through Integration 64

List of Tables

Table 1: Increased Threat Due to Architectural Vulnerability 34

Table 2: Survivability Tactics Addressing Types of Attacks 43

Table 3: Tabular Format for Survivability Requirements 47

Table 4: Initial eBiz Survivability Requirements 57

Table 5: Final eBiz Survivability Requirements 61

Acknowledgements

The authors thank members of the Software Engineering Institute (SEI) that helped in the production of this report. Members of the SEI's CERT Centers, including Dr. Sven Dietrich, Casey Dunlevy, Richard Linger, Dr. Howard Lipson, Dr. Thomas Longstaff, Dr. John McHugh, Dr. Nancy Mead, and Dr. Timothy Shimeall, provided valuable discussion, comments, and encouragement. Members of the SEI's Architecture Tradeoff Analysis Initiative, particularly Dr. Len Bass, provided valuable discussion of quality attribute design primitives and architectural tactics. We are also grateful to Pamela Curtis for her careful editing of multiple versions of this report. Finally, we thank our sponsors for support of this work.

Executive Summary

Military and business information system planners and engineers face daunting challenges to acquire, develop, and maintain complex, inter-networked systems that ensure mission success despite increasingly sophisticated computer network attacks. Decision-makers must choose among a vast array of information technologies while considering a host of vulnerabilities increasingly exploited by a malicious and coordinated attacker community. High confidence in a system's survivability requires an accurate understanding of the system's threat environment and the impact of that environment on system operations. Unfortunately, existing methods for building secure information systems are of limited help in guiding decision-makers toward coherent, holistic solutions that are both effective and affordable.

Existing secure system development methods typically promote isolated solutions to address individual concerns, resulting in patchwork designs that are rarely robust under malicious attack. Such solutions become uncoupled from the risks they are intended to address, obscuring the justification for their application. This usually results in either overkill—where solutions suggested are stronger, less efficient, and more costly than needed—or underkill—where solutions do not adequately address the mission-relevant threats. Part of the problem is that existing techniques focus almost exclusively on the bottom-up design of systems from existing components, while losing sight of the overall mission. There is little real understanding of how attacks that are likely to occur would affect the survival of what is important to the organization. Developers need to define a specific *survivability strategy* for the systems they build that describes the overall approach to resist, recognize, recover from, and adapt to mission-compromising attacks. Both tactical and strategic approaches are needed, but tactical mechanisms must satisfy strategic objectives to ensure mission success.

This report proposes an intrusion-aware design model called trustworthy refinement through intrusion-aware design (TRIAD). TRIAD helps information system decision-makers formulate and maintain a coherent, justifiable, and affordable survivability strategy that addresses mission-compromising threats for their organization. The goals of a survivability strategy are to provide a documented response to the primary threats to the mission; to provide a justification for and the limitations of the system design; to support the design and implementation of the desired system behavior across multiple systems and multiple development teams; and to support maintenance and evolution as the system operations and threat environment evolve over time.

Formulation of a survivability strategy helps to ensure a solid basis for DoD system acquisition and development. Maintenance of the strategy is necessary to ensure that it remains robust in the face of inevitable changes in the threat environment. TRIAD also helps decision-makers evaluate and maintain an information system design in terms of its ability to implement a survivability strategy. Evaluation of the system design helps to ensure that the strategy is implemented properly. Such a capability will be invaluable in evaluating responses to DoD system acquisition request for proposals (RFPs). Maintenance of the design is necessary to ensure that it continues to implement the strategy in the face of the inevitable evolution of the system design and implementation.

The Model

Survivability is the capability of a system to fulfill its mission by preserving essential services, even when systems are penetrated and compromised. Survivable systems development is a domain in which the optimal refinement strategy is unclear during the early stages of system design, particularly where unbounded, network-based systems are involved. Much experimentation and analysis is needed before a solution can be found with an acceptably small degree of residual risk of mission failure. We thus adopt the structure and philosophy of Boehm's spiral model as the basis for TRIAD. Each iteration of TRIAD gradually refines the system architecture based on the whiteboard prototyping, risk analysis, and risk mitigation of any previous iteration. This iteration permits adjustments and corrections to be made to the requirements, architecture, or resulting risks based on new experience and evidence.

The spiral structure of TRIAD proceeds through three sectors:

I. Architectural Strategy – This sector derives justifiable system survivability requirements and high-level conceptual architecture from the need to ensure mission success despite penetrations and compromises.
II. Architectural Instantiation – This sector refines the technical architecture within the constraints set by the conceptual architecture by identifying and integrating the critical technical building blocks.
III. Environmental Analysis – This sector represents the threat environment and analyzes its impact on system operation, including the system's ability to carry out its mission successfully.

Whereas sector I activities refine the conceptual architecture top-down from the mission objectives, sector II activities instantiate the conceptual architecture, as a technical architecture, from available technical components. Both refinement and instantiation are an essential part of the system development process, and TRIAD supports them explicitly in each iteration. The combination of the conceptual architecture, produced in sector I, and the technical architecture, produced in sector II, constitutes the system's survivability architecture. Sector III activities ensure that the threat environment is considered consistently through all iterations of architectural refinement. The refinement and analysis on which TRIAD is based uses generic, reusable information that should make the overall process affordable and efficient.

The initial iterations of the TRIAD spiral model focus on defining the survivability strategy; later iterations focus on the technical refinement of this strategy. The development of the survivability strategy cannot be completely isolated from its technical refinement. Refining the survivability strategy requires a certain amount of bottom-up thought, to ensure that the strategy is, in fact, implementable in a cost-effective way. Likewise, refining the technical architecture requires a certain amount of top-down thought, to ensure that the strategy is implemented in support of the overall mission. Progress continues until the set of artifacts produced for each sector is final and the level of residual risk of mission failure is acceptable to the stakeholders involved.

The practicality of TRIAD depends on being able to characterize effective and affordable threat and response patterns that are useful in building survivable systems. Survivability tactics help characterize such patterns. A *survivability tactic* is a generic representation of an architectural approach to resist, recognize, recover from, or adapt to some pattern of attack in a specific context. Survivability tactics describe strategic responses to general patterns of attack, such as the various forms of denial of service attacks and responses. A conceptual architecture formed from such survivability tactics forms the survivability strategy for the system. We expect survivability tactics to be useful at all levels of threat analysis and system development, from the most strategic to the most tactical.

Benefits

TRIAD helps engineers understand complex interactions among the information system, its mission, and its threat environment at all levels of system architectural refinement. Information systems include any combination of information technology and people's activities using that technology to support operations, management, and decision-making. TRIAD focuses on patterns of attack and strategies for surviving attack at an architectural level to avoid being overwhelmed by the details of individual component vulnerabilities or piecemeal security solutions. We focus on malicious attacks, rather than non-malicious failures or accidents, because of the increasing sophistication, frequency, and severity of such attacks and the inadequacy of existing approaches for dealing with them. We focus primarily on large-scale, highly distributed, and inter-networked information systems, such as Internet-based applications. Where available, TRIAD promotes using available security building blocks to help resist, recognize, and respond dynamically to likely intrusions. We consider both technological and procedural building blocks, since individual technological solutions to specific survivability problems may be unavailable, too immature, or too costly for the organization building the system. TRIAD facilitates planning for the inevitable change to the threat and operational environment and helps determine the effect of that change on continued mission success.

The strategic nature of TRIAD motivates its use during the early phases of system development, during requirements capture and high-level architecture formulation. TRIAD analysis provides insight and understanding into alternative business structures and strategies that optimally exploit information technology and clarify the role of that technology to ensure sur-

vival of the organizational mission. The holistic nature of TRIAD helps to ensure that all potential threat and solution areas are considered, down to an architectural level of analysis. TRIAD provides a means for analyzing the effects of observed trends in attacker behavior. Linkages between mission threats and risk mitigators are preserved through traceability documentation. Although accurate incident and vulnerability data is becoming more readily available, there are still large gaps in our understanding of intruder behavior. Our methods benefit from the availability of such data where it exists, but do not depend on it in order to provide useful insights into the impact of the threat environment on system operations.

Application

TRIAD is generic in nature, partitioning the design space into three primary sectors. Activities within each sector can be assembled into a specific, working model in many ways. The details of the best assemblage will depend largely on the domain of application and the skills of the development team. This report illustrates a detailed instantiation of TRIAD that shows how one might initiate the process, followed by iteration through the sector activities, and completing when an acceptable degree of residual risk is determined. This instantiated TRIAD model is used to refine the survivability strategy for a hypothetical business that sells products over the Internet. TRIAD is used to analyze the threat of and strategic responses to a rise in fraudulent purchases. Although this example is more illustrative than realistic, we expect that real-world systems could be analyzed at a strategic level with only a modest increase in complexity.

TRIAD does not deal specifically with many issues required of a comprehensive system development life cycle. Developers will need to resolve these issues to incorporate TRIAD into their system development and maintenance process. We believe that mission-related survivability requirements must be used to determine the overall shape of the architecture and must, therefore, be the focus of the initial iterations of the design process. Functions or properties required or desired that do not contribute to the mission must fit within the parameters defined by the survivability architecture and must not significantly lower the confidence that the system owners have in that architecture. This report outlines two approaches for incorporating TRIAD into a comprehensive system development life cycle: as a separate up-front mini-spiral or by more fully integrating design activities into the life-cycle process. A detailed approach of how to do this depends largely on the details of the system problem domain and the development environment, and is beyond the scope of this report.

Future Work

TRIAD provides a solid foundation for the further refinement, experimentation, and validation of an approach to exploit knowledge of intruder behavior to improve system architecture design and operations. We plan a two-pronged approach: TRIAD tool development and TRIAD application.

Certain aspects of TRIAD are amenable to some form of automated or semi-automated tool support. Developing appropriate tools will support TRIAD's application to larger and more complex problems in varied domains. We believe that system dynamics provides a foundation for developing methods and tools that help engineers understand, characterize, and communicate the impact of a malicious threat environment on organizational and system operations and their respective missions. Further development of threat dynamics promises to provide structured and justifiable guidance on how an organization can best adopt policies, procedures, and technology to respond to the threat environment. TRIAD tool support will integrate and refine existing tools as appropriate (e.g., tools for system dynamics, attack trees, or intrusion analysis) and support the documentation and use of survivability tactics.

We also plan to continue to explore the viability of TRIAD and refine it through its application to the focused analysis of very specific problem situations. Each example will involve the identification of a specific problem situation, a TRIAD analysis and mitigation of that situation, and a characterization of the improvement gained through the analysis and mitigation. By focusing on specific problems in a diverse set of narrow domains, we expect to get quick feedback on the efficacy, flexibility, and scalability of the model and insights into how to improve it.

Later work will involve a full-scale application of TRIAD and the tool support developed to demonstrate its scalability to more complex problems. TRIAD targets systems where there should be tighter integration between the security and system architectures. TRIAD demonstrations could target

- a new system in the early phases of development
- an existing system in which there are significant survivability reengineering issues
- an ongoing development in which TRIAD could document and analyze the tradeoffs between the system and security architectures

Demonstrations will require assembling TRIAD activities and structures into a working system development life-cycle model appropriate to the application domain and development environment. In addition to refining TRIAD based on the full-scale application, we plan to develop a tutorial for its use, with relevant examples, and initiate transition of the technology to an interested organization. TRIAD tool support, documentation of TRIAD case studies, and a detailed set of guidelines for TRIAD's application in varied settings should help make a compelling case for the model's use and transition. Ultimately, with effective tool support and evidence of its efficacy, we expect that TRIAD will be integrated with more comprehensive life-cycle models for the development and maintenance of high-confidence systems.

Abstract

High confidence in a system's survivability requires an accurate understanding of the system's threat environment and the impact of that environment on system operations. Unfortunately, existing development methods for secure and survivable information systems often employ a patchwork approach in which the focus is on deciding which popular security components to integrate rather than making a rational assessment of how to address the attacks that are likely to compromise the overall mission. This report proposes an intrusion-aware design model called trustworthy refinement through intrusion-aware design (TRIAD). TRIAD helps information system decision makers formulate and maintain a coherent, justifiable, and affordable survivability strategy that addresses mission-compromising threats for their organization. TRIAD also helps in evaluating and maintaining an information system design in terms of its ability to implement a survivability strategy. This report demonstrates the application of TRIAD to the refinement of a survivability strategy for a business that sells products over the Internet.

TRIAD provides a solid foundation for the further refinement, experimentation, and validation of an approach to exploit knowledge of intruder behavior to improve system architecture design and operations. Ultimately, with effective tool support and evidence of its efficacy, TRIAD will be integrated with more comprehensive life-cycle models for the development and maintenance of high-confidence systems.

1 Introduction

Military and business information system planners and engineers face daunting challenges to acquire, develop, and maintain complex, inter-networked systems that ensure mission success despite increasingly sophisticated computer network attacks. Decision-makers must choose among a vast array of information technologies while considering a host of vulnerabilities increasingly exploited by a malicious and coordinated attacker community. Unfortunately, existing methods for building secure information systems are of limited help in guiding decision-makers toward coherent, holistic solutions that are both effective and affordable.

Existing secure system development methods typically promote isolated solutions to address individual concerns, resulting in patchwork designs that are rarely robust under malicious attack. Such solutions become uncoupled from the risks they are intended to address, obscuring the justification for their application. This usually results in either overkill—where solutions suggested are stronger, less efficient, and more costly than needed—or underkill—where solutions do not adequately address the mission-relevant threats. Part of the problem is that existing techniques focus almost exclusively on the bottom-up design of systems from existing components, while losing sight of the overall mission. There is little real understanding of how attacks that are likely to occur would affect the survival of what is important to the organization. Developers need to define a specific *survivability strategy* for the systems they build that describes the overall approach to resist, recognize, recover from, and adapt to mission-compromising attacks. Both tactical and strategic approaches are needed, but tactical mechanisms must satisfy strategic objectives to ensure mission success.

Information system administrators often, unintentionally, evolve the systems that they manage in directions that solve narrow, short-term problems at the expense of strategic objectives and mission survivability. A locally optimal decision-making approach often leads down a path that is far from globally optimal. This is analogous to winning the battle, but losing the war. Administrators' narrow focus on the resolution of the "problem of the day" is not surprising since, even if there is a documented survivability strategy, it is often difficult to tell how well an existing information system implements that strategy. Exacerbating this situation, the threat environment for Internet-based systems is extremely dynamic, requiring regular re-evaluation of survivability in light of a change in attacker activity or an improved understanding of perceived threats.

High confidence in a system's survivability requires an accurate understanding of the system's threat environment and the impact of that environment on system operations. Reduc-

tionist techniques that delve into low-level design and implementation while losing sight of the overall environment are doomed to failure. This report proposes an intrusion-aware design (IAD) model called trustworthy refinement through intrusion-aware design (TRIAD). TRIAD helps information system decision-makers

- formulate and maintain a coherent, justifiable, and affordable survivability strategy that addresses mission-compromising threats for their organization. Formulation of a survivability strategy helps to ensure a solid basis for DoD system acquisition and development. Maintenance of the strategy is necessary to ensure that it remains robust in the face of inevitable changes in the threat environment.
- evaluate and maintain an information system design in terms of its ability to implement a survivability strategy. Evaluation of the system design helps to ensure that the strategy is implemented properly. Such a capability will be invaluable in evaluating responses to DoD system acquisition RFPs. Maintenance of the design is necessary to ensure that it continues to implement the strategy in the face of the inevitable evolution of the system design and implementation.

TRIAD helps engineers understand complex interactions among the information system, its mission, and its threat environment at all levels of system architectural refinement. Information systems include any combination of information technology and people's activities using that technology to support operations, management, and decision-making. We focus primarily on large-scale, highly distributed, and inter-networked information systems, such as Internet-based applications.[1] Modern computer/network-based information systems typically cross organizational boundaries and have no central administration and no unified security policy. One cannot control, or even know the number and nature of, nodes connected to unbounded Internet-based information systems. The distinction between insider and outsider may be dynamic in that a partner for one activity may be a competitor or adversary for another.

Society is becoming increasingly vulnerable to high-impact threats to complex, unbounded systems. TRIAD enables information system engineers to use known and hypothesized attack patterns to iteratively improve and continually maintain system survivability, even as the system and threat environment evolve over time. TRIAD focuses on patterns of attack and strategies for surviving attack at an architectural level to avoid being overwhelmed by the details of individual component vulnerabilities or piecemeal security solutions. We focus on malicious attacks, rather than non-malicious failures or accidents, because of the increasing sophistication, frequency, and severity of such attacks and the inadequacy of existing approaches for dealing with them. We focus on attacks that are likely for the system of interest, rather than on all attacks that are theoretically possible, to ensure cost-efficiency and relevancy of TRIAD application and the solutions that it promotes. Where available, the model

[1] Henceforth, unless otherwise indicated, our use of the term "system" specifically refers to such a large-scale, highly distributed, inter-networked information system, which includes both information technology and its operational context in combination.

promotes using available security and survivability building blocks to help resist, recognize, and respond dynamically to likely intrusions. We consider both technological and procedural building blocks, since individual technological solutions to specific survivability problems may be unavailable, too immature, or too costly for the organization building the system.

TRIAD facilitates planning for the inevitable change to the threat and operational environment and helps trace the effects of change back to the survivability requirements and architecture. In particular, we require traceability of the architectural solutions back to the intrusions that they are supposed to address. Traceability documentation is essential for system modifications caused by changes in the organization's risk profile, the appearance of new attack patterns, the availability of new technology supporting both functional and security requirements, and changes in the underlying work processes that affect the vulnerability and risk analysis.

This report describes the primary elements, key relationships, and supporting techniques of TRIAD. The model does not represent the whole development process, but only that part having to do with architectural refinement and only from the perspective of survivability. In particular, we do not represent those parts of the process needed to refine more general system function or to consider other quality attributes in addition to survivability. Nevertheless, this model provides a solid foundation for the further refinement, experimentation, and validation of an approach to exploit knowledge of intruder behavior to improve system architecture design and operations.

1.1 Background

Developers in many engineering disciplines rely on engineering failure data to improve their designs. Imagine the result if bridge builders had ignored the lessons learned from the torsional oscillations that caused the Tacoma Narrows Bridge to collapse. Or if ship builders had ignored the lessons learned about inadequate lifeboat space and manning that allowed the great loss of life when the Titanic sank. Engineering success requires that we also learn from the less famous disasters. The aerospace community, for example, has institutionalized a means for learning from air traffic accidents that has resulted in very low risk of death during air travel, despite its inherent hazards. Successful architects design structures to survive known faults in building materials, construction methods, and the environment.

Businesses and governments have historically been reluctant to disclose information about security failures, i.e., intrusions, on their systems for fear of losing public confidence or for fear that other attackers would exploit the same or similar vulnerabilities. However, increased public interest and media coverage of the Internet's security problems have resulted in increased publication of attack data in books, Internet newsgroups, and CERT® security adviso-

® CERT is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

ries. Unfortunately, information system developers use information on security failures, i.e., intrusions, in only a reactive way, to patch systems that they have already fielded, and even then in a very incomplete and inefficient manner [Arbaugh 00]. Information systems being built and managed today are prone to the same or similar vulnerabilities that have plagued them for years.

Survivability is the capability of a system to fulfill its mission by preserving essential services, even when systems are penetrated and compromised. Survivability is highly dependent on the flexibility of information system structures, which must be planned for and built in very early in the system design process. Bart Prakken aptly describes this need for flexibility:

> *The structure of organizations is comparable with the skeleton of vertebrates. These creatures need some solidity to survive in a hostile environment. However, if vertebrates have too much structure, because of arthritis for instance, their mobility will be hampered, with consequences that are probably fatal. The same arguments apply to the structure of organizations. Too much structure diminishes the mobility—flexibility—in an unacceptable way. And flexibility is especially important in turbulent (hostile) environments. Therefore, organization structures (containing information structures), with a well-considered balance between rigidity (structure) and flexibility, are necessary preconditions for creating opportunities for long-term survival. [Prakken 00]*

Flexibility is particularly crucial for the survivability of Internet-based information systems in order to ensure the availability of adequate responses to a rapidly changing threat environment.

Survivability requires a strategy to recognize, recover from, and adapt to intrusions, as well as, to the extent possible, to prevent intrusions in the first place. Survivability properties typically emerge from the architectural interaction of system components, and as such survivability must be considered very early in the development process [Fisher 99]. Considering survivability too late can create a system design that "hard-codes" mission vulnerabilities, making it too difficult, too costly, or downright impossible to build survivable implementations of that design. A survivable system design evolves, not by chance, but through insightful planning to build in the flexibility needed to respond to the likely threats. An organization's business work processes, including the operation and administration of the technology that supports those processes, are absolutely essential to the survivability of an organization's mission.

TRIAD is built on the premise that a much more proactive use of available attack information is needed to build cost-effective systems that survive attack with high confidence. Building affordable survivability architectures demands an understanding of the system's threat environment so that effort is spent on the likely intrusions rather than all possible ones. Unfortu-

nately, much of the available attack information is very detailed in terms of software versions, enterprise-specific configurations, and attacker-specific scripts. Such details have a relatively short life as the attackers create and revise their tools and methods. However, the general patterns of attack are much more constant over time. Attacks may target people, processes, and physical structures, as well as the system technology. Likewise, a survivability strategy must allow procedural, physical, and technological remedies to mission vulnerabilities to ensure the viability and affordability of the remedy [Anderson 01].

Gaining confidence in a system's survivability requires showing that the system is adequately resilient to likely patterns of attack. The dynamic nature of the intrusion environment demands that TRIAD, and the analysis techniques on which it is based, help discover and hypothesize about new sources and patterns of attack, in addition to known attacks by known adversaries. Attack patterns describe general attack strategies, such as the various forms of denial of service attacks, and can be structured so that they can be applied in a variety of contexts [Moore 01a]. The CERT Coordination Center's (CERT/CC's) experience analyzing the survivability of real systems across industry and government and collecting actual Internet-based attack data is leading to a more in-depth understanding of attack patterns, trends, and countermeasures [Ellison 99, CERT 02].

1.2 Related Work

A few efforts across industry and government are pursuing research to improve development methods for secure and survivable inter-networked systems of systems, with a focus on the threat environment. Neumann provides important insights into and an overview of supporting mechanisms for the development of system survivability architectures [Neumann 00]. The Information Assurance Technical Framework (IATF) includes extensive guidelines for choosing security mechanisms to incorporate into potentially large-scale, mission-critical systems, based on a high-level characterization of the threat and value of information protected [IATF 02]. Another paper outlines a secure system engineering methodology based on a more extensive analysis of the threat environment, and is, therefore, somewhat more aligned with our approach [Salter 98]. Work in the area of intrusion tolerance, which is primarily funded in the U.S. by DARPA and in Europe through the MAFTIA project, is intrusion focused and tackles large-scale distributed systems survivability, but that research has usually ignored non-technical attacks and countermeasures [MAFTIA 02].

There have been many listings of building blocks for system security through the years. TRIAD derives a survivability strategy using building blocks that we call survivability tactics. Survivability tactics are a special kind of architectural tactic, which attribute-based design methods developed at the Software Engineering Institute (SEI) use to support achieving quality attributes in an architecture [Bachmann 02]. Other recent works by Anderson and Ramachandran are noteworthy, the latter of which describes useful primitives in the context of architectural design [Anderson 01, Ramachandran 02]. The IATF, described above, also

presents many security and survivability building blocks that are graduated according to strength of protection against malicious attack. Depending on the threat level expected and properties required of the application domain, the framework recommends particular mechanisms and the assurances required for those mechanisms.

There are many sources across the security and survivability literature of potential tactics for ensuring mission success. The RAND Corporation, for example, has published a method for improving the survivability of systems based on categories of predefined survivability vulnerabilities and techniques [Anderson 99]. Although RAND's method has not been applied extensively, the study surveyed a wide range of existing systems and research efforts on security and survivability to derive vulnerability and technique categories. The survivability techniques identified provide a good start at identifying techniques for building survivable systems that are useful for IAD. Other work on survivability architectures also provides useful input to the IAD process [Knight 00, Neumann 00].

A great range of work in security risk analysis contributes to our effort, including the areas of adversary modeling, attack specification, vulnerability/threat analysis, security-related taxonomies and databases, impact analysis, and red teaming. Security risk analysis involves the analysis of system threats and vulnerabilities and their potential impact on the system's mission. The three primary elements of risk can be defined as follows [DoD 99, DoD 00]:

- threat: any circumstance or event with the potential to cause harm to a system
- vulnerability: a system characteristic that could be exploited by a threat to harm a system
- impact: the extent of harm to a system that results from a threat's exploitation of a system vulnerability

Risk is formally defined as "a combination of the likelihood that a threat will occur, the likelihood that a threat occurrence will result in an adverse impact, and the severity of the resulting impact" [DITSCAP 99]. For our purposes, then, a malicious threat can be viewed as any activity that exploits a vulnerability in a system and results in a negative impact on mission success.[2]

Experience over the years in security risk analysis suggests a number of pitfalls to avoid [Soo Hoo 00].

- *Complexity*. Techniques often require explicitly considering all threats and vulnerabilities, from the most common to the most obscure, without some screening with regard to likelihood or impact. The resulting complexity tends to overwhelm the analysis.

[2] Henceforth, we refer to "malicious threat" simply as "threat," since this is our primary focus. We specifically refer to "non-malicious threats" where that distinction is needed.

- *Incompleteness*. Techniques often ignore key aspects of the risk management problem or make incorrect assumptions about the problem domain. This may, for example, result in technological threats or solutions being emphasized over procedural ones.
- *Data unavailability*. Techniques often require obtaining precise, quantitative data on the likelihood of threats and the severity of impact. In the real world, such data continues to be inconsistently collected and reported, and highly uncertain even when it is. Using highly uncertain "estimates" in places where precise data is required often leads to obviously faulty results or, even worse, to very misleading, but plausible, nonsense.
- *Threat/countermeasure decoupling*. Techniques of managing security risk solely through the use of popular security technology and practices without a link to the mission objectives or threats tend to decouple the countermeasures from the risk they are supposed to reduce. This lack of traceability makes it difficult to accurately assess the actual residual risk resulting from the use of technology and practices.
- *Static analysis*. Techniques generally deal only with the current threat environment with little regard to managing the system under changing threats. Increasingly rapid changes in the threat environment, which are characteristic of modern Internet-based systems, demand techniques that can be applied as part of an evolutionary design and maintenance life cycle.

There are, of course, no easy solutions to these problems. Early research in security risk analysis generally promoted comprehensive solutions that became overly complex. More recent approaches simplified the methods at the expense of completeness [Soo Hoo 00]. We make no claims to having solved these problems but believe our approach to intrusion-aware design makes inroads to managing the risk analysis problem from the survivability perspective.

While the above work contributes to developing a model for IAD, none of the efforts take advantage of the full potential of exploiting available attack information for improving system survivability. No one that we know of is looking in-depth at the problem of using attack patterns and trends during system architecture refinement to maintain system security and survivability, in a way that copes well with the transient nature of the threat environment. The objective of our work is to address this problem, dealing directly with survivability maintenance as the system mission, architecture, and threat environment change. TRIAD involves survivability risk mitigation at an architectural level. We do not "reinvent" security risk analysis, but leverage existing analysis techniques as appropriate. In the longer term, we hope to improve the accuracy and speed of risk analysis techniques by documenting commonly recurring attack patterns in a generic and reusable form.

1.3 Structure of this Report

Section 2 provides an overview of the model structure, which contains three primary sectors of activities: Architectural Strategy, Architectural Instantiation, and Environmental Analysis. Execution of the model can be viewed at an abstract level as starting with the development of the survivability strategy, which is embodied as a conceptual architecture, followed by the implementation of that strategy as a specific technical architecture. Since most existing methods focus on the bottom-up development of security and survivability architectures from existing technologies, the rest of the report focuses on the strategic aspects of survivability architecture development as a complement to existing methods. Section 3 describes the primary artifacts required to document and justify the survivability strategy. Section 4 describes a process that helps refine a coherent, justifiable, and affordable survivability strategy. Section 5 demonstrates the application of this process to the refinement of a survivability strategy in the e-commerce domain. Section 6 concludes the paper by discussing limitations of TRIAD and how to ameliorate some of these limitations through its incorporation in the larger system development life cycle. We also summarize directions for future work. Finally, an appendix provides definitions for important terms as they are used in this report.

2 TRIAD Overview

It is widely accepted that much of system architecting is creative in nature:

> *"Architectural design processes are inherently eclectic and wide-ranging, going abruptly from the intensely creative and individualistic to the more prescribed and routine. While the processes may be eclectic, they can be organized. Of the various organizing concepts, one of the most useful is stepwise progression or 'refinement'" [Maier 00].*

TRIAD was formulated around the central notion of refinement in architecting, which motivated the 'R' in TRIAD. This section describes an overview of the general model structure, followed by a more detailed discussion of the execution of the model to produce a robust survivability strategy and technical implementation of that strategy.

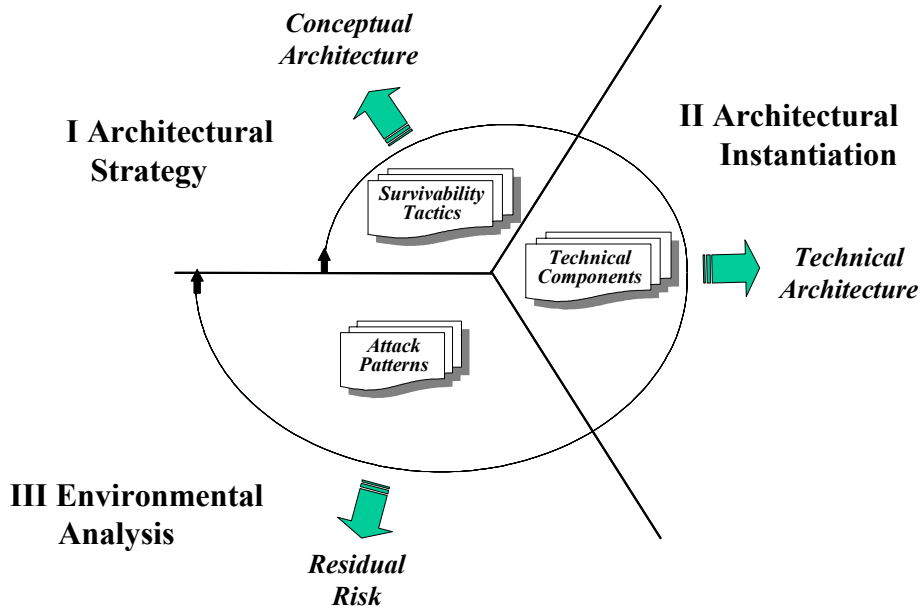
2.1 Model Structure

Maier observes that the process of system architecting is best "characterized as episodic, with episodes of abstraction reduction alternating with episodes of reflection and purpose expansion" [Maier 00]. To reflect this episodic nature of system architecting, TRIAD adopts the structure and underlying philosophy of the spiral model of system development [Boehm 88, Marmor-Squires 89]. The spiral model is intended for system and software development and enhancement in complex domains with which the developers have limited experience or domains where the best (or even a good) direction for system refinement is highly uncertain. Such domains require iterated refinement where each iteration gradually refines the system requirements, design, and implementation based on the experience of any previous iteration. This iteration permits adjustments and corrections to be made in the directions chosen for system refinement based on new evidence such as risk analysis, prototyping, and simulation. The original spiral model proceeds through four quadrants, each quadrant making progress toward improved understanding and refined documentation of the system requirements, design, and/or implementation.

Survivable systems development is certainly a domain in which the optimal refinement strategy is unclear during the early stages of system design, particularly where unbounded, network-based systems are involved. Much experimentation and analysis is needed before a solution can be found with an acceptably small degree of residual risk of mission failure. The spiral structure of TRIAD, which is shown in Figure 1, proceeds through three sectors: (I)

Architectural Strategy, (II) Architectural Instantiation, and (III) Environmental Analysis. Although the figure shows only the general structure of the model, a fully instantiated model involves multiple iterations through these sectors. Consistent with the original spiral model, each iteration gradually refines the system architecture based on the whiteboard prototyping, risk analysis, and risk mitigation of any previous iteration. Progress starts in the middle of the figure in sector 1 and proceeds along the spiral, the angular dimension of which indicates cumulative progress. An instantiation of the model involves multiple iterations through the sectors, which permits adjustments and corrections to be made to the requirements, architecture, or resulting risks based on new experience and evidence. Like the spiral model, TRIAD is equally applicable to the development of new systems and the enhancement of existing systems. Subsequent discussion describes the primary activities in each sector.

Figure 1: TRIAD Process Overview



Sector Overview

The Architectural Strategy sector (sector I) starts by elaborating the overarching mission of the system under design. Activities in the sector derive justifiable system survivability requirements and a high-level conceptual survivability architecture from the need to ensure mission success despite penetrations and compromises. The conceptual survivability architecture (henceforth abbreviated to conceptual architecture) describes the system function and structure at a level appropriate for the customer. As shown in Figure 1, the conceptual architecture derives from a collection of survivability tactics. A *survivability tactic* is a generic representation of an architectural approach to resist, recognize, recover from, or adapt to

some pattern of attack in a specific context.[3] Survivability tactics describe strategic responses to general patterns of attack, such as the various forms of denial of service attacks and response [CERT 01]. A conceptual architecture formed from such survivability tactics forms the survivability strategy for the system.

Activities in the Architectural Instantiation sector (sector II) refine the technical architecture within the constraints set by the conceptual architecture by identifying and integrating the critical technical building blocks. The technical survivability architecture (henceforth abbreviated to technical architecture) describes the function and structure of the system at a level of technical detail sufficient to actually build the system. This sector's activities proceed by identifying low-level technical components to instantiate the conceptual architecture. A *technical component* is any existing architectural building block such as commercial off-the-shelf (COTS) software and hardware.

Activities in the Environmental Analysis sector (sector III) represent the threat environment and analyze its impact on system operation, including the system's ability to carry out its mission successfully. The threat environment is derived from a collection of attack patterns. An *attack pattern* is a generic representation of deliberate and malicious activity that commonly occurs in a specific architectural context. An attack pattern may target people (e.g., social engineering attacks that use a computer virus), the operation of the technology (e.g., distributed denial of service attacks), or the context in which people do work (e.g., dumpster-diving attacks).

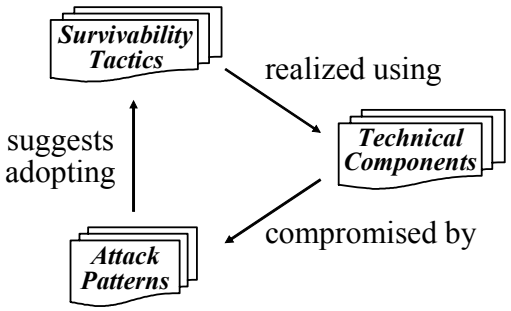
The distinction between the sectors may not always seem clear, and there is bound to be some overlap, just as there was in the original spiral model. However, we have fairly concrete distinctions between each of the three sectors. The difference between sector I and sector II is similar to the difference between requirements and a specification. Requirements may describe a general solution strategy, but leave open many design- and implementation-level details; a specification makes many of the concrete decisions on how to proceed, often in terms of specific components and connectors. Sector I activities refine the conceptual architecture top-down from mission objectives, whereas sector II activities instantiate the conceptual architecture, as a technical architecture, from available technical components. Both refinement and instantiation are an essential part of the system development process, and TRIAD supports them explicitly in each iteration. The combination of the conceptual architecture and the technical architecture makes up the system's survivability architecture. Finally, sector III focuses on the analysis of threat and impact given the architectural constraints specified in sector II, whereas sector I focuses on the description of requirements to mitigate the resulting risk. Sector III activities ensure that the threat environment is considered consistently through all iterations of architectural refinement.

[3] The use of the word tactic in this context is not meant to imply that the architectural approach addresses only short-term goals, but that the approach addresses specific concerns in an isolated context.

Data Relationships

The essential relationship of the data on which each sector is based is shown in Figure 2. Strategic approaches to ensure mission success suggest the use of specific technical components for survivability. These technical components, in turn, have certain vulnerabilities within the context of a system architecture that promotes certain attack patterns. Attack patterns, in turn, suggest the adoption of other survivability tactics. Of course, this could result in a never-ending cycle of analysis. The challenge for the intrusion-aware designer is to converge gracefully to a set of survivability tactics for survivability, each member of which is implemented as a set of technical components, that address likely attack patterns in an affordable and effective manner.

Figure 2: Data Relationships



Design maintenance may be needed due to changes in the mission objectives, changes in the underlying architecture, or changes in the threat environment.

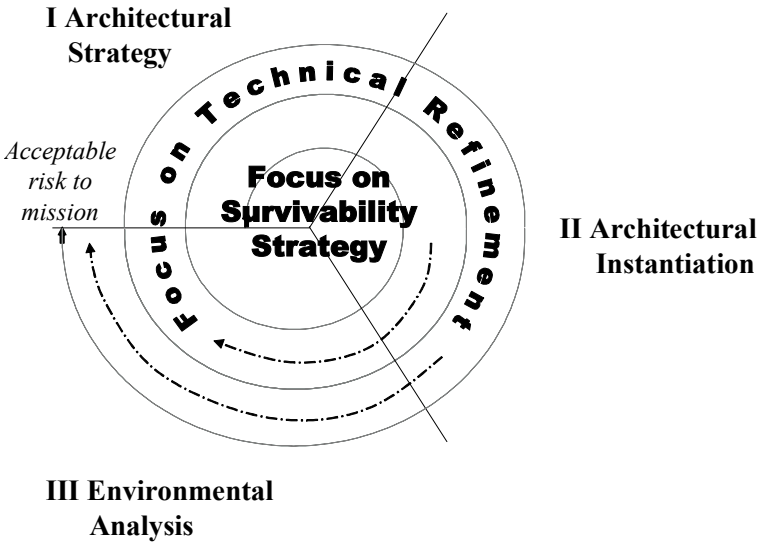
- Expanding the system mission may, for example, require a military command and control system for U.S. forces to also operate jointly with coalition forces. Contracting the mission or modifying its fundamental nature may, for example, require an eBusiness focusing on sales of high-end merchandise to transition to a strategy of high-volume discounted sales of lower-end merchandise because of various market pressures.
- Changes in the system architecture may be procedural or technological in nature. A business may decide to relax hiring practices in response to a highly competitive job market. A technological change may result when an eBusiness expands to physical sales of merchandise at multiple distributed sites. This change would require online inventory management and a level of trust in the workflows between the distributed sites.
- Changes in the threat environment may involve new types of attackers that need to be considered, or old types of attackers using new methods. New types of attackers may threaten an eBusiness when, for example, recent news reports publicize the eBusiness's dealings with unpopular organizations, making its system operations more susceptible to "hactivist" attack. Attackers who were previously considered a threat might take on new relevance with the appearance of a new class of attack tools that can be used to penetrate corporate perimeters and take control of intranet operations.

TRIAD emphasizes the interrelationships among the three sectors. Changes in mission objectives may lead directly to changes in system structure to support the modified objectives. Changes in system structure can, in turn, affect the threat environment, for example, through increased exposure. Finally, a change in the threat environment may lead to modified requirements to preserve survivability, and ultimately structural changes to support these requirements. The documentation promoted by our approach emphasizes traceability among sector artifacts to support continued maintenance of system survivability even after the system is fielded.

2.2 Model Execution

Execution of TRIAD, when viewed at an abstract level, starts with the development of the survivability strategy and continues with the implementation of that strategy as a specific, concrete survivability architecture. Figure 3 shows that the initial iterations of the model focus on defining the survivability strategy. Later iterations focus on the technical refinement of this strategy.

Figure 3: Execution of TRIAD



While the initial focus is on the composition of survivability tactics to produce the survivability strategy, the development of the survivability strategy cannot be isolated from its technical refinement. Refining the survivability strategy requires a certain amount of technical feasibility analysis to ensure that the strategy is, in fact, implementable in a cost-effective way. Likewise, refining the technical architecture requires top-down analysis to ensure that the strategy is implemented in support of the overall mission. Progress continues until the set of artifacts produced for each sector is final and the level of residual risk of mission failure determined by the Environmental Analysis sector's activities is acceptable to the stakeholders involved.

involved. The exact number of iterations of the spiral required for completion varies depending on the complexity of the application and the developers' experience with the application domain, but we typically expect convergence on an acceptable solution in two to four iterations.

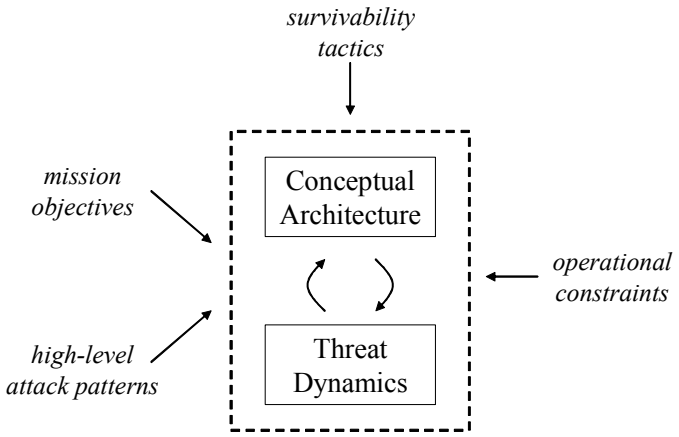
This section provides an overview of the development of the survivability strategy embodied as a conceptual architecture, followed by an overview of the technical refinement of the strategy into a technical architecture. The approach outlined here helps information system decision-makers at all levels understand the impact of a malicious threat environment on their organization's ability to achieve its mission. Furthermore, the approach helps formulate an effective and affordable strategic response to the attacks that are likely to compromise the mission. If a response is already planned or in place, the approach helps evaluate that response in light of the threat and make recommendations to improve its efficacy. The approach facilitates planning for the inevitable changes to the threat environment and system operations, and helps determine the effect of those changes on continued mission success.

Focus on Survivability Strategy

A successful organization has an implicit or explicit mission that characterizes its primary purpose as a set of high-level objectives. An organization's information technology, policies, procedures, personnel, and overall work context all exist to support the mission. We must be able to assess at any stage of architectural refinement the impact of a potentially evolving threat environment on the system and its overall mission as described. Our approach to the development and evaluation of the survivability strategy is based on a branch of operations research called *system dynamics* [Sterman 00]. We develop a specialized sub-domain, which we call *threat dynamics*, that interprets system dynamics to include explicitly hostile actions and the system operational response to such actions. Threat dynamics enables the modeling of the structure and dynamics of complex human-based systems, of which the relationship between the Internet-based attacker community and Internet-based information systems is a specific example. By defining a holistic view of the threat environment in the context of existing or proposed system operations, threat dynamics provides an overview of the general influences that the threat environment has on the ability of the system to fulfill its mission and a better understanding of strategic responses to counter likely threats.

Figure 4 depicts our high-level approach to survivability strategy refinement. The strategy, which is embodied as a conceptual architecture, is derived iteratively through its evaluation using threat dynamics. The strategy derives from the overall mission objectives and experience with recurring high-level attack patterns that document the primary intrusion scenarios that must be considered. Survivability tactics are broad architectural approaches to ensuring that such attacks do not threaten the survivability of the mission. Survivability tactics may help formulate the conceptual architecture, but such formulation must fit within the operational constraints of the application domain.

Figure 4: Survivability Strategy Refinement Process Overview



In terms of the TRIAD sectors, the conceptual architecture is defined primarily in sector I and is evaluated using threat dynamics in sector III. The operational constraints arise primarily due to technical considerations in sector II. Most of the development and evaluation of the survivability strategy occurs in sectors I and III. An essential role for sector II is to ensure that the survivability strategy can be implemented, although the details of exactly how to do that are left to later technical refinement. In addition, there may be a need to adjust the conceptual architecture in order to satisfy technical constraints that were not previously considered.

While threat dynamics helps enable understanding of the influence of the threat environment on the ability to achieve the mission, survivability tracing helps to document and justify the support that the survivability architecture provides to the mission. Survivability traceability is essential for managing changes in a way that maintains an organization's survivability over time. In this broader context, traceability can be defined as "a characteristic of a system in which the requirements are clearly linked to their sources (backward traceability) and to artifacts created during the system development life cycle based on these requirements (forward traceability)" [Ramesh 97]. In this definition, linkages are considered bidirectional. TRIAD sector I is responsible for backward traceability to the mission objectives, whereas sector II is responsible for forward traceability to the technical architecture. The threats addressed are those identified in sector III.

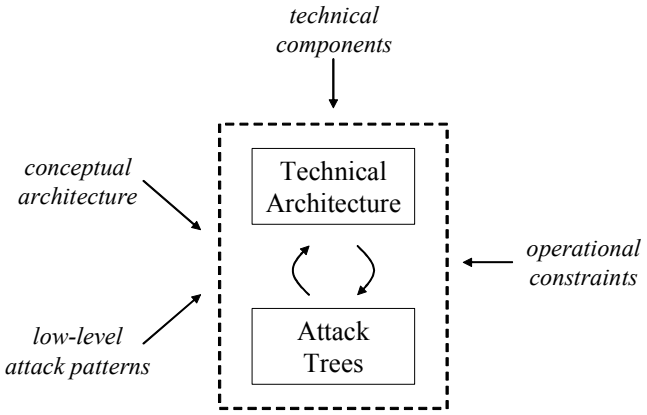
Traceability of requirements and decision choices from an organization's mission helps determine a system's survivability dependencies. Backward traceability can help assess the impact of changes to an organization's mission or threat environment. Forward traceability can help assess the impact of changes to the system architecture. Conventional wisdom in the requirements traceability community dictates that traceability be maintained only for mission-critical requirements [Ramesh 98]. This wisdom is exactly aligned with the mission focus of survivability, since the organizational mission provides the starting point for TRIAD tracing.

Survivability traceability requires a broader scope than typically adopted for general requirements traceability because of the breadth of the threats (e.g., from social engineering to technological compromise) and the countermeasures (e.g., from personnel, to procedural, to technological). Requirements and design choices must be consistently managed throughout the system lifetime to support continual risk management so that new threats and system operations do not lead to mission failure. Since TRIAD is an iterative refinement process, the mission, threats, requirements, and architecture may be only partially defined on any given iteration of the spiral. Therefore, the requirements definition and traces are incrementally refined as well.

Focus on Technical Refinement

Implementing the survivability strategy involves developing a technical architecture that instantiates the conceptual architecture. Figure 5 depicts our high-level approach to technical architecture refinement within the constraints set forth by the conceptual architecture. The technical architecture is derived iteratively through its evaluation using a technique called *attack trees*. Attack trees can be built by composing low-level attack patterns that have proven to be both likely and consequential. The attack patterns of interest for a particular application are those that may compromise the mission. Technical refinement must maintain the traceability of the conceptual architecture through the technical architecture.

Figure 5: Technical Architecture Refinement Process Overview



Critical areas of consideration in developing the technical architecture include

- the Intranet, which includes the organization's databases, applications, servers, workstations, internal networks, and procedures for their use
- the Perimeter, which includes firewalls, gateways, and physical mechanisms that protect the organization's intranet assets from external access
- the Extranet, which includes any networks outside the perimeter that must be relied on to achieve the organization's mission

The survivability tactics used in the conceptual architecture are implemented in terms of available technical components (non-developmental items) and, where needed, custom development. Technical components may be available either commercially or through government-sponsored research and development programs. Technical refinement involves identifying the responsibilities of individual components of the technical architecture that help achieve the survivability requirements. Refining requirements may involve tradeoffs in terms of costs or complexity of administration that may suggest changes to the conceptual architecture.

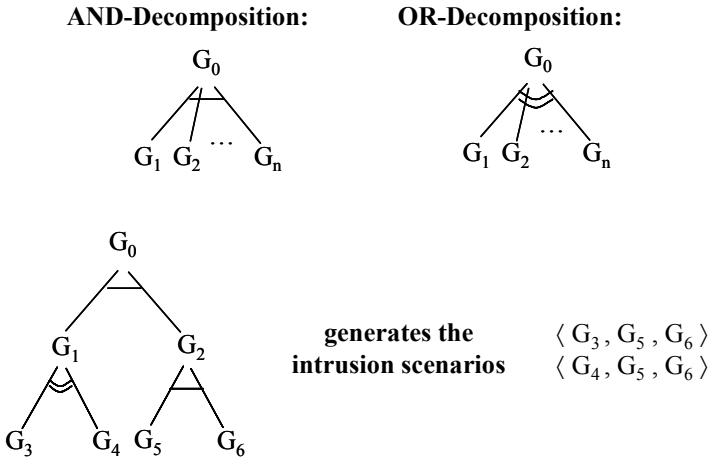
The large number of intrusions possible for any nontrivial system necessitates a scheme to organize related intrusions. Attack trees provide such an organizational scheme [Salter 98, Schneier 99, Schneier 00a]. They refine information about intrusions by identifying the compromise of enterprise security or survivability as the root of the tree. The ways that an attacker can cause this compromise are refined incrementally as lower level nodes of the tree.

A system typically has a set, or forest, of attack trees that are relevant to its operation. The root of each tree in a forest represents an event that could significantly harm the system's mission. Each attack tree enumerates and elaborates the ways that an attacker could cause the event to occur. Each path through an attack tree represents a unique intrusion on the enterprise. We decompose a node of an attack tree as one of the following:

- a set of attack subgoals that is represented as an AND decomposition. All of these goals must be achieved for the attack to succeed.
- a set of attack subgoals that is represented as an OR decomposition. If any of these goals is achieved, the attack succeeds.

We represent decompositions graphically as shown in Figure 6. The AND-decomposition represents a goal G_0 that can be achieved if the attacker achieves all of the goals G_1 through G_n. The OR-decomposition represents a goal G_0 that can be achieved if the attacker achieves any one of goals G_1 through G_n. In practice, we often represent attack trees textually, since the graphical representation can be awkward for nontrivial attack trees.

Figure 6: Attack Tree Representation



Attack trees consist of any combination of AND- and OR-decompositions. We generate individual intrusion scenarios from an attack tree by traversing the tree in a depth-first manner, an example of which is shown in Figure 6. In general, leaf goals are added onto the end of intrusion scenarios as they are generated. OR-decompositions cause new scenarios to be generated. AND-decompositions cause existing scenarios to be extended. Intermediate nodes of the attack tree do not appear in the intrusion scenarios, since they are elaborated by lower level goals.

Attack trees allow the refinement of attacks to a level of detail chosen by the developer. They exhibit the property of referential transparency as characterized by Prowell:

> *"Referential transparency implies that the relevant lower level details of an entity are abstracted rather than omitted in a particular system of higher level description, so that the higher level description contains everything needed to understand the entity when placed in a larger context" [Prowell 99].*

This property permits the developer to explore certain attack paths in more depth than others, while still allowing the developer to generate intrusion scenarios that make sense. In addition, refining the branches of the attack tree generates new leaves, resulting in intrusion scenarios at the new lower level of abstraction. The notion of referential transparency is critical to managing the complexity inherent in attack tree representations by constraining the refinement to an architectural level of abstraction. Moore describes the details of the above approach and an example of its application in a report available online [Moore 01a].

Attack trees can be used to improve a technical architecture by asking resistance, recognition, recovery, and adaptation questions at each of the attack tree nodes. Resistance questions ask how to prevent an attacker from successfully traversing this node to compromise the mission. Of course, the answer to resistance questions may not always be a cost-effective or practical

solution. Fundamental to the goal of survivability is the existence of recovery plans for those attacks that we cannot effectively prevent. We thus ask,

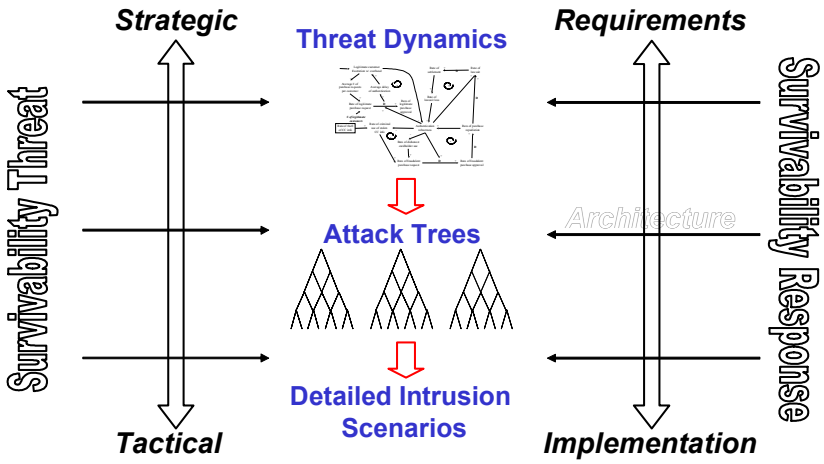
- How can we detect an attacker during an attempted attack or after a successful attack?
- How can we recover from any compromise?
- How can we adapt the system so that the intrusion cannot happen again?

Resisting an OR-branch of an attack tree resists the intrusion scenarios associated with that branch. Resisting attack nodes higher up in the attack tree hierarchy results in more effective blockage of the attacker, but also potentially in more extensive and costly changes to the system architecture and operations. Resisting an AND-branch of an attack tree resists all intrusion scenarios associated with the parent node of the branch. This leverage is gained because the attacker must traverse all branches of an AND-decomposition to achieve his goal; resisting any one of the AND-branches resists the goal defined by the parent node. The best technique (or combination of techniques) is chosen based on cost, practicality, and assurance of implementation. The type of recognition, recovery, and adaptation needed depends on the type of attack, i.e., the branch of the attack tree traversed. Attack trees need only be refined to a level that supports architectural analysis in a way that the enterprise stakeholders can accept as both sufficient and affordable. Moore describes a simple example of this type of analysis [Moore 01b].

2.3 Intrusion-Awareness of TRIAD

Figure 7 depicts an overview of TRIAD architectural analysis for robustness against mission-compromising attacks. The type of analysis ranges from the purely strategic, at the top of the figure, to the purely tactical, at the bottom of the figure. Threat dynamics is used to model the strategic impact of the threat environment on the ability of an organization to achieve its mission and to determine strategic responses to ameliorate the adverse effects. Threat dynamics provides insight and understanding into alternative business structures and strategies that optimally exploit information technology and clarifies the role of that technology to ensure survival of the organizational mission.

Figure 7: Structured Intrusion Analysis



The approach outlined above manages the complexity of the survivability risk analysis problem by focusing only on threats that can compromise the mission and only on vulnerabilities at a gross architectural level. The holistic nature of the threat dynamics starting point helps to ensure that all potential threat and solution areas are considered down to an architectural level of analysis. Threat dynamics provides a means for analyzing the effects of observed trends in attacker behavior. Linkages between the threats and the risk mitigators are preserved through the survivability tracing. Threat dynamics analysis benefits from the availability of incident and vulnerability data where it exists, but does not depend on it in order to provide useful insights into the impact of the threat environment on system operations.

The rest of this report focuses on the strategic aspects of TRIAD, including the threat dynamics aspect of Figure 7. As mentioned previously, many of the existing secure system development methodologies and risk analysis techniques focus on the more technical aspects of threat analysis and response. In addition, previous work at the CERT/CC has outlined an approach for using attack trees for the survivability analysis of systems at a technical level [Moore 01a]. In this light, the methods identified and exemplified here fill a critical gap in our ability to build secure and survivable information systems.

3 Survivability Strategy Documentation

Within the context of TRIAD, a survivability strategy is an integrated approach to resist, recognize, recover from, and adapt to mission-compromising attacks. The goals of a survivability strategy are to

- *provide a documented response to the primary threats to the mission*—Examples include the variety of responses possible for denial of service attacks [CERT 01]. The strategy for a deployed system should document the expected response and training for both system management and operations in support of that response.
- *provide a justification for and the limitations of the system design*—The strategy provides design rationale describing how the architecture supports the desired response to the threats. Justification may include information assurance policy, certification requirements, or arguments concerning due diligence. The strategy documents design options and tradeoffs and provides input supporting the review, inspection, and testing of the design and implementation. Limitations are often expressed as design assumptions that must be valid for the justification to hold.
- *support the design and implementation of the desired system behavior across multiple systems and multiple development teams*—The strategy is documented in a way that supports communication among multiple development teams during both acquisition and engineering. During an acquisition, the strategy provides an excellent starting position for request for proposals (RFPs), and can be useful for assessing the response to those proposals. The strategy documents the shared risks and responsibilities among multiple organizations. The strategy also supports incorporating system security requirements into the software development process as described in DoD information assurance policy and certification requirements [DoD 02, DITSCAP 99].
- *support system maintenance and evolution*—The strategy helps maintain design assumptions and verify the continued effective response to threats. This may involve analyzing the impact of new threats and changes in the operating environment. The strategy used by an existing system may have to be reengineered from existing documentation if that strategy was not explicitly documented as part of its development. TRIAD helps developers formulate and document the strategy as part of the overall system development process.

The primary information artifacts that pertain to the survivability strategy include

- mission objectives—the high-level purpose of the system in the eyes of the system owners
- mission threats—the threats to achieving the mission objectives
- survivability requirements—the requirements that support the resistance, recognition, recovery from, and adaptation to the mission threats
- conceptual architecture—a description of the system structure and function that ensures the survivability requirements are met with sufficient assurance

The survivability requirements and conceptual architecture taken together constitute the survivability strategy. The rest of this section characterizes these artifacts in more detail, starting with the required traceability among them.

3.1 Survivability Traceability

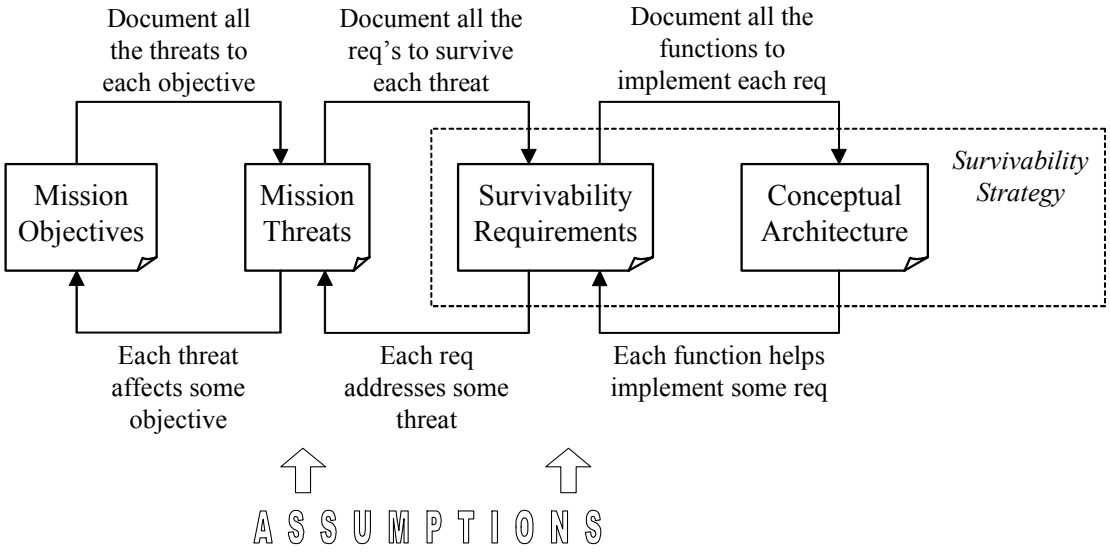
The justification of the survivability strategy requires arguing that the conceptual architecture supports mission success. Traceability has long been used to help ensure that a system's design and implementation conform to their requirements [Ramesh 97]. We define *survivability traceability* as the characteristic of a system in which the survivability requirements are clearly linked to their sources (mission objectives) and to the artifacts created during the system development life cycle based on these requirements (survivability architecture).

Figure 8 illustrates the tracing of mission objectives through the mission threats down to the survivability requirements, which include both the functional and non-functional system requirements. The survivability requirements are then traced onto the conceptual architecture. Justifying the survivability strategy requires arguing that the conceptual architecture supports mission success. This argument depends on the survivability tracing as a whole and is likely to rely on a set of assumptions about the operational environment, e.g., the stability of the external interfaces of the system. Such assumptions must be documented as part of the tracing process.

Survivability traceability also plays an essential role in helping system administrators manage the inevitable (and often unexpected) changes to an organization's objectives, structures, behaviors, or threat environment in a way that maintains the organization's survivability over time. This requires proactive change management that helps determine and, when possible, contain the effects of change. Survivability traceability maintenance promotes change management in TRIAD:

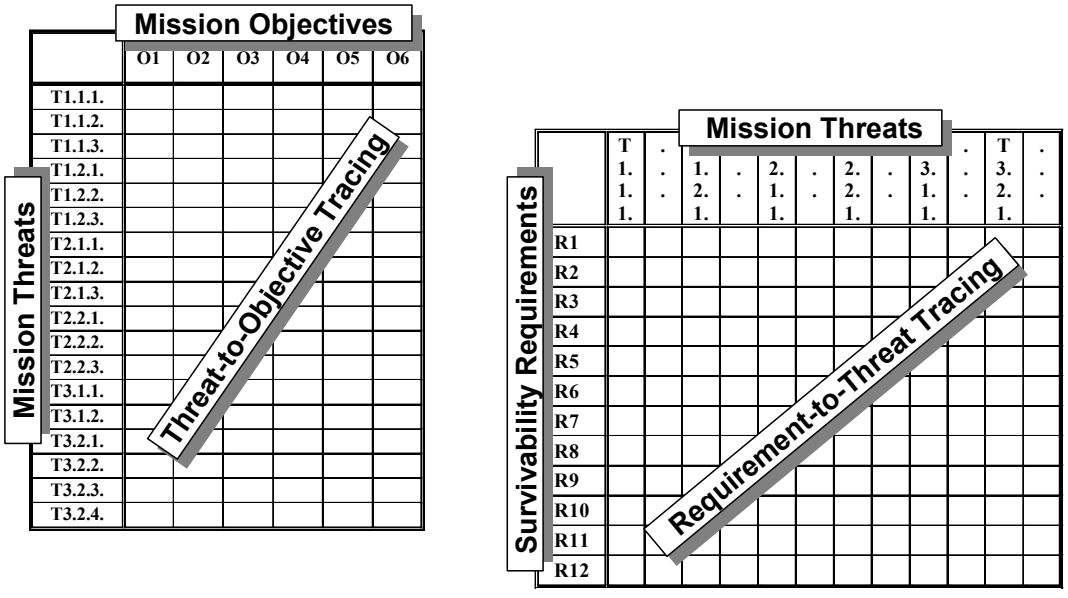
- The effects of changes to the architecture or threat environment on the mission can be assessed by following the tracing in a bottom-up manner.
- The effects of changes to the mission or threat environment on the architecture can be assessed by following the tracing in a top-down manner.

Figure 8: Survivability Tracing from Mission to Conceptual Architecture



Substantially new threats or new functionality in the architecture may require re-entering the survivability development process and reworking the tracing. In these cases, threat dynamics will provide valuable assistance in understanding the impact of the changes on the ability to accomplish the mission. The tracing can be conveniently documented in tabular format, as exemplified in Figure 9.

Figure 9: Example Survivability Tracing Tables



In summary, traceability helps decision-makers

- demonstrate that mission-critical requirements are satisfied
- identify the source and justification for requirements and design choices
- understand the impact of errors and failures on the system's ability to achieve its mission
- understand the impact of change due to the evolution of the organization's objectives, structures, behaviors, or threat environment

The next section describes in more detail the structure of the survivability strategy artifact documentation: mission objectives, mission threats, survivability requirements, and conceptual architecture.

3.2 Documentation Artifacts

The survivability strategy is a work in progress throughout the system development life cycle. It might start with the documentation of the identified threats and their impact on operations. A later version could add design guidance and survivability specifications for both the computing infrastructure and the supported applications. As a system nears deployment, the documented threat responses and their justifications support acceptance testing and certification.

Survivability analysis starts with a description of the expected operational environment and the desired computing support for the essential work processes, as might be found in a concept of operations. The operational environment description is refined throughout the system life cycle. The scope of the analysis depends on the nature of the operational environment: the complexity of system interaction, the distribution of work processes, the work-process dependencies, and the sharing of survivability risks and responsibilities among multiple organizations. The survivability strategy requires understanding the top-level operational properties (including the mission objectives and threats) that affect the survivability analysis and design choices. The survivability strategy documents the system's survivability requirements and conceptual architecture within the constraints set forth by the operational environment.

The rest of this section provides an overview of the primary documentation artifacts. Section 4 provides more information on these artifacts within the context of the survivability strategy development process.

Mission Objectives

Survivability requires that the mission objectives be explicitly documented and the system's achievement of those objectives be tracked both statically, during development and maintenance, and dynamically, during system operation. The mission objectives describe the high-

level purpose of the system in the eyes of the system owners and should answer the question of why the system is needed and what the system needs to accomplish. The objectives may be documented simply as a set of requirements, the formality of which is largely an issue of negotiation between the system owner and the developer. However, the mission objectives must make clear exactly what constitutes adequate support of the mission.

Deciding whether an organization has achieved its mission is not always straightforward [Ellison 99]:

> *Judgments as to whether or not a mission has been successfully fulfilled are typically made in the context of external conditions that may affect the achievement of that mission. For example, assume that a financial system shuts down for 12 hours during a period of widespread power outages caused by a hurricane. If the system preserves the integrity and confidentiality of its data and resumes its essential services after the period of environmental stress is over, the system can reasonably be judged to have fulfilled its mission. However, if the same system shuts down unexpectedly for 12 hours under normal conditions (or under relatively minor environmental stress) and deprives its users of essential financial services, the system can reasonably be judged to have failed its mission, even if data integrity and confidentiality are preserved.*

Exactly what constitutes an acceptable downtime for the above financial system needs to be specified as part of the mission objectives. However, detailing all the ways that threats may cause the financial system to crash is a subject of later analysis and documentation. Of course, this is true whether or not the threats are caused intentionally, which is why malicious threats to the mission must be analyzed and carefully documented as a part of TRIAD.

Mission Threats

The system operational environment is also the operational environment for the attacker, who can exploit vulnerabilities in the systems, in system administration, and in operations. The survivability strategy documents the characteristics of attacks that influence design. Those characteristics may include

- profiles for highest risk attackers
- operating environment vulnerabilities
- probable targets and strategies that meet attacker objectives
- required attacker actions, such as obtaining system privileges
- detailed intrusion scenarios in terms of the architecture of the implemented system and the supported operations

- intrusion scenarios that target vulnerabilities associated with the implemented system, including the computing infrastructure, system administration, and the supported operations

Survivability Requirements

The mission and operational environment generates survivability requirements in terms of expected system behavior and establishes constraints for the responses to attacks. For example, there may be limited system administrative resources available at some locations. Workflows involving multiple organizations could require interoperability among diverse security architectures rather than the use of a common infrastructure. The survivability strategy for a military mission with a short response time requirement could require the implementation of alternative operational actions that don't depend on the impacted systems.

Documentation of the survivability requirements may include

- desired operational and system response to the identified threat scenarios
- operational impact of attacks
- type of desired response: off-line recovery, reduced service
- constraints or requirements for
  - allocation of responsibility for the implementation of the strategy across multiple systems or organizations
  - allocation of the strategy in terms of people, systems, technology, and operations
  - allocation of the strategy in terms of weight given to resistance, recognition, recovery, and adaptation

Conceptual Architecture

The conceptual architecture must implement the survivability requirements in terms that the customer can assess and accept as an approach to meeting their needs. Documentation of the conceptual architecture may include

- *architectural views*—Architectural views may document the allocation of the threat response across the physical resources, or the execution flow of the response. Traditional architectural views include the component and connector view, which concentrates on the runtime behavior of the system, and the architectural resource view, which maps the components and connectors onto hardware [Clements 02].
- *general design assumptions*—Design assumptions may be documented as survivability requirements, including fault management responsibilities, for the supporting infrastructure, applications, or operations.
- *architecture tradeoffs*—Tradeoffs may involve functional properties of the system or non-functional attributes such as performance and maintainability.

Other associated documents may include disaster or attack response plans, training for system administrator and operations personnel, agreements defining shared responsibility and response among multiple organizations, and quality of service agreements.

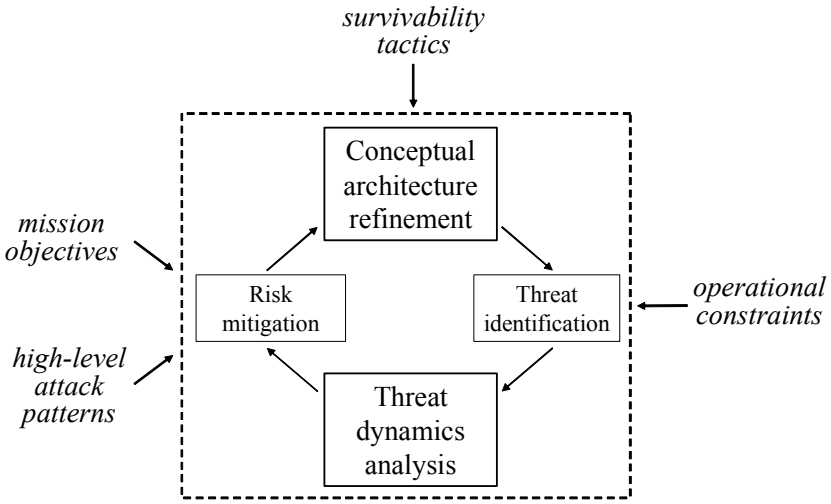
4 Survivability Strategy Development

The primary objective of the initial iterations of TRIAD is to formulate a coherent, justifiable, and affordable survivability strategy. Subsequent iterations use that strategy to guide system design and maintenance. TRIAD helps decision-makers identify architectural options, measure their effectiveness, and analyze their impact on operations and on system properties such as performance. The analyses performed as part of TRIAD support incremental design decisions, as well as justifications needed for acquisition, due diligence, and certification.

Figure 10 refines the process of developing a survivability strategy that was initially specified in Figure 4. Threat identification is a necessary precursor to evaluating the robustness of the conceptual architecture using threat dynamics. Risk mitigation is needed to translate the threat dynamics analysis into effective, strategic improvements to the conceptual architecture. This activity requires assessing the vulnerability of the conceptual architecture to mission failure and may require tradeoffs between different system quality attributes. For example, the threat identification activity may indicate a high likelihood of external and network-based denial of service attacks. Threat dynamics analysis would indicate the architectural vulnerability to such attacks and the tradeoffs with other mission objectives such as high performance and usability of Web services. Risk mitigation activities would propose approaches to lessen the impact of such attacks, e.g., network filtering, intruder trace-back, or increased network or server capacity.

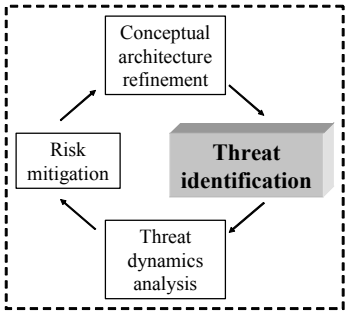
A by-product of this survivability strategy refinement process is a justification for the survivability of the system design, including the acceptance or rejection of design alternatives considered. The justification involves tracing from the mission objectives to the conceptual architecture that helps achieve those objectives despite active attack. The high-level attack patterns support the identification of possible attacks within the context of system operations. The application of survivability tactics suggests responses to those attacks. Of course, the conceptual architecture must ultimately fit within the operational constraints of the application domain.

Figure 10: Survivability Strategy Refinement Process



The rest of this section describes the activities of the survivability strategy refinement process: threat identification, threat dynamics analysis, risk mitigation, and conceptual architecture refinement. We start with threat identification, since this would be one of the first activities in the development of a survivability strategy for a system after the mission objectives have been characterized. Conceptual architecture refinement is characterized last, since it evolves as a result of the other three activities. As the process activities are presented, we describe how the survivability strategy is documented in terms of the mission objectives, mission threats, survivability requirements, and conceptual architecture. This should be helpful both for understanding the content of a survivability strategy and for developing a survivability strategy.

4.1 Threat Identification



The growing reliance of government and commercial organizations on large-scale, widely interconnected information systems amplifies the consequences of malicious attacks and compromises. In addition, the complexity and openness of these systems to the general public increases their exposure and vulnerability to malicious activity. The result is that increasingly sophisticated attacks are exploiting exposed vulnerabilities at an alarming rate. As seen by recent Internet worms and viruses released (e.g., Melissa, Love Letter, Code Red, Nimda), attackers share tools and knowledge to amplify their capability [CERT 02]. Each attack method builds off the knowledge, experience, and code of the previous attack method, which ironically makes the attack (virus, worm, etc.) more survivable as a result.

Increasingly sophisticated attacker tools permit relatively inexperienced individuals to execute very advanced attacks.

In addition, we have seen such attacks escalate with the intensity of political conflicts, such as the war in Kosovo, the tensions between the U.S. and China, and the conflict between India and Pakistan [Vatis 01]. While these attacks are often in the form of embarrassing Web site defacements, attackers are starting to surreptitiously target the perceptions of users, such as the attempts to modify the content of major new publications or company press releases [Cybenko 02]. With the ongoing war on terrorism, we are only likely to see more cyber-deception attacks to undermine military mission survivability. A recent report analyzing the possibilities for cyber-terrorism concludes that "a semantic attack on a news site or government agency site, causing its Web servers to provide false information at a critical juncture in the war on terrorism, could have a significant impact on the American population" [Vatis 01]. In short, attacks by individuals more sophisticated than the average recreational hacker (e.g., industrial spies and international cyber-terrorists) are more likely and are more difficult to counter.

A broad, but not uncommon, view of threat includes the potential harmful results due to malicious attack, user errors/lapses, technological faults, and natural disasters. Traditional reliability analysis often deals with a static list of faults with known failure rates. The analysis in that context can lead to an accurate assessment of the cost-benefit of preventive strategies such as replicated storage. Survivability, in contrast, has to manage a non-static list of maliciously generated, and often very rare, faults. Our current efforts limit the scope of this analysis to malicious attack, since threats due to unintentional acts, faults, or accidents are random events that can be analyzed with existing dependability and fault tolerance techniques. Malicious attacks, however, often involve the worst possible set of contrived inputs or actions delivered at the most inopportune time, resulting in mission failure. In addition, the threat environment is extremely dynamic; attacks two years from now are likely to use entirely new tools to exploit previously undiscovered vulnerabilities.

Attacker Characterization

Identifying the threats that are relevant to an organization's operations involves characterizing the types of attackers that are likely to threaten the organization's mission and the types of attacks that those attackers are likely to carry out. Attackers can broadly be characterized according to a number of attributes:

- **Resources -** Resources include funds, personnel, and the skill levels of those personnel.
- **Time -** An attacker may have very-near-term objectives or may be very patient and wait for an opportunity.
- **Tools -** The sophisticated attacker can tailor attack tools to change their signature to avoid detection, or can develop tools or an email virus to target a specific system.

- **Risk -** An attacker may seek publicity. An attacker operating outside of the United States may not be threatened by legal actions.
- **Access -** Intruder access can be described in terms of
  - the access mechanisms used in the attack, such as a dialup modem, a digital subscriber line (DSL), or the Internet
  - the origination of the attack, such as outside of a firewall, on a LAN, or connected from a trusted site
  - the organizational position of the attacker, if any, such as employee, system administrator, or contractor
- **Objectives -** An attacker's objectives may include political, financial, criminal, military, and personal motivations.

Characterizing specific types of attackers is beyond the scope of this report. There is a plethora of books that describe the attributes and techniques of fairly unsophisticated, but malicious, individuals often called hackers, or crackers. The characterization of more sophisticated attackers, such as industrial spies and international cyber-terrorists, is usually sensitive and sometimes classified [OPSEC 00].

Attack Characterization

Individual attacks can be broadly classified as to whether they are people based, technology based, or context based. These attack classes, respectively, target

- people's wants, needs, capabilities, or perceptions. Examples include social engineering, semantic attacks, extortion, and physical harm. Such attacks can exploit greed, fear, or gullibility; corrupt morals; or incapacitate essential personnel.
- computing and networking technology. Examples include
  - network-based attacks: attacks on communication infrastructure and supporting services. Examples include network-based denial of service attacks, including distributed denial of service.
  - application-based attacks: attacks on the architecture component applications such as a Web server, email services, or supporting application infrastructure. Examples include exploits that target vulnerabilities of a Web server, such as a buffer overflow vulnerability, to gain increased access.
  - data-centered attacks: attacks on the data stream or content presented by transactions. Such attack patterns can exploit or corrupt data and services or disrupt or deny essential services. Examples include attacks that target trust relationships between different machines, or that target the gullibility of users (such as email attachments that contain malicious code).
- the context in which people perform their jobs. Examples include attacks on work support, customer demand, the value of corporate stocks, or legal constraints under which

people and corporations work. Such attacks can exploit or deny critical resources or damage corporate market, capability, or assets.

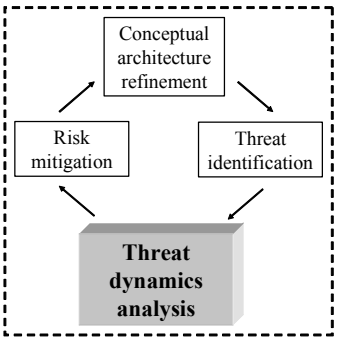
Intrusion scenarios involve interactions from the adversary's view, a negative view with respect to system functionality, rather than a legitimate user's normal view, a positive view. We define an intrusion scenario as a description of people interacting with systems in a malicious way, thereby intentionally causing harm to an organization.

An intrusion scenario can be represented as a sequence of attacks that leads to a specific compromise of the system's mission. An attack may or may not be completely successful, but it always changes the state of the system in some way. An intrusion, on the other hand, always leads to a specific mission compromise through the execution of the sequence of at least partially successful attacks. Related intrusions can be conveniently organized into attack trees where the root of the tree describes the mission compromise to which the intrusions contribute [Moore 01a]. However, attack trees are of limited use in formulating an overall survivability strategy, since at least a high-level architecture must already exist to develop an attack tree. In addition, any changes made to the architecture resulting from attack tree analysis lead to, at best, an incremental improvement of the architecture. Such changes help if the architecture is of sufficiently high quality, but do little for an architecture that is way off track. Nevertheless, attack trees, and the intrusion scenarios that they generate, provide an incremental approach to formulate a low-level design, and ultimately an implementation, that is robust against likely attacks.

A technique related to our use of intrusion scenarios, called *misuse cases* or *abuse cases*, leverages the use case concept of the Unified Modeling Language™ (UML) for information security [McDermott 99, Sindre 00]. The most common view is that a use case is a general specification of a set of related concrete usage scenarios. Abuse cases are to use cases as intrusion scenarios are to usage scenarios, i.e., they take an adversary's view rather than a user's view. Therefore, we can view abuse cases as a standard way to describe a set of related intrusion scenarios. UML explicitly identifies actors in a use case diagram and shows how these actors interact with the system. Attackers in an abuse case diagram correspond to the actors in a use case diagram. Abuse cases describe these malicious actors in detail according to their resources, skills, and objectives.

™ Unified Modeling Language is a trademark of Rational Software Corporation.

4.2 Threat Dynamics Analysis



While there is significant variation in the details of specific attacks, it is the common aspects of attacks that provide the most insight in directing survivable system development. For example, many attacks share requirements to identify user accounts or to sketch the topology of the network that supports the workflow. Attacks can be categorized in terms of the kind of access and privileges required to execute the attack: user privileges are typically required for access to protected application or data; system privileges are usually required to compromise logs to disrupt forensics; and network access is required to probe the network to identify available and vulnerable services.

In addition, while vulnerabilities are often thought of in terms of the flaws of low-level components, vulnerabilities at an architectural level may be a much higher threat to an organization's mission. In general, vulnerabilities may be apparent in human operations, the architecture of the technology, or individual technical components. Table 1 provides several examples of how attacks can exploit gross vulnerabilities at an architectural level.

Table 1: Increased Threat Due to Architectural Vulnerability

Vulnerability	Impact
Distributed system administration in terms of sites or in terms of applications, servers, and networks	Detection and recovery are difficult to coordinate. An attacker can exploit confusion or poorly defined areas of responsibilities.
Multiple applications on a LAN, each with an external user community	An attacker can successfully gain access via an application exploit and then use trust shared among applications to attack other services.
Shared infrastructure	Compromised infrastructure can impact multiple applications and sites.
Workflow that crosses multiple administrative domains	Local administrative errors can be exploited. Local attacker activity may not be observable to the targeted system.

The primary objective of threat dynamics is to develop and demonstrate methods to determine effective strategic responses to the actual threats to large-scale, inter-networked information systems. Threat dynamics enables decision-makers to assess the impact of a potentially evolving threat environment on the system and its overall mission. Threat dynamics modeling provides a holistic view of the general influences that the threat environment can have on the ability of the system to fulfill its mission. This big picture view permits analyzing

dynamically the effects of changes in attacker activity, system operational responses to attacker activity, changes in system operations or architecture, or the availability of new data that characterizes perceived threats in a new light. Threat dynamics analysis clarifies the role that technology has in accomplishing the larger organizational mission.

System Dynamics Background

System dynamics was developed by Jay Forrester to show how a model of the structure of a human activity system and the policies used to control it could be used to deepen our understanding of the operation and behavior of that system [Forrester 61]. System dynamics has been used extensively as a general modeling tool to enable better understanding of the structure and dynamics of complex human-based systems, particularly in the area of business strategy and public policy [Sterman 00, Wolstenholme 90].

System dynamics can be defined as a method to model and analyze the holistic behavior of complex, managed systems as they evolve over time. Managed systems include any system that people control, or try to control, in some way. The goal of system dynamics is to understand how information feedback governs system behavior and to design feedback structures and control policies that improve the management and operation of the system. Coyle defines system dynamics in terms of the well-established field of control engineering as "the application of the attitude of a control engineer to the improvement of the dynamic behavior in managed systems" [Coyle 96]. Whereas control engineers design mechanical systems such as central heating systems or auto-piloting for aircraft, system dynamics engineers design policy controls for human-based systems such as the criminal justice system or national security. System dynamics is, in fact, grounded in the theories of nonlinear dynamics and feedback control known for many years by mathematicians, physicists, and engineers.

System dynamics scopes the term system broadly to include any collection of interacting elements that are organized for a purpose. System dynamics is particularly useful for modeling and analyzing systems with high dynamic complexity. Static (or combinatorial) complexity arises when trying to make an optimal choice among an overwhelming number of possibilities, as might be seen when scheduling a major airline's flights and crews. In contrast, dynamic complexity arises from the nature of the interactions among the system elements over time, especially the speed and intensity of those interactions. Information feedback, time delays, non-linearity, uncertainty, and volatility of behavioral responses to stimuli all complicate our understanding of how dynamic systems behave, especially over the long term.

The simplest form of qualitative problem description and analysis in system dynamics is the influence diagram. Figure 11 shows two very simple influence diagrams, one representing a central heating system and the other representing the inherent effect of the birth rate on population growth. Diagram variables represent the system elements involved. System elements may be animate or inanimate, tangible or intangible. Elements shown in italics are, for the

purposes of the ongoing analysis, constant factors (or parameters) that act as inputs to the calculation of system variables. Signed arrows represent the system interactions, where the sign indicates the pair-wise influence of the variable at the source of the arrow on the variable at the target of the arrow:

- A positive (+) influence indicates that if the value of the source variable increases, then the value of the target variable increases above what it would otherwise have been, all other things being equal. And, if the value of the source variable decreases, then the value of the target variable decreases below what it would otherwise have been, all other things being equal. So in the first influence diagram in Figure 11, at a particular thermostat setting, as the rate of heat input increases (decreases), then the temperature of the room increases (decreases) above (below) what it would have been.
- A negative (-) influence indicates that if the value of the source variable increases, then the value of the target variable decreases below what it would otherwise have been, all other things being equal. And, if the value of the source variable decreases, then the value of the target variable increases above what it would otherwise have been, all other things being equal. So as the room temperature increases (decreases), the rate of heat input decreases (increases) below (above) what it would have been, as would be expected by a central heating system.

Figure 11: Simple Influence Diagrams

Two key drivers of dynamic behavior are feedback loops and time delays. Feedback loops can be self-reinforcing (+) or self-limiting (-). The polarity of a feedback loop is determined by "multiplying" the signs along the path of the loop. The central heating system of Figure 11 is self-limiting since it has an odd number of negative signs along its path. Self-limiting loops describe aspects of a system that tend to drive variable values to some goal state. In the case of the central heating system, the goal state is a room temperature equal to the thermostat setting. In general, self-limiting loops describe aspects that oppose change, and usually involve self-regulation through adaptation to external influences. Of course, these aspects may or may not be desirable. For example, recent studies show that lowering the nicotine in cigarettes, supposedly to the benefit of smoker's health, only results in people smoking more cigarettes and taking longer, deeper drags to meet their nicotine needs. An example of a beneficial self-limiting loop is the use of an active network defense, which recognizes and recovers from malicious attacks on the network to maintain a desired level of security or survivability.

Self-reinforcing loops describe system aspects that tend to drive variable values consistently upward or consistently downward. The second influence diagram of Figure 11 shows a loop that is self-reinforcing due to the even number of negative signs along its path, where zero is considered even for this purpose. Self-reinforcing loops may help explain explosive growth or implosive collapse of a system. For example, the nuclear arms race was an example of self-reinforcing feedback whereby the U.S.S.R. built nuclear arms to counter the nuclear threat posed by the U.S. This spurred the iterative increase of U.S. nuclear weapon stockpiles followed by even more by the Soviets, resulting in the explosive nuclear arms build-up. Microsoft and Intel benefited from explosive growth by being dominant market forces early in the rise of the personal computer, which motivated software vendors to target the Windows/Intel platform, thus furthering the companies' dominance. People's Express Airlines showed similar explosive growth in the 1980s, due in part to low prices and no frills customer-oriented service. This rise was followed by a similarly dramatic collapse when major airlines started offering low-cost fares to attract non-business travelers. Self-reinforcing loops may also help explain the rise and fall of self-replicating computer worms, such as the Code Red and Nimda worms that caused many problems during the summer of 2001, but on much more compressed time scales than the previous examples.

Time delays can make the dynamic behavior of systems seem erratic. Such delays can separate cause and effect in a way that makes the long-term effect very different from the short-term effect. Considering time delays in feedback loops, or the interaction of multiple feedback loops, helps to explain what seems like counter-intuitive behavior. Time delays in self-limiting loops can create instability and oscillation such as is observed in stop-and-go traffic or getting a shower to deliver water of the appropriate temperature. One feedback loop can amplify or moderate the influence of another feedback loop. When too narrow a view of the system is taken, the analyst only sees part of the whole picture, making perfectly explainable behavior seem erratic or unpredictable.

Threat Dynamics for Survivability

Preliminary literature searches have yielded very little published work that applies system dynamics to study the effectiveness of information technology. One of the few works available describes an approach that uses system dynamics to study the impact that introducing a management information system has on an organization's mission [Wolstenholme 93]. Wolstenholme develops two case studies to evaluate the operational impact of a military logistics system and a battlefield, tactical command and control system. He argues that the approach "has a great deal to offer in the design phases of Management Information Systems, and the fuzzy (often iterative) boundary between design and assessment. The ability of the technique to incorporate subjective data in these phases is particularly advantageous" [Wolstenholme 93]. Wolstenholme's work is related to our effort, providing some evidence for its overall value and feasibility. However, we are not aware of any work using system dynamics to explicitly study the threat environment or its impact on system operations.

Nevertheless, we believe that system dynamics provides a foundation for developing methods and tools that help engineers understand, characterize, and communicate the impact of a malicious threat environment on organizational and system operations and their respective missions. Large-scale, inter-networked information systems are subject to volatility, non-linearity, uncertainty, and time delays that add to their dynamic complexity and make assuring their security or survivability so difficult.

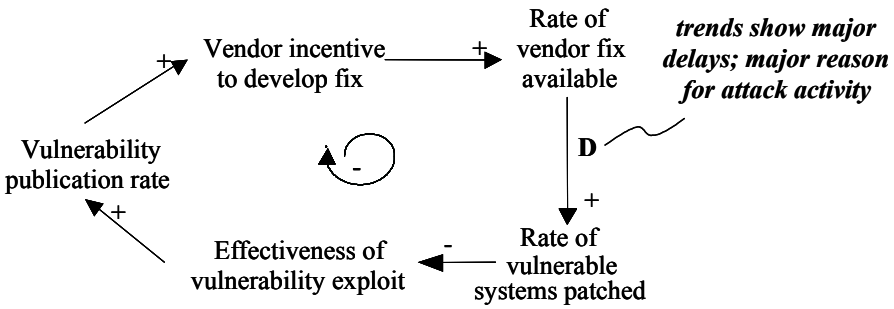
- Volatility: The increasingly rapid development of attacker tools and coordination of the attacker community promotes a very volatile threat environment for business and military information systems [CERT 02]. Ensuring the security and survivability of such systems demands techniques that can be applied as part of an evolutionary design and maintenance life cycle.
- Non-linearity: The organizational mission may be drastically more vulnerable due to only small increases in attacker capability or small changes in system policies, controls, or architecture. Such non-linearity makes maintaining the security and survivability of even relatively simple networked systems very difficult given the volatility of the threat environment and the nature of active network defense.
- Uncertainty: Although accurate incident and vulnerability data is becoming more readily available, there are still large gaps in our understanding of intruder behavior, creating a fair amount of uncertainty. Threat dynamics analysis benefits from the availability of such data where it exists, but does not depend on it in order to provide useful insights into the impact of the threat environment on system operations. Threat dynamics analysis, and its system dynamics basis, can be performed in a qualitative, a quantitative, or combined manner [Coyle 00].
- Time Delay: Major delays often exist between the time that an attacker engages in malicious activity and the time that we understand the full extent of that activity. Such delays make implementing strategic countermeasures and assessing their effectiveness very difficult, especially when real-time reconfiguration may be necessary to thwart the adversary.

While system dynamics is widely applicable, it is most useful in systems that use derived information to exert feedback control over its resources. Such feedback control is a critical technique for building survivable information systems. Active defense approaches monitor attack activity and respond through a variety of recovery and adaptation techniques to ensure mission success. Thus, survivable systems control their information resources based partly on feedback from the attack-monitoring activity. System dynamics helps represent and analyze such feedback control but has generally not assumed the presence of hostile agents.

Figure 12 illustrates some of the system dynamics concepts described in the context of threats to Internet-based systems. The figure depicts a feedback loop that describes an aspect of the behavior to control the vulnerability of Internet-based systems [Arbaugh 00]. Starting at the "Effectiveness of vulnerability exploit" element at the lower left-hand side of the figure, we

see that effectiveness positively influences the rate of publication about the vulnerability, in the sense that an increase in effectiveness leads to an increase in publication (perhaps due to increased media attention), with all other things being equal.[4] Likewise, increased publication leads to increased incentive to fix, and the ultimate availability of relevant patches. This leads to patching of systems, which in turn reduces the effectiveness of the vulnerability exploit. The delay in patching, signified by the "**D**" along the arrow on the right side of the figure, is a trend that has been described as a major reason for heavy Internet-based attack activity, and the general vulnerability of the Internet. Nevertheless, the overall feedback loop described is a balancing one (indicated by the negative loop symbol in the center), in that patching generally helps to control overall Internet vulnerability.

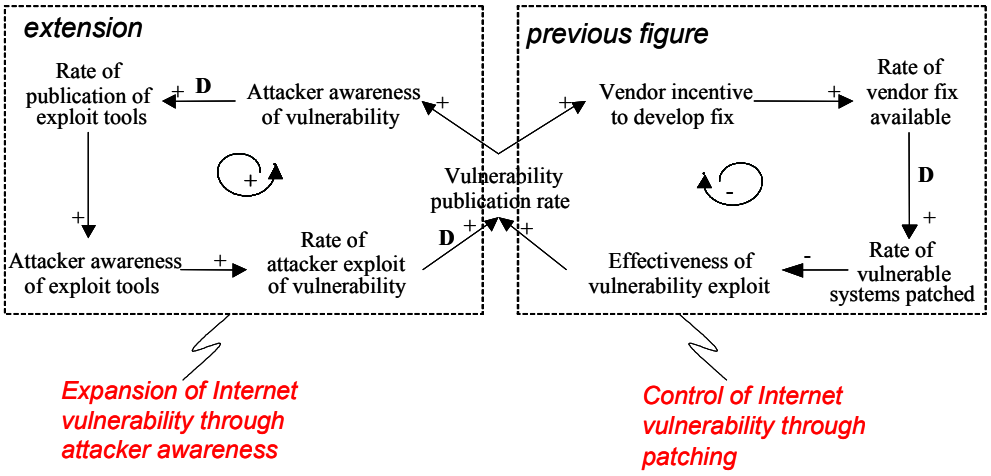
Figure 12: A Feedback Loop for Controlling Vulnerability



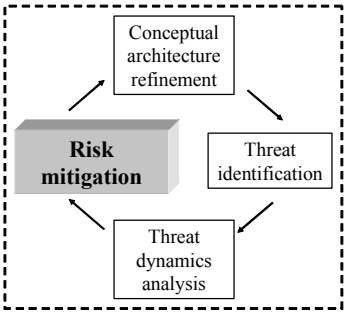
Influence diagrams can be composed as illustrated in Figure 13. The right side of the figure shows the influence diagram described above. The left side shows a feedback loop that describes an effect of the vulnerability publication rate on the publication of exploit tools and, ultimately, on the attacker exploit of the vulnerability. This is an example of a positively reinforcing feedback loop, as indicated by the positive loop symbol in the center. This figure illustrates an ongoing debate in the Internet community about whether publishing vulnerability data helps or hinders the overall security of the Internet. Recent analysis indicates that delays in patching are the primary cause of Internet vulnerability, while the publication of vulnerability data is a secondary driving force [Arbaugh 00]. The diagram does not, of course, help resolve the debate, since it is strictly qualitative in nature.

[4] The phrase "all other things being equal" should always be assumed when thinking about pair-wise influences in an influence diagram. For simplification, we omit this phrase from future descriptions.

Figure 13: The Effects of Vulnerability Publication on Internet Vulnerability



4.3 Risk Mitigation



Threat dynamics helps map out the influences of attacks and countermeasures on the ability of the system to achieve the organizational mission. The influences indicate potential softspots in the conceptual architecture and tradeoffs that must be made toward achieving an effective and affordable solution. Solutions for shoring up these softspots and for making decisions regarding the tradeoffs may be passive or active in nature. Preventive techniques such as authentication, authorization (access control), and encryption increase the resistance to attacks and are typically *passive*. The system response to an attack in terms of recovery of services or continued (possibly degraded) operations may be to change the system configuration or security policy, for example to strengthen authentication or tighten access control. The responses in this context are *active* in the sense that the system must detect the attack and then actively reconfigure system operations to ensure mission success in the presence of the attack. The system architecture has to support continued service in the presence of attacks through redundant services or data recovery services so as to restore full service following an attack. The selection and weights given to passive and active approaches are critical to the survivability strategy.

Limitations of Existing Technology

A solely reactive approach to building and maintaining system security and survivability is doomed to failure because of practical limits on shrinking the window of exposure of vulnerable systems [Arbaugh 00, Schneier 00b]. In fact, no amount of hardening can ensure that intrusions on unbounded systems will not occur. The detection of and recovery from successful attacks are an essential part of system survivability.

Unfortunately, intrusion detection technology can address only a small part of the problem, at least in its current form. Existing intrusion detection technology targets the identification of only computer- and network-based attacks. Attacks that "fly over the radar" of intrusion detection technology, such as social engineering and physical attacks, need to be taken as seriously as technological attacks [Anderson 01]. Intrusion detection technology does not identify the correlated activity of what are actually multi-stage attacks—attacks that may involve coercion, corruption, or deception of people in addition to the exploitation of technological vulnerabilities. In general, attacks can target a system's internal users and components, as well as external trusted systems and user communities. Disregarding the human factor could be very misleading and result in large gaps in our system defenses.

Another problem with existing intrusion detection technology is the high rate of false positives (detecting an attack when there is none) and false negatives (not detecting an attack that really took place). False positives require a human analyst to go through the audit logs to identify whether an intrusion actually took place. While such analysis may be a necessary part of a survivable system design, intrusion detection technology that has high false positive rates may be an unnecessary burden to administrators and actually be a detriment to survivability of the mission in the long run.

False negatives are perhaps even more pernicious than false positives. CERT/CC analysts are seeing increasingly stealthy attacks that "fly under the radar" of existing intrusion detection technology. A single probe executed once per day may allow a patient adversary to map out an organization's network just as effectively as broad scans, and without being detected. In addition, most of the attack patterns on which intrusion detection technology is based do not represent the correlated activity of a capable attacker but merely some intermediate point of an attack that is often unfocused and perpetrated by a relatively unskilled novice. These patterns do not completely nor accurately represent the behavior of sophisticated and motivated attackers and, therefore, are not an adequate basis for identifying the threat that they pose or detecting the attacks that they perpetrate.

Existing intrusion detection technology promotes the same bottom-up approach to survivability that was discussed in the introduction to this paper. Without a view to the larger mission, an organization may waste much time and resources attempting to detect and analyze attacks that have no impact on their ability to succeed. That said, intrusion detection will likely be an increasingly important part of building survivable systems. A recent report on the state of the practice of intrusion detection technologies recommends that, among other things, future technologies should integrate a more diverse source of attack data to ameliorate inaccuracies, defend against attacks that are more sophisticated than those of the average hacker, and integrate human analysis as part of event diagnosis [Allen 00]. We agree with these recommendations, but suggest taking them a step further to deal directly with the inherent limitations of a strictly technological approach. Organizations should focus on intrusion detection and re-

sponse holistically by integrating a comprehensive intrusion detection and response capability with an organization's policies and procedures, as well as with the technology.

Survivability Tactics

The use of survivability tactics in TRIAD derives from the notion of an *architectural tactic* developed at the SEI [Bachmann 02]. The objective of this work is to describe how quality attributes such as performance or modifiability exert influence over architectural design, how these influences can be codified, and how these notions can be used to analyze architectures. They define an architectural tactic as "a design decision that helps achieve a specific quality-attribute response and that is motivated by a quality-attribute analysis model." For performance and latency, such design decisions include the size and number of servers and the management of concurrency on the server with the analysis supported by queuing and scheduling models. Modifiability tactics include the management of public and private information for a module to localize expected modifications. The associated analysis examines the dependencies among systems, the probability of changes, and the impact of making those changes.

Survivability tactics, a particular class of architectural tactics, codify design decisions that help mitigate the risk associated with a malicious threat environment. While there is significant variation in the details of specific attacks, it is the common attributes of attacks that provide the most insight in directing survivable system development. Survivability tactics describe how particular design decisions mitigate the risk associated with all attacks that share these common attributes. For example, a popular attack technique is to exploit a vulnerability on an Internet-accessible server such as an FTP or Web host and then use the increased access obtained from that exploit to attack related systems. This technique corresponds to the following survivability tactic: the use of firewalls in a demilitarized zone (DMZ) configuration to limit the attacker's access to other systems after the initial penetration.

Survivability tactics are also useful for specifying intrusion recovery schemes. System and application logs have to support the analysis that follows an attack to identify the scope and impact and generate the detailed recovery plan. Survivability tactics that target the computing infrastructure, such as replicated services, improve recovery but may increase system administrative costs. For work processes that involve multiple locations or organizations, a recovery tactic can be to restore services locally and then synchronize the collection of systems later. Notice that a survivability tactic may help address new, never-seen-before attacks, if those new attacks share the common attributes addressed by that tactic. Table 2 describes other example survivability tactics.

Table 2: Survivability Tactics Addressing Types of Attacks

Attack Type	Attacker Strategy	Survivability Tactics
Denial of Service	Target server or network. Compromise the operations of an infrastructure service such as a directory server or the network management console, which impacts a wide range of computing services. Denial of service attacks do not necessarily require user access to the system.	Network architecture: packet filter, intruder trace-back, spare capacity, distributed services Infrastructure architecture: replication, accelerated recovery Application architecture: service proxied to monitor content
Compromise of Application Content	Target applications or data management services. Examples: email virus, use social engineering to induce employee to enter invalid data, a successful attack on a trusted site inserts compromised information into the data stream.	Application: virus filtering and scanning, repeated tests for data integrity and consistency even for data from trusted sites, monitor data access and block suspicious activity Personnel: training

Many of the above survivability tactics have been useful in the analysis of the survivability of real-world systems, as applied using the Survivable System Analysis Method (SSA) [Mead 00]. Of particular concern in previous applications of the SSA has been the allocation of mission-preservation responsibilities across multiple organizations. Responses to attacks involve a combination of resistance, recognition, and reaction to events. Some responses depend on immediate operational changes with the concerned organization, while others use a specific architecture or technology to limit the impact of the attack on operations. Deciding on the optimal response depends on a variety of factors, including

- operational fault sensitivities: Do essential operations have a near-real time response requirement? What is the impact of limited access to data? For a distributed workflow, what local processing can continue if the network is compromised?
- operational impact for classes of responses: Response options include continuing operations with recovery taking place in the background; reducing service in terms of the user community or supported functionality; continuing operations locally and eventually synchronizing; and removing the system from operation until recovery is complete and the threat contained.
- operational constraints: Constraints may include personnel skills for operations and system administration and limitations imposed by legacy systems, contractual agreements, or limited authority.

Attacks often exploit errors in system component software. An exploitable error is often called a system vulnerability. The execution of the attack generates the fault associated with the error. Consequently, many of the survivability tactics involve fault management. A com-

mon objective for fault management is to hide network and hardware faults from the applications. But attacks can generate rare combinations of independent faults or exploit the tight integration among components. Survivability tactics may require some faults to be visible to applications. For example, a coordinated attack on a military tactical network could require that the applications that use that network adjust their behavior if the available network bandwidth has been reduced. The application of correct and safe system configuration changes requires accurate and timely system status information. The *self-repair* administrative components have to monitor network and sensor faults that could impact the delivery and integrity of the system status information and potentially limit the reconfigurations when there is low confidence in the data.

The architectural placement of the responsibility for attack recognition and response depends on the kind of faults generated by the class of attacks of interest. Network management and network-based intrusion detection target network denial of service attacks as well as attacks that exploit the IP protocol. Host-based intrusion detection systems concentrate on attacks that exploit system or server vulnerabilities. The detection of and response to attacks that target the data content in an exchange could be the responsibility of applications that understand the semantics of the transaction.

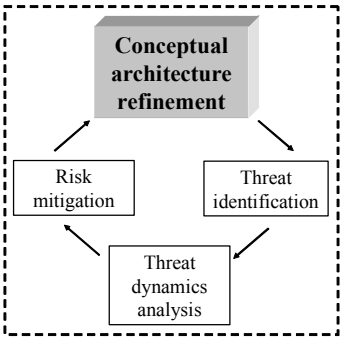
Experience with real attacks on systems through the years emphasizes the need to consider the big picture, including both the technology and its operational environment, in order to develop strong and cost-effective solutions [Anderson 01, Schneier 00a]. The following general techniques are useful in developing survivability tactics, either individually or in combination. These techniques may be implemented manually (through human procedures), automatically (through technology), or through a combination of the two.

- *Redundancy* - Anderson defines redundancy as "maintaining a depth of spare components or duplicated information to replace damaged or compromised assets" [Anderson 99]. Replicating components, connections, and/or data, often not co-located with the original copy, combined with good replication management can allow continued service when the original copy fails or is compromised.
- *Diversity* - Diversity involves the use of different methods, components, and/or platforms to prevent attackers from exploiting the same vulnerabilities repeatedly. Examples include the use of different hardware, different operating systems, or even different programming techniques such as n-version programming. When such diversity is used at different system entry points it can increase the attacker work factor.
- *Deception* - Deception can be used by the defender as well as a survivability threat that can be used by an adversary. Anderson defines deception, as it pertains to ensuring survivability, as an "artifice aimed at inducing enemy behaviors that may be exploited" [Anderson 99]. The most common example is the use of a collection of misinformation to waste an attacker's time as other mechanisms mount an appropriate response to the attack. This misinformation is often euphemistically referred to as a honeypot.

- *Identification/Authentication* - NSA defines authentication as the verification of a claimed identity as legitimate and belonging to the claimant [NCSC 91]. Most types of access control require accurate identification and authentication of users. The most common technique used by far is username/password. Stronger techniques using biometrics, tokens, and cryptographic signatures are also possible.
- *Intrusion Detection* - Intrusions require "both an overt act by an attacker and a manifestation, observable by the intended victim, that results from that act" [McHugh 00]. The goal of intrusion detection is to observe and report on the manifestation of an attacker's intrusion. Reporting can occur manually, through human analysis, or automatically using intrusion detection systems. Intrusion detection can take place in real time or through the off-line analysis of system activity audit data recorded separately. Intrusions may be detected by looking for signatures of known attacks (virus checking is a common example) or by looking for anomalies—system activity that does not fit "normal" usage patterns. Available intrusion detection systems target either low-level network traffic or higher level application usage [McHugh 00]. Integrity-checking system executables based on expected parameters is also a form of intrusion detection.
- *Recovery/Adaptation* - Recovery and adaptation are the system responses to intrusion detection. Recovery is typically the near-term repair or replacement of data, components, or communications damaged due to intrusion. Adaptation typically involves longer-term planning and reconfiguration to prevent similar intrusions in the future. Examples range from complex techniques such as dynamic resource allocation to high-priority assets and activities and self-organization of distributed autonomous agents [Anderson 99] to simple techniques such as restoration from stored backups and error correction.
- *Physical, Logical, Cryptographic, and Temporal Separation* - The security community has long regarded separation as fundamental to providing information security. Rushby and Randell first introduced these four primary types of security-related separation [Rushby 83], primarily as a means to separate entities of different classification levels. These strategies have also proven useful for information integrity and availability. The oldest strategy, physical separation, promotes security using spatial distribution and physical security mechanisms [NCSC 88], such as reinforced buildings, locks, and various types of shielding. Logical separation uses software-based mechanisms, such as message filters, functional wrappers, and security kernels, to control access. Cryptographic separation uses encryption and key management to protect data confidentiality and to detect data corruption to a degree proportional to the strength of the cryptographic algorithm and of the protection of private keys. Finally, temporal separation separates critical functions' execution in time. It is most closely associated with periods processing, "the processing of various levels of sensitive information at distinctly different times" [NCSC 88].
- *Personnel Management* - The survivability of any mission depends greatly on the trustworthiness, knowledge, and capability of the people in charge of mission support or execution. Trustworthiness is typically assessed through personnel security procedures [NCSC 88] such as periodic investigation of the backgrounds of people who have mis-

sion responsibilities. Ongoing performance appraisals are often a necessary complement to such investigations and provide additional information in terms of an individual's understanding and ability to perform the job adequately. Periodic training is also important to educate people on the role their jobs play in successfully achieving mission goals, the importance of security policy and procedures, and the possible impacts of inadequate performance.

4.4 Conceptual Architecture Refinement



A system's conceptual architecture evolves, as described above, through threat analysis and risk mitigation using survivability tactics. Throughout this evolution, TRIAD requires documentation of the system's survivability requirements. Our technique for specifying survivability requirements derives from the use of scenarios for requirements description and analysis [Weidenhaupt 98].[5] Potts describes scenarios as a sample execution of a system, the antithesis of specifications:

> *Whereas a specification describes behavior generally, a scenario exemplifies behavior by presenting specific, concrete episodes. Whereas it is possible to deduce scenarios from a specification, it is possible only to induce a specification from a collection of scenarios [Potts 95].*

System requirements for both functional requirements and non-functional requirements can be defined as scenarios. Non-functional requirements are requirements about quality attributes such as performance, usability, and maintainability. Bass describes an approach to characterize a quality attribute as a set of *general scenarios* [Bass 01]. A general scenario is described in terms of a stimulus and a response measure. For example,

- A modifiability general scenario is spurred by *changes arriving* and results in *their propagation through the system specification and implementation*. Modifiability general scenarios reflect the various classes of change possible.
- A performance general scenario is spurred by *events arriving* and results in *a response to the event with some latency*. Performance general scenarios reflect the various classes of performance response required.

Bass proposes that a collection of such system-independent scenarios can serve to completely characterize a quality attribute. Furthermore, specializations of general scenarios, called *specific scenarios*, can be used to describe system-dependent, non-functional requirements. The notion that a quality attribute can be characterized as a set of scenarios has clear relevance to the design of survivable systems. A general survivability scenario is spurred by *attacks perpe-*

[5] In UML terminology, scenarios are called use cases [Jacobson 99].

trated and results in *resistance, recognition, recovery, and adaptation so as to continue to provide essential services*. Specific survivability scenarios reflect the various classes of requirements to resist, recognize, recover from, and adapt to attacks.

Survivability requirement elicitation has to represent those with responsibility for the mission, the eventual system users, and those with responsibility for execution of the response. Survivability requirements can conveniently be organized into a table as exemplified in Table 3. The table illustrates a framework for specifying the various ways that a system is required to respond to an attack stimulus. An attack class to be addressed by the system may be broken into a number of subclasses. The table shows how to specify responses as a combination of resistance, recognition, recovery, and adaptation techniques. As shown, an individual response may address multiple subclasses of attacks. Also, an individual subclass of attacks may be addressed by multiple responses, even within the same class of response techniques, as shown for resistance to attack subclass #1. This provides a capability to specify defense-in-depth against particular attacks. Section 5 provides specific examples using this tabular format for specifying survivability requirements.

Table 3: Tabular Format for Survivability Requirements

<table>
<tr><th colspan="2" rowspan="2">Stimulus</th><th colspan="5">Response</th></tr>
<tr><th colspan="2">Resistance</th><th>Recognition</th><th colspan="2">Recovery</th><th>Adaptation</th></tr>
<tr><td rowspan="2">Primary class of attack</td><td>Subclass #1 of primary attack class</td><td>First technique to resist attacks in subclass #1</td><td>Second technique to resist attacks in subclass #1</td><td rowspan="2">Technique to recognize attacks in both subclass #1 and subclass #2</td><td rowspan="2">Technique to recover from attacks in both subclass #1 and subclass #2</td><td>Additional technique to recover from attacks in subclass #1</td><td>Technique to adapt to attacks in subclass #1</td></tr>
<tr><td>Subclass #2 of primary attack class</td><td colspan="2">Technique to resist attacks in subclass #2</td><td>Additional technique to recover from attacks in subclass #2</td><td>Technique to adapt to attacks in subclass #2</td></tr>
</table>

TRIAD typically starts with a description of the expected operational environment and the desired computing support for the essential work processes, as might be found in the concept of operations. The operational environment is refined through out the system life cycle. The scope of the analysis is influenced by the nature of the operational environment: the complexity of system interaction, the distribution of work processes, the work-process dependencies, and the sharing of survivability risks and responsibilities across multiple organizations. Survivability scenarios should describe the system's response both to expected stimuli and to

unexpected stimuli. Successful attacks on systems often result from stimuli that are outside the range of what the designers of the system expected. The behavior of the system when confronted with such stimuli is, therefore, very important for establishing the system's survivability. The conceptual architecture must, ultimately, document the system's response to both expected and unexpected stimuli.

Survivability tactics serve as the building blocks for architecting systems to satisfy specific survivability scenarios. This is analogous to the way Bachmann uses architectural tactics to satisfy quality attributes [Bachmann 02]. For example,

- Encapsulation is an architectural tactic intended to primarily improve modifiability by limiting the ripple effect of changes.
- Replication is an architectural tactic intended to improve performance by reducing response time through locality or improving reliability by providing redundant copies of function or data.

TRIAD introduces survivability tactics into the architecture iteratively, to address attacks that target different elements and that require increasing degrees of attacker sophistication. Just as with other quality attributes, these tactics serve to satisfy specific survivability scenarios, which characterize survivability for the application.

5 Example: TRIAD Application

This section presents an example application of TRIAD to develop a survivability strategy for a hypothetical eBusiness, which we call eBiz. This example is meant to be more illustrative than realistic, although we expect that real-world systems could be analyzed at a strategic level with only a modest increase in complexity.

TRIAD, as described in Section 2, is very generic in nature, partitioning the design space into three primary sectors. Sections 3 and 4 describe in more detail the primary activities that need to occur and artifacts that need to be documented in each of the model sectors. These activities and artifacts could be assembled into a specific, working model in many ways. The details of the best assemblage will depend largely on the domain of application and the skills of the development team. Figure 14 depicts the process used to develop the survivability strategy for eBiz within the 3-sector TRIAD model.

Figure 14: eBiz Survivability Strategy Development Process

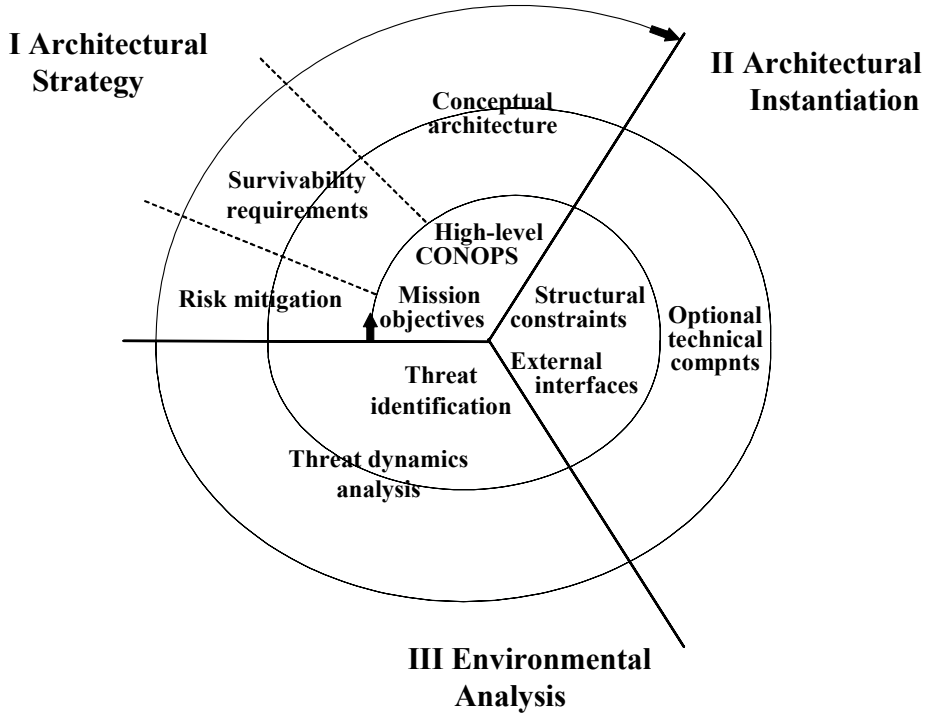


Figure 14 shows two iterations of the TRIAD spiral, culminating in a final set of survivability requirements and conceptual architecture that represent the eBiz survivability strategy. The first iteration, which is shown in the center of the figure, starts the process off at a very high level of abstraction by characterizing overall mission objectives, the general concept of operations, and any structural constraints or external interfaces with which any architecture will have to conform. The objective at this point is not to necessarily have a firm idea of how the survivability of the mission would be assured, but just to describe what the information system needs to accomplish and some idea of how this can be done within existing constraints. Sector III activities of this first iteration involve establishing who the adversaries to the organization are likely to be and, from a high-level point of view, how they can impact eBiz operations. Threat dynamics, as discussed previously, will be useful in this preliminary analysis.

After determining how to mitigate the threats identified, the second iteration of the spiral in Figure 14 focuses on developing an initial conceptual architecture. Survivability tactics play a primary role in mitigation analysis and the derivation of the conceptual architecture. As mentioned previously, refining the survivability strategy requires a certain amount of technical feasibility analysis, to ensure that the strategy can be implemented in a cost-effective way. The second iteration of sector II involves studying alternative technical components to ensure this is, in fact, the case. After further threat dynamics and mitigation analysis, a conceptual architecture is finalized based on the analysis.[6]

We justify the basis of TRIAD in the spiral model because survivable systems development is a domain in which the best directions for refinement are very unclear during the early stages of system conception and refinement. Experimentation and analysis are needed before a solution can be found with an acceptably small degree of residual risk of mission failure. The specification and analysis performed within each sector is gradually refined based on the experience of the previous iteration. For example, subsequent iterations of the spiral in Figure 14, which are not shown here, would refine the technical architecture within the constraints set forth by the conceptual architecture. There is a chance that the implementation of the survivability strategy would be hampered by lower level technical details that were not foreseen. In this case, the survivability strategy may have to be revised in light of this new information. The spiral model specifically supports such revisions based on new insights.

The rest of this section refines the eBiz survivability strategy using the process outlined in Figure 14. We split the refinement into the two iterations of the spiral, followed by the presentation of the final conceptual architecture.

[6] Traceability plays an important role in the development, evaluation, and maintenance of the survivability strategy and its implementation. We do not, however, explicitly show this aspect of the development of the eBiz survivability strategy to simplify its presentation.

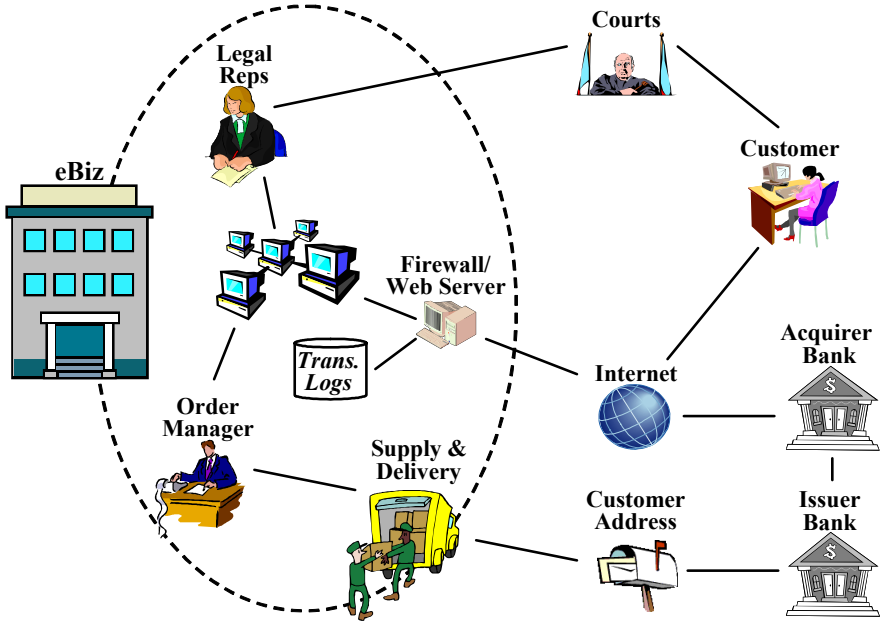
5.1 First Iteration

eBiz sells widgets over the Internet. Widgets range in quality and cost from fairly inexpensive, low-end widgets to premium quality, more expensive widgets. Most of eBiz's business comes from low-volume sales of low-end widgets to individuals, but recently they have seen an increasing amount of high-volume and high-end widget sales to other businesses. All of eBiz's sales are online credit card purchases. eBiz's mission, then, is to provide a high-quality service selling widgets over the Internet in a way that is both profitable and legal.

Operational Concept

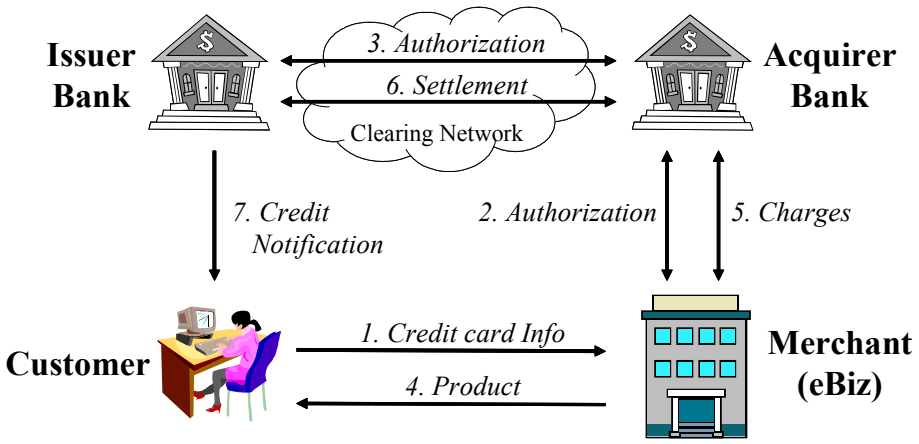
Figure 15 depicts the basic concept of operations for eBiz. Customers buy widgets by accessing eBiz's web server. eBiz has the usual connections to the Internet provided by a local Internet service provider. For security purposes, eBiz's intranet is protected from the Internet by a firewall. Widgets are not digital in nature and thus cannot be transmitted over the Internet. An order manager ensures all orders are filled and that a delivery service delivers widgets to the customer's address. To the extent possible, the order manager also makes sure that all credit card transactions are honored. Unfortunately for eBiz, current laws make the repudiation of Internet sales very easy for the consumer. We discuss online credit card transactions and repudiation in more detail below. When customers try to abrogate their agreement to honor the transaction, eBiz may file a civil suit to obtain remuneration. eBiz employs competent legal representation and keeps detailed transaction logs to execute lawsuits when necessary.

Figure 15: eBiz Concept of Operations



The online process for credit card payment is very similar to the traditional process for purchasing goods with a credit card [Hassler 01]. The main difference is that transactions occur over the Internet. Customers must, of course, have a credit card issued by an accredited institution, usually referred to as the *issuer bank*. The merchant's bank is usually called the *acquirer bank* because it acquires payment records, such as payment charge slips, from the merchant. Merchants must register with a payment service provider that connects the Internet with the private interbank clearing network to which both the issuer and the acquirer banks are linked. This sets up the infrastructure that supports online purchases with a credit card as shown in Figure 16.

Figure 16: Online Credit Card Payment Transaction

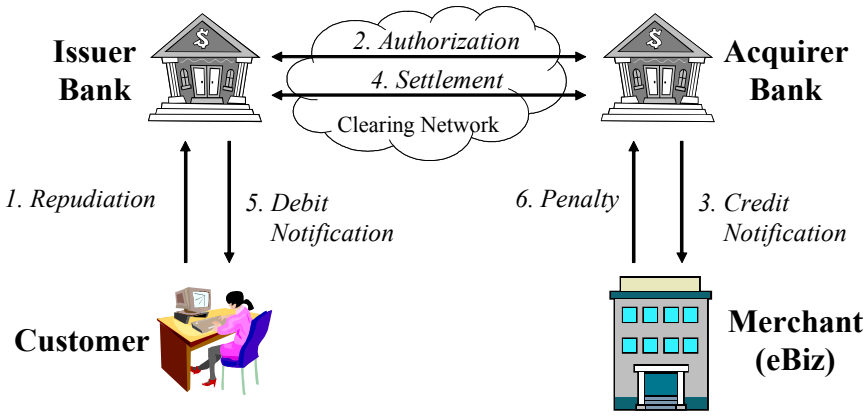


Hassler describes the typical online credit card payment transaction depicted in Figure 16 as follows: "The customer gives his credit card information (i.e., issuer, expiry date, number) to the merchant (1). The merchant asks the acquirer bank for authorization (2). The acquirer bank sends a message over the interbank network to the issuer bank asking for authorization (3). The issuer bank sends an authorization response (3). If the response is positive, the acquirer bank notifies the merchant that the charge has been approved. Now the merchant can send the ordered goods or services to the customer (4) and then present the charge (or a batch of charges representing several transactions) to the acquirer bank (5 up). The acquirer bank sends a settlement request to the issuer bank (6 to the left). The issuer bank places the money into an interbank settlement account (6 to the right) and charges the amount of sale to the customer's credit card account. At regular intervals (e.g., monthly) the issuer bank notifies the customer of the transactions and their accumulated charges (7). The customer then pays the charges to the bank by some other means (e.g., direct debit order, bank transfer, or check). Meanwhile the acquirer bank has withdrawn the amount of sale from the interbank settlement account and credited the merchant's account (5 down)" [Hassler 01]

Figure 17 shows the transaction required for purchase repudiation. Of particular note is the customer's ability to repudiate eBiz charges without eBiz's cooperation. Once the customer

denies authorization of a credit card transaction (1), the issuer bank automatically revokes the charges over the clearing network, with the full cooperation of the acquirer bank (2 and 4). The acquirer bank notifies eBiz of the repudiation (3) and assesses a financial penalty for the revoked transaction. Once settlement has taken place, the issuer bank notifies the customer of the charge revocation.

Figure 17: Online Credit Card Payment Repudiation



Concept Analysis

eBiz has had a thriving and relatively consistent business, even through all of the dotcoms ups and downs. However, in the last year they have been subject to increasing rates of purchase denial, i.e., a seemingly legitimate purchases made from customers who later deny that they ever made them. For eBiz, the detection of repudiated purchases is automatic through bank notification, but does this necessarily indicate fraudulent activity? For the purposes of our analysis, we assume that all other possible reasons for the repudiation, e.g. an accounting or delivery mistake, have been ruled out or are considered to be extremely unlikely. The two overriding reasons for purchase repudiation are either that a criminal has used stolen credit card information or that a legitimate, but dishonest or forgetful, card owner is trying to get away with not paying for widgets received. These are the two primary intrusion scenarios that we consider in our analysis.

Since the major credit card issuers hold eBusinesses completely liable for online purchase denials, this trend is having a severe effect on eBiz's bottom line. Civil lawsuit is of little use when criminals use stolen credit card numbers to make purchases online, since tracking those criminals can be extremely difficult. eBiz suspects the demand for widgets on the black market makes them an especially attractive target for this type of fraudulent purchase. eBiz needs to drastically reduce the volume of challenged sales in order to survive in an increasingly competitive widget market.

Increasing the accountability that customers must accept to make purchases with eBiz will help curb fraudulent activity, whether from criminal use of stolen credit card numbers or from dishonest cardholders themselves. Accountability is defined as "the property that enables activities on a system to be traced to individuals who may then be held responsible for their actions" [NCSC 88]. Accountability of action is usually increased through robust identification and authentication techniques, which, in eBiz's case, helps to verify that the person executing a particular transaction is the one authorized to use the credit card. This verification must be able to be used at a later date, if needed, so accountability requires that some sort of persistent records of the transaction be maintained.

Accountability can be used to control the extent of fraudulent activity that is taking place, where, for example, authentication and logging acts as a sort of feedback control on eBiz service. Figure 18 depicts an influence diagram that characterizes the influence of increased accountability on fraudulent purchases.[7] The figure shows the rate of both fraudulent credit card transactions as input to the element "Rate of fraudulent purchase request" in the lower left corner. Although eBiz does not control the rate of theft of credit card information, this rate certainly influences the rate of fraudulent criminal use of that information, as shown in the top left portion of the figure. An increased rate of fraudulent purchase request tends to increase the rate of fraudulent purchase approval, and ultimately to increase the rate of purchase repudiation, as shown on the right side of the figure. The arrow input to accountability robustness indicates that eBiz can use the choice of accountability techniques as feedback control on the rate of repudiation. The dashed line indicates that high levels of fraudulent activity should be dealt with by more robust accountability. The solid arrow pointing to "Rate of criminal use of stolen CC info" implies that this action will tend to decrease the extent of fraudulent activity.

Taking a step back, Figure 18 shows two self-limiting feedback loops. The outermost one, labeled Loop1, shows how feedback control is used to limit the rate of criminal use of stolen credit card information. The loop labeled Loop2 shares much of the same structure as Loop1, but reflects the influence of dishonest card owners on the rate of purchase repudiations. Both loops rely on accountability robustness as feedback control on the rates of repudiation. Not shown in the influence diagram, due to its qualitative nature, is the fact that robust accountability will tend to prevent the use of stolen credit card information, whereas it will tend to just make dishonest card owners think twice about the wisdom of denying the purchase after the fact.

[7] Underlined text shown in this and subsequent influence diagrams serves to provide additional explanation that may be helpful in understanding or interpreting the diagram. Underlined text may also describe specific attack trends, where information on those trends is available. External forces over which the system has no control are enclosed in boxes, as is the "Rate of theft of CC info" in Figure 18.

Figure 18: Dynamics of Fraudulent Card Use

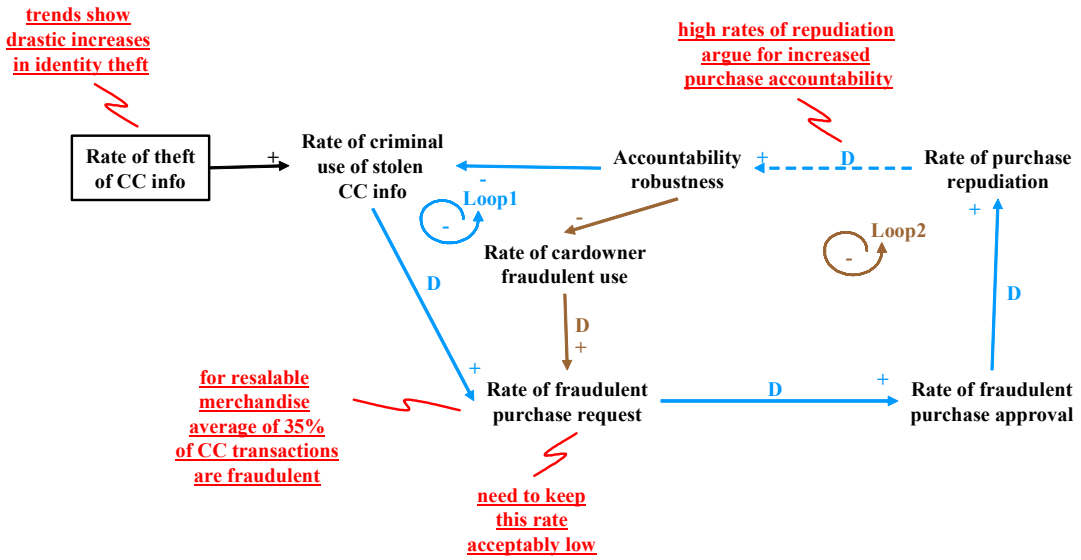


Figure 19 expands on the legal aspects of the choice of accountability mechanisms. Loop3 in the figure shows that accountability robustness tends to increase the strength of the legal case against repudiated purchases, which argues in favor of actually filing a lawsuit. Increased rates of lawsuit will, in turn, increase pressures to maintain or increase the robustness of the accountability measures. This clearly shows that the strength of accountability mechanisms influences the extent that legal action is possible to challenge the fraudulent activity. In isolation, the self-reinforcing loop argues for strong accountability measures to increase chances of recovering losses due to repudiated claims. Of course, the prosecution of suspected fraud will also depend on the amount of the purchase denied, since legal action will have some overhead associated with its use.

Figure 19: Extent of Legal Action

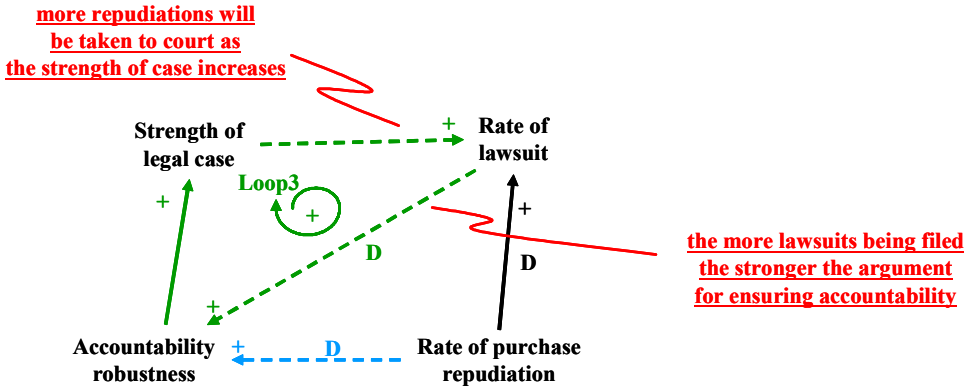
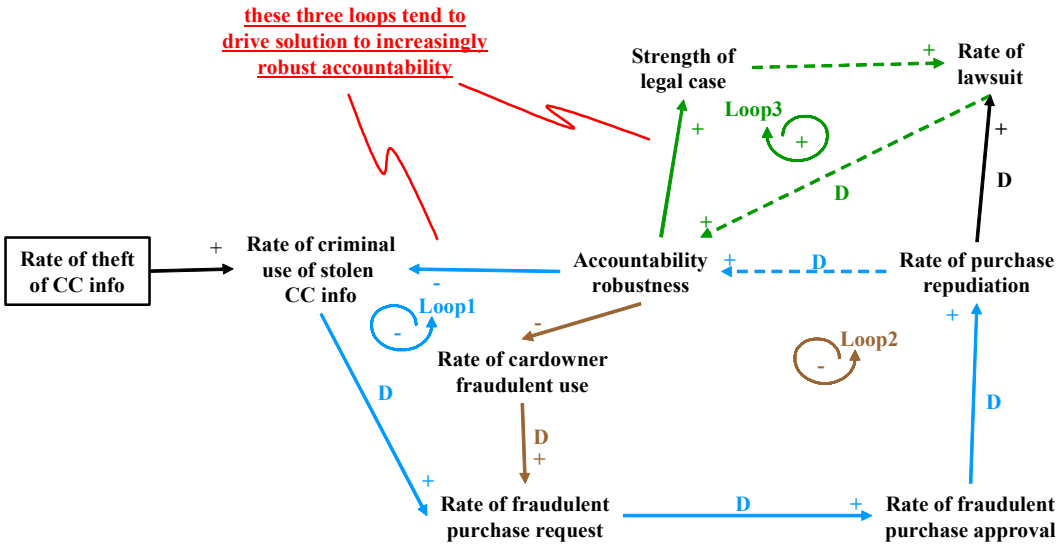


Figure 20 composes the influence diagrams of Figure 18 and Figure 19. The composite diagram shows that eBiz can respond to high rates of repudiation by (1) increasing accountability robustness to prevent fraudulent use of credit cards and (2) recovering costs through prosecution, where sufficient evidence of malfeasance exists and the amount of the purchase denied is sufficiently high. The next section describes the second iteration of the TRIAD model, which incorporates more robust accountability mechanisms into eBiz's conceptual architecture.

Figure 20: Composing Influence Diagrams of the First Iteration



5.2 Second Iteration

The last section described a threat dynamics and mitigation analysis that argues in favor of increased accountability of eBiz transactions. This section describes the second iteration of TRIAD, which investigates the use of digital signatures for increasing the accountability of eBiz customers on eBiz's overall mission. Digital signatures, combined with the infrastructure required for their use, constitute our initial survivability tactic for dealing with fraudulent purchases.

Refined Conceptual Architecture

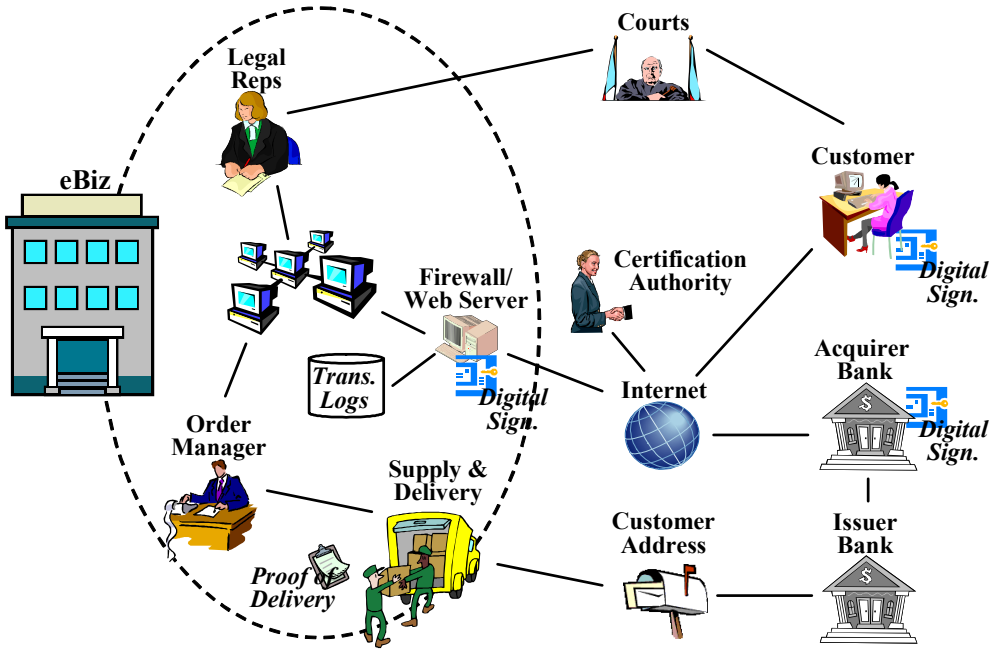
Table 4 presents the initial survivability requirements for eBiz. Figure 21 depicts an extension of eBiz's conceptual architecture that incorporates the use of digital signatures on all purchases of widgets from eBiz. The proper use of digital signatures supports the non-repudiation of purchases that eBiz so desperately needs to curb fraudulent purchases. Customers must digitally sign their purchase request as proof that they made the widget order. The banks must digitally sign their payment authorization as proof to eBiz that the transaction was approved. Of course, customers are going to demand proof of payment. Therefore,

eBiz must digitally sign a payment receipt sent to the customer. Likewise, the banks may require proof that eBiz asked for the amount of sale to be paid to eBiz's account. This requires that eBiz digitally sign any request for credit card payment to eBiz's corporate account.

Table 4: Initial eBiz Survivability Requirements

<table>
<tr><th colspan="2" rowspan="2">Stimulus</th><th colspan="5">Response</th></tr>
<tr><th colspan="2">Resistance</th><th>Recognition</th><th colspan="2">Recovery</th><th>Adaptation</th></tr>
<tr><td rowspan="2">Purchase request to eBiz with fraudulent intent</td><td>Uses stolen credit card info</td><td>Purchase requires digital signature</td><td>Block known lost or stolen cards</td><td rowspan="2">Automatically through bank notification of purchase repudiation</td><td>Civil lawsuit of digital signature owner</td><td rowspan="2">Proof of delivery</td><td>Update blocked card list regularly</td></tr>
<tr><td>Uses own credit card</td><td colspan="2">Purchase requires digital signature</td><td>Civil lawsuit of card owner</td><td>N/A</td></tr>
</table>

Figure 21: Initial eBiz Conceptual Architecture



The strategy described above supports the prevention of fraudulent purchase through the rigorous accountability mechanisms imposed by digital signatures. The strategy supports recovery of certain fraudulent purchases through civil lawsuit. The combination of digital signatures certified by a trusted authority is strong evidence for such a suit. For eBiz, the detection of repudiated (possibly fraudulent) purchases is automatic through bank notification. eBiz should track lost or stolen cards through "hot card lists," which banks routinely provide. Of course, scrupulous records need to be kept in case a lawsuit is brought later to recover repudiation losses. Proof of delivery to the card owner address would also be necessary in case the purchase is ever repudiated.

Conceptual Architecture Analysis

While eBiz's lawyers may be happy with the above strategy, its customers may have a different perspective. While strengthened countermeasures may be used as feedback control on malicious activity, such strengthening may also increase peripheral costs. Generally, these costs may be indirect, such as the increased complexity of administration or operation of the system as a whole. In our case, highly robust accountability can make it difficult for customers to deal with eBiz and may actually drive eBiz's customers away! This illustrates a tradeoff between security and usability, which has to be made in system design and maintenance, but is often realized only when the system goes into operation and changes are very expensive to make.

Figure 22 depicts the primary influences of accountability robustness on eBiz's legitimate customers. As shown, strengthened accountability tends to increase delays of legitimate purchase approvals, primarily due to increased authentication delays (e.g., telephone verification) or authentication setup overhead (e.g., through the use of trusted third parties). Such delays and overhead tend to increase customer frustration with the purchase process, which, in turn, tends to result in fewer customer purchases. This is a plausible scenario, since in the initial conceptual architecture eBiz requires digital signatures on all transactions for nonrepudiation. Most customers will go somewhere else before subscribing to the third party certification of digital signatures that would be required to do business with the company. Loop4 in the figure shows that the level of legitimate customer frustration can be used as feedback control on the strength of accountability. Indications of frustration could be assessed by survey or monitoring rates of purchase, for example.

Figure 23 depicts the influence diagram that composes the two instantiated threat response patterns. It also shows the primary influences of the system elements on eBiz profits, as a function of the revenue generated and the costs incurred. In summary, loops labeled Loop1, Loop2, and Loop3 describe the aspects of eBiz that argue for increasingly robust accountability mechanisms toward increased robustness, while Loop4 describes the aspects of eBiz that argue for less robust accountability mechanisms. Since the influence diagram is qualitative in nature, it does not describe exactly what will happen as time progresses to eBiz's rate of purchase repudiation, its rate of legitimate purchases, or, more importantly, its bottom line. It

does, however, help to set forth the tradeoffs that need to be made to identify an effective and affordable solution to eBiz's purchase repudiation problem.

Figure 22: Customer Frustration Due to Strengthened Accountability

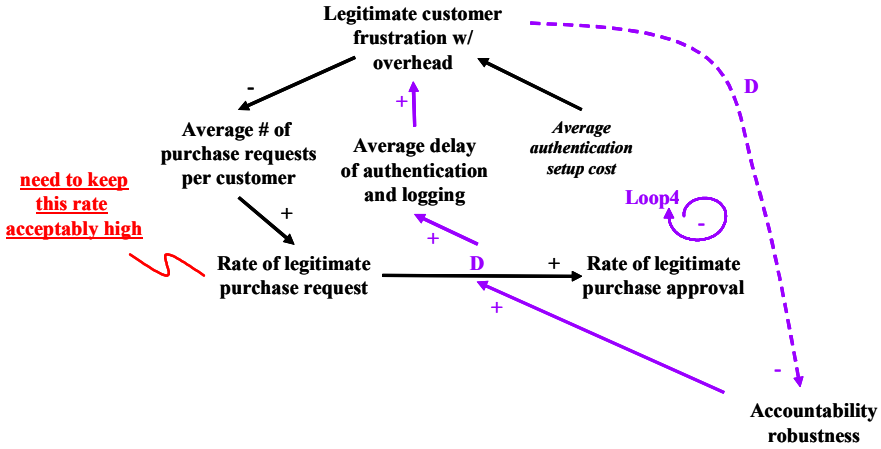
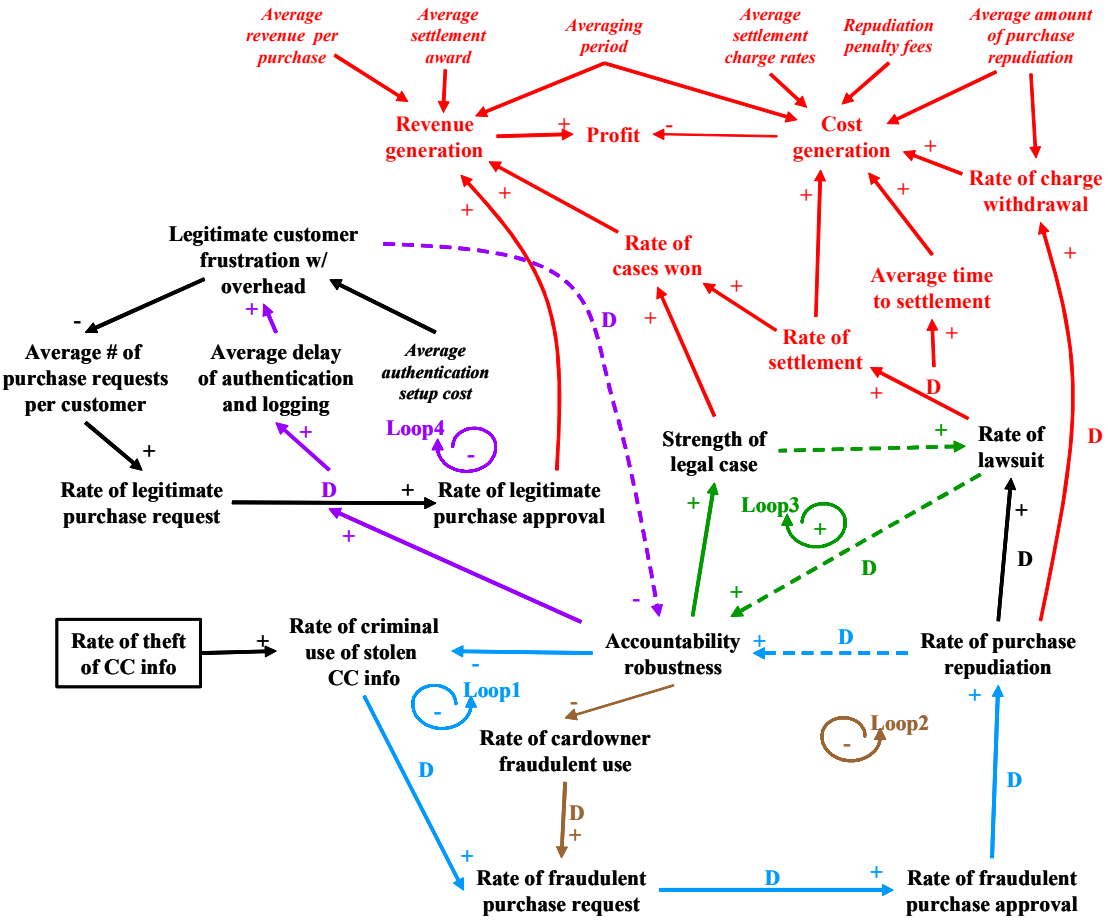


Figure 23: Composite Influence Diagram



5.3 Final Concept

The threat dynamics and mitigation analysis shows that eBiz cannot afford to be saddled by too cumbersome an accountability mechanism. However, they could choose to use more robust accountability mechanisms for purchases over a certain amount. For example, eBiz could adopt the policy that the purchase request for all sales over *X* dollars must be digitally signed, or there must be a redundant confirmation of the request. eBiz would need to experiment to determine the optimal policy parameters. If *X* is too high, eBiz may still be faced with much repudiation of lower cost purchases and won't have adequate evidence for a lawsuit to recover any losses. If *X* is too low, the policy will discourage sales to individuals because of high authentication overhead. Business-to-business purchases may not suffer much from rigorous accountability, since higher end purchasers are likely to accept more overhead, such as trusted third party certification of digital signatures, for their own added protection.

Policy options for redundant confirmation would also need to be explored. An example would be telephone verification of the purchase request using the telephone number associated with the card, which is often available from the credit card companies. In addition to tracking lost or stolen cards, eBiz should track past repudiation abusers, but should not expect the banks to support this process. Scrupulous records need to be kept in case a lawsuit is brought later to recover repudiation losses. A final precaution that eBiz could take is to require that all sales be sent to the address of the card owner unless a digital signature is received or redundant confirmation is made.

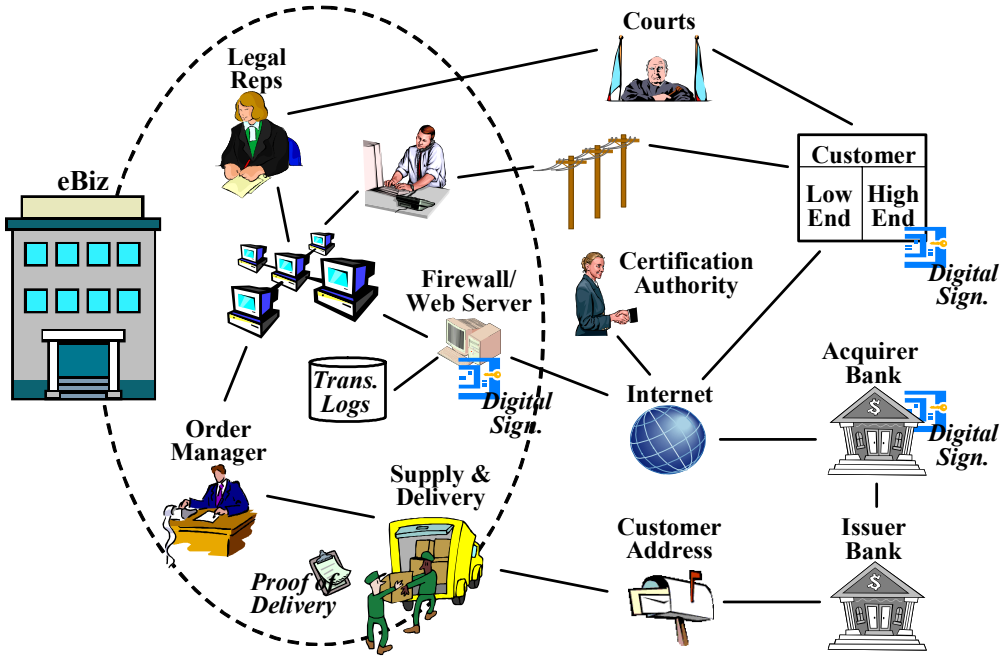
Table 5 represents the final survivability requirements for eBiz. Figure 24 depicts the final conceptual architecture. The survivability requirements and conceptual architecture embody the survivability strategy for eBiz. As shown, high-end customers are required either to digitally sign their transactions or await redundant confirmation of their transactions by phone. Clearly, not all the important decisions have been made, but an overall strategy is in place. The strategy supports prevention, to the extent practicable, of online fraudulent purchase without encumbering day-to-day customers with burdensome security requirements. The strategy supports recovery of certain fraudulent purchases through civil lawsuit.

Subsequent iterations of TRIAD would refine eBiz's technical architecture within the constraints set by the survivability strategy. Such refinement would require a certain amount of top-down thought to ensure that the strategy is implemented in support of the overall mission. As described previously, the implementation of the survivability strategy may be hampered by unforeseen lower level technical details. In this case, the survivability strategy would have to be revised in light of this new information. The spiral model specifically supports such revisions based on new insights.

Table 5: Final eBiz Survivability Requirements

Stimulus			Response						
			Resistance		Recognition	Recovery		Adaptation	
Purchase request to eBiz with fraudulent intent	Uses stolen credit card info	Purchase < $X	Block known lost or stolen cards	N/A	Automatically through bank notification of purchase repudiation	Send only to card owner address	Proof of delivery	Update blocked card list regularly	Notify card owner that card is being used fraudulently
		Purchase >= $X		Purchase requires digital signature or telephone confirmation		Civil lawsuit of digital signature owner			
	Uses own credit card	Purchase < $X	Block past repudiation abusers	N/A		Send only to card owner address		Document patterns of repudiation abuse	Notify cc company that card is being used fraudulently
		Purchase >= $X		Purchase requires digital signature or telephone confirmation		Civil lawsuit of card owner			

Figure 24: Final eBiz Conceptual Architecture



6 Conclusion

This report outlines an intrusion-aware design model, called TRIAD, for systematically refining information system architectures in complex, potentially unbounded, domains to resist, recognize, recover from, and adapt to known and hypothesized patterns of attack. TRIAD facilitates planning for the inevitable change to the threat and operational environment and helps trace the effect of change back to the survivability requirements and architecture. The spiral structure of the model iterates through three sectors of activity for developing the architectural strategy, for instantiating the architecture using technical components, and for analyzing the impact of the threat environment on system operations. This section describes the current limitations of the TRIAD model, how TRIAD can be used in the context of more comprehensive system development life cycles, and future work for further refining and applying TRIAD to increasingly complex, real-world examples.

6.1 Model Usage

TRIAD deals with only a small, but important, part of the survivable system development life cycle. In particular the model does not deal specifically with

- the implementation, evolution, or maintenance of the derived survivability architecture
- functions or properties required or desired of the system that do not contribute to the mission
- survivability-relevant failures due to internal faults or accidents
- program risks, such as funding or development team shortfalls, that are not due to malicious activity

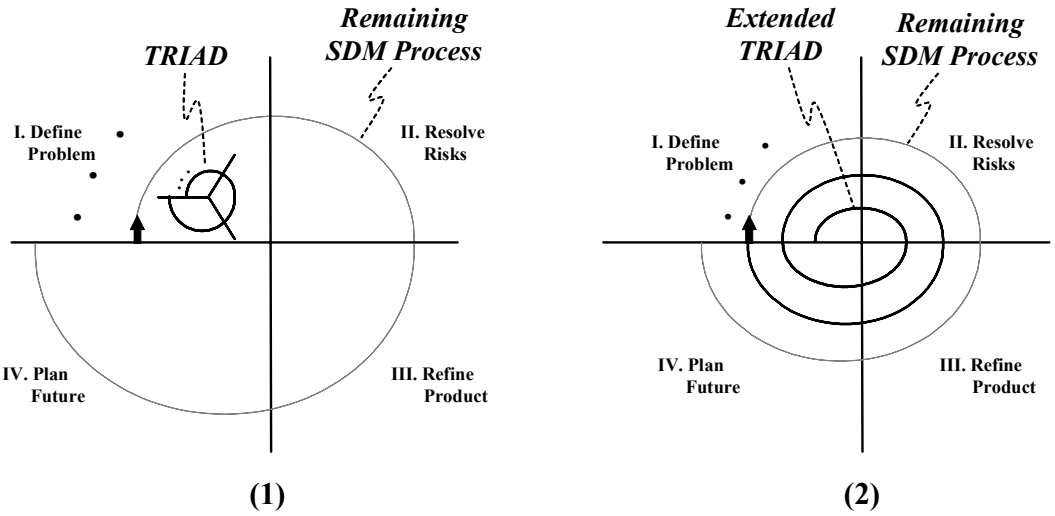
Incorporating TRIAD into an overall system development and maintenance (SDM) process will require resolving many of these issues. A detailed approach of how to do this depends largely on the details of the system problem domain and the development environment, and is beyond the scope of this report. We do, however, discuss some of the issues involved to provide a basis for formulating a comprehensive SDM process that incorporates IAD concepts. Fortunately, iterative spiral models are as useful for characterizing system maintenance (or enhancement) as they are for system development [Boehm 88].

As mentioned previously, the development of high-confidence information systems in complex settings where the impact of intrusion failure is severe demands an iterated, risk-driven process like the spiral model to gradually resolve uncertainties in the most efficacious manner. Using TRIAD in the context of a comprehensive SDM spiral can proceed in two primary ways (see Figure 25):

- viewing TRIAD as an up-front mini-spiral. In this case, execution of the IAD process leads to an advanced starting point for the larger SDM spiral.
- unrolling TRIAD activities and documented structures into the first few cycles of the SDM spiral. In this case, a more comprehensive integration of the two processes occurs.

The first of these methods is a viable alternative due to TRIAD's focus on mission. We believe that mission-related survivability requirements must be used to determine the overall shape of the architecture and must, therefore, be the focus of the initial iterations of the design process. Functions or properties required or desired that do not contribute to the mission must fit within the parameters defined by the survivability architecture and must not significantly lower the confidence that the system owners have in that architecture.

Figure 25: TRIAD in SDM Process (1) As Mini-Spiral or (2) Through Integration



The first method described above does not specifically allow for risks associated with non-malicious activities or events to be considered during the refinement of the survivability architecture. TRIAD can be extended in a fairly straightforward manner to deal with survivability-related non-malicious failures and accidents. Threat dynamics modeling and analysis of the impact of external failures and natural accidents can proceed in much the same manner as for malicious attacks. Accurately predicting the impact of internal faults on the mission may require specifying greater detail of internal operations in the threat dynamics model. In addition, attack tree modeling can be extended with fault tree analysis to analyze faults and accidents at a lower level of abstraction, because of the parallels between the two techniques.

Non-malicious program risks, such as resource shortfalls, are more difficult to handle with the first approach, since TRIAD deals only with operational risks during architecture formulation. Choosing the second approach is appropriate if program risks are high or if dealing with them explicitly within the process cannot be postponed until after architecture formulation. In this case, integrating IAD activities into the first few cycles of the comprehensive system development spiral enables resolution of program risk early on, before too many resources are expended in a programmatic dead end.

6.2 Future Work

TRIAD provides a solid foundation for the further refinement, experimentation, and validation of an approach to exploit our understanding of intruder behavior to improve system architecture design and operations. We plan a two-pronged approach: TRIAD tool development and TRIAD application.

Certain aspects of TRIAD are amenable to some form of automated or semi-automated tool support. Developing appropriate tools will support TRIAD's application to larger and more complex problems in varied domains. We believe that system dynamics provides a foundation for developing methods and tools that help engineers understand, characterize, and communicate the impact of a malicious threat environment on organizational and system operations and their respective missions. Further development of threat dynamics promises to provide structured and justifiable guidance on how an organization can best adopt policies, procedures, and technology to respond to the threat environment. TRIAD tool support will integrate and refine existing tools as appropriate (e.g., tools for system dynamics, attack trees, or intrusion analysis) and support the documentation and use of survivability tactics.

We also plan to continue to explore the viability of TRIAD and refine it through its application to the focused analysis of very specific problem situations. Each example will involve the identification of a specific problem situation, a TRIAD analysis and mitigation of that situation, and a characterization of the improvement gained through the analysis and mitigation. The improvement characterization will be a comparison of the problem situation before and after TRIAD analysis and mitigation. We plan to document survivability tactics in a structured way that facilitates their comparison, composition, and analysis. Preliminary work on this problem shows how attack patterns can be structured so that they can be applied in a variety of contexts [Moore 01a]. We plan to build on ongoing work at the CERT Coordination Center (CERT/CC) that involves the study and analysis of existing incident and vulnerability data to learn more about security incidents on the Internet.

By focusing on a specific problem in a narrow domain, we expect to get quick feedback on the efficacy of the model and insights into how to improve it. Feedback will help us understand the relationships and dependencies among sector activities and artifacts. Different prob-

lem and mitigation approaches will be investigated in the examples to increase the experience gained and insights gleaned, e.g., passive versus active defenses, military versus commercial domains, COTS versus custom solutions, technological versus procedural countermeasures. In each case the problem situation will be restricted to a particular malicious threat and its impact in the domain of interest. We expect this focused approach will streamline the TRIAD mitigation and analysis to one iteration of the full model, with little or no formal requirements tracing, thus ensuring the relative expediency of results.

Later work will involve a full-scale application of TRIAD and the tool support developed to demonstrate its scalability to more complex problems. Full-scale application will require assembling TRIAD activities and structures into a working system development life-cycle model appropriate to the application domain and development environment. This report illustrates an approach to develop a survivability strategy for a business that shows how one might start the TRIAD spiral process, followed by iterations through the sector activities and completing when an acceptable degree of residual risk of mission failure is determined. In addition to refining TRIAD based on the full-scale application, we plan to develop a tutorial for its use, with relevant examples, and initiate transition of the technology to an interested organization. TRIAD tool support, documentation of TRIAD case studies, and a detailed set of guidelines for TRIAD's application in varied settings should help make a compelling case for the model's use and transition.

Ultimately, with effective tool support and evidence of its efficacy, we expect that TRIAD will be integrated with more comprehensive life-cycle models for the development and maintenance of high confidence systems.

Appendix: Glossary

attack pattern – a generic representation of deliberate and malicious activity that commonly occurs in a specific architectural context

attack tree – a mission-critical compromise of a system and a hierarchical organization of *intrusion scenarios*, each of which accomplishes that compromise by different means

conceptual architecture (or conceptual survivability architecture) – a description of the system structure and function that addresses the need to ensure mission success despite penetrations and compromise at a level appropriate for the customer of the system

impact – the extent of harm to a system that results from a *threat*'s exploitation of a system *vulnerability* [DoD 00]

information system – any combination of information technology and people's activities using that technology to support operations, management, and decision-making

intrusion scenario – a description of people interacting with systems in a malicious way, thereby intentionally causing harm to an organization

security risk – a combination of the likelihood that a *threat* will occur, the likelihood that a *threat* occurrence will result in an adverse *impact*, and the severity of the resulting *impact* [DITSCAP 99]

survivability – the capability of a system to fulfill its mission by preserving essential services, even when systems are penetrated and compromised

survivability architecture – the combination of a system's *conceptual architecture* and *technical architecture*

survivability strategy – an overall approach to resist, recognize, recover from, and adapt to mission-compromising attacks

survivability tactic – a generic representation of an architectural approach to resist, recognize, recover from, or adapt to some pattern of attack in a specific context

survivability traceability – a characteristic of a system in which the survivability requirements are clearly linked to their sources (mission) and to the artifacts created during the system development life cycle based on these requirements (*survivability architecture*) [Ramesh 97]

survivability tracing – the process of ensuring *survivability traceability*

system dynamics – a method to model and analyze the holistic behavior of complex, managed systems as they evolve over time

technical architecture (or technical survivability architecture) – a description of the system structure and function that addresses the need to ensure mission success despite penetrations and compromise at a level of technical detail sufficient to actually build the system

technical component – any existing architectural building block, such as commercial off-the-shelf software or hardware

threat – any circumstance or event with the potential to cause harm to a system [DoD 00]

threat dynamics – an application of *system dynamics* that explicitly addresses hostile, malicious actions by individuals and the system operational response to such actions

vulnerability – a system characteristic that could be exploited by a *threat* to harm a system [DoD 00]

References

[Allen 00] Allen, J.; Christie, A.; Fithen, W.; McHugh, J.; Pickel, J.; & Stoner, E. *State of the Practice of Intrusion Detection Technologies* (CMU/SEI-99-TR-028, ADA375846). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000. <http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html>.

[Anderson 99] Anderson, R. H.; Feldman, P. M.; Gerwehr, S.; Houghton, B. K.; Mesic, R.; Pinder, J.; Rothenberg, J.; & Chiesa, J. R. "Securing the U.S. Defense Information Infrastructure: A Proposed Approach" (RAND Report MR-993-OSD/NSA/DARPA). Santa Monica, CA: RAND Corporation, 1999. <http://www.rand.org/publications/MR/MR993> (1999).

[Anderson 01] Anderson, R. *Security Engineering: A Guide to Building Dependable Distributed Systems*. New York: John Wiley & Sons, 2001.

[Arbaugh 00] Arbaugh, W. A.; Fithen, W. L.; & McHugh, J. "Windows of Vulnerability: A Case Study Analysis." *IEEE Computer 33*, 12 (December 2000): 52-59.

[Bachmann 02] Bachmann, F.; Bass, L.; & Klein, M. *Illuminating the Fundamental Contributors to Software Architecture Quality* (CMU/SEI-2002-TR-025). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002. <http://www.sei.cmu.edu/publications/documents/02.reports/02tr025.html> (2002).

[Bass 01] Bass, L.; Klein, M.; & Bachmann, F. "Quality Attribute Design: Primitives and the Attribute Driven Design Method." *4th Conference on Product Family Engineering*. Bilbao, Spain, 4 October 2001. <http://www.sei.cmu.edu/plp/bilbao_paper.pdf> (2001).

[Boehm 88] Boehm, B. "A Spiral Model of Software Development and Enhancement." *IEEE Communications 21*, 5 (May 1988): 61-72.

[CERT 01] CERT Coordination Center. *Managing the Threat of Denial-of-Service Attacks*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, October 2001. <http://www.cert.org/archive/pdf/Managing_DoS.pdf>.

[CERT 02] CERT Coordination Center. *Overview of Attack Trends*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002. <http://www.cert.org/archive/pdf/attack_trends.pdf>.

[Clements 02] Clements, P.; Bachmann, F.; Bass, L.; Garlan, D.; Ivers, J.; Little, R.; Nord, R.; & Stafford, J. *Documenting Software Architectures: Views and Beyond*. Boston, MA: Addison Wesley Longman, 2002.

[Coyle 96] Coyle, G. *System Dynamics Modeling*, New York: Chapman & Hall, 1996.

[Coyle 00] Coyle, G. "Qualitative and Quantitative Modeling in System Dynamics: Some Research Questions." *System Dynamics Review 16*, 3 (Fall 2000): 225-244.

[Cybenko 02] Cybenko, G.; Giani, A.; & Thompson, P. "Cognitive Hacking: A Battle for the Mind" *IEEE Computer 35*, 8 (August 2002): 50-63.

[DITSCAP 99] U.S. Department of Defense. *DoD Information Technology Security Certification and Accreditation Process (DITSCAP)*. DoD Instruction 5200.40, 30 November 1999. <http://www.sabi.org/history.htm> (2002).

[DoD 99] U.S. Department of Defense. *Introduction to Threats to Department of Defense Information Systems*. Secret and Below Interoperability Initiative Report, 30 September 1999. <http://www.sabi.org/history.htm> (2002).

[DoD 00] U.S. Department of Defense. *Basic Risk Management for Department of Defense Information Systems: Informal Reference Guide Edition 1.1*. Secret and Below Interoperability Initiative Report, 21 January 2000. <http://www.sabi.org/history.htm> (2002).

[DoD 02] U.S. Department of Defense. *Information Assurance (IA) Directive* 8500.1, October 22, 2002. <http://www.dtic.mil/whs/directives/corres/pdf/d85001_102402/d85001p.pdf>.

[Ellison 99] Ellison, R. J.; Fisher, D. A.; Linger, R. C.; Lipson, H. J.; Longstaff, T. A.; & Mead, N. R. *Survivable Network Systems: An Emerging Discipline* (CMU/SEI-97-TR-013, ADA341963). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1997, revised 1999. <http://www.sei.cmu.edu/publications/documents/97.reports/97tr013/97tr013abstract.html> (1999).

[Fisher 99] Fisher, D. A. & Lipson, H. J. "Emergent Algorithms: A New Method for Enhancing Survivability in Unbounded Systems." Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1999. <http://www.cert.org/archive/html/emergent-algor.html>.

[Forrester 61] Forrester, J. W. *Industrial Dynamics*. Republished by Productivity Press, Portland, OR. Cambridge, MA: MIT Press, 1961.

[Hassler 01] Hassler, V. "Security Fundamentals for E-Commerce." *Proceedings of the 1st Conference On Computer and Communications Security*. Norwood, MA: Artech House, Inc., 2001.

[IATF 02] Information Assurance Technical Forum. "The Information Systems Security Engineering Process." IATF Release 3.1, September 2002.

[Jacobson 99] Jacobson, Ivar; Booch, Grady; & Rumbaugh, James. *The Unified Software Development Process*. Boston, MA: Addison Wesley Longman, 1999.

[Knight 00] Knight, J. C.; Sullivan, K. J.; Elder, M. C.; & Wang, C. "Survivability Architectures: Issues and Approaches." *Proceedings of the 2000 DARPA Information Survivability Conference and Exposition (DISCEX 2000)*. Hilton Head, South Carolina, Jan. 25-27, 2000. Los Alamitos, CA: IEEE Computer Society, 2000.

[MAFTIA 02] MAFTIA Partners. "Malicious- and Accidental-Fault Tolerance for Internet Applications." IST Programme RTD Research Project IST-1999-11583. <http://www.newcastle.research.ec.org/maftia> (2002).

[Maier 00] Maier, M. W. & Rechtin, E. *The Art of Systems Architecting*. Boca Raton, FL: CRC Press, 2000.

[Marmor-Squires 89] Marmor-Squires, A. B.; McHugh, J.; Branstad, M.; Danner, B.; Magy, L.; Rougeau, P.; & Sterne, D. "A Risk Driven Process Model for the Development of Trusted Systems." 184-192. *Proceedings of the 1989 Computer Security Applications Conference.* Tucson, Arizona, December 4-8, 1989. Los Alamitos, CA: IEEE Computer Society, 1990.

[McDermott 99] McDermott, J. & Fox, C. "Using Abuse Case Models for Security." *Proceedings of the 15th Annual Computer Security Applications Conference*. Phoenix, Arizona, Dec. 6-10, 1999. Los Alamitos, CA: IEEE Computer Society, 1999. <http://www.computer.org/proceedings/acsac/0346/0346toc.htm> (2002).

[McHugh 00] McHugh, J.; Christie, A.; & Allen, J. "Defending Yourself: The Role of Intrusion Detection Systems." *IEEE Software 17*, 5 (September/October 2000): 42-51.

[Mead 00] Mead, N.; Ellison R.; Linger R.; Longstaff, T.; & McHugh, J. *Survivable Network Analysis Method* (CMU/SEI-2000-TR-013, ADA383771). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000. <http://www.sei.cmu.edu/publications/documents/00.reports/00tr013.html> (2000).

[Moore 01a] Moore, A. P.; Ellison, R. J.; & Linger, R. C. *Attack Modeling for Information Security and Survivability* (CMU/SEI-2001-TN-001, ADA388771). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2001. <http://www.sei.cmu.edu /publications/documents/01.reports/01tn001.html> (2001).

[Moore 01b] Moore, A. P. & Ellison, R. J. "Attack Modeling for Survivable System Analysis," in *Proceedings of the Information Systems Survivability Workshop, Dependable Systems and Networks Conference*, Gothenburg, Sweden, July 2001. <http://www.cert.org/archive/pdf/intrusion-aware.pdf> (2001).

[Neumann 00] Neumann, P. G. *Practical Architectures for Survivable Systems and Networks*. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, 30 June 2000. <http://www.csl.sri.com/neumann/survivability.pdf> (2000).

[NCSC 88] National Computer Security Center. "Glossary of Computer Security Terms." NCSC-TG-004 Version 1, October 1988.

[NCSC 91] National Computer Security Center. "A Guide to Understanding Identification and Authentication in Trusted Systems." NCSC-TG-017 Version 1, September 1991.

[OPSEC 00] The Centre for Counterintelligency and Security Studies, *Intelligence Threat Handbook*. Greenbelt, MD: Interagency OPSEC Support Staff, June 2000.

[Potts 95] Potts, C. "Using Schematic Scenarios to Understand User Needs*."* 247-256. *Proceedings of DIS'95—ACM Symposium on Designing Interactive Systems: Processes, Practices, Methods, & Techniques*. Ann Arbor, Michigan, Aug. 23-25, 1995. New York: ACM Press, 1995.

[Prakken 00] Prakken, B. *Information, Organization and Information Systems Design*, Norwell, MA: Kluwer Academic Publishers, 2000.

[Prowell 99] Prowell, S. J.; Trammell, C. J.; Linger, R. C.; & Poore, J. H. *Cleanroom Software Engineering: Technology and Process*. Boston, MA: Addison Wesley Longman, 1999.

[Ramachandran 02] Ramachandran, J. *Designing Security Architecture Solutions*. New York: John Wiley & Sons, 2002.

[Ramesh 97] Ramesh, B.; Stubbs, C.; Powers, T.; & Edwards, M. "Requirements Traceability – Theory and Practice," *Annals of Software Engineering*, 3 (1997): 397-415.

[Ramesh 98] Ramesh, B. "Factors Influencing Requirements Traceability Practice." *Communications of the ACM 41*, 12 (December 1998):37-44.

[Rushby 83] Rushby, J. M. & Randell, B. "A Distributed Secure System." *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, California, April 25-27, 1983. Maryland: IEEE Computer Society Press, 1984.

[Salter 98] Salter, C.; Saydjari, O.; Schneier, B.; & Walner, J. "Toward a Secure System Engineering Methodology." *Proceedings Of New Security Paradigms Workshop*. Charlottesville, Virginia, Sept. 22-25, 1998. New York: ACM Press, 1998.

[Schneier 99] Schneier, B. "Attack Trees: Modeling Security Threats." *Dr. Dobb's Journal*, December 1999.

[Schneier 00a] Schneier, B. *Secrets and Lies: Digital Security in a Networked World*. New York: John Wiley & Sons, 2000.

[Schneier 00b] Schneier, B. "Closing the Window of Exposure: Reflections on the Future of Security." *Securityfocus.com*, 2000. <http://online.securityfocus.com/guest/3384>.

[Sindre 00] Sindre, G. & Opdahl, A. L. "Eliciting Security Requirements by *Mis*use Cases," *Proceedings of Conference on Technology of Object-Oriented Languages and Systems*. Sydney, NSW, Australia, Nov. 20-23, 2000. Los Alamitos, CA: IEEE Computer Society Press, 2000.

[Soo Hoo 00] Soo Hoo, K. J. *How Much Is Enough? A Risk-Management Approach to Computer Security*. Report of the Consortium for Research on Information Security and Policy, Center for International Security and Cooperation, Stanford University, June 2000. <http://ldml.stanford.edu/cisac/pdf/soohoo.pdf> (2002).

[Sterman 00] Sterman, J. D. *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Burr Ridge, IL: McGraw-Hill Higher Education, 2000.

[Vatis 01] Vatis, M. A. *Cyber Attacks During the War on Terrorism: A Predictive Analysis*. Hanover, NH: Institute for Security Technology Studies at Dartmouth College, 2001. <http://www.ists.dartmouth.edu/ISTS/counterterrorism/cyber_attacks.htm>.

[Weidenhaupt 98] Weidenhaupt, K.; Pohl, K.; Jarke, M.; & Haumer, P. "Scenarios in System Development: Current Practice." *IEEE Software 15,* 2 (March/April 1998): 34-45.

[Wolstenholme 90] Wolstenholme, E. F. *System Enquiry: A System Dynamics Approach*. New York: John Wiley and Sons, 1990.

[Wolstenholme 93] Wolstenholme, E. F.; Henderson, S.; & Gavine, A. *The Evaluation of Management Information Systems: A Dynamic and Holistic Approach*. New York: John Wiley and Sons, 1993.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	March 2003	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Trustworthy Refinement Through Intrusion-Aware Design (TRIAD) Revised 2003	F19628-00-C-0003

6. AUTHOR(S)

Robert J. Ellison, Andrew P. Moore

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2003-TR-002

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116	ESC-TR-2003-002

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

High confidence in a system's survivability requires an accurate understanding of the system's threat environment and the impact of that environment on system operations. Unfortunately, existing development methods for secure and survivable information systems often employ a patchwork approach in which the focus is on deciding which popular security components to integrate rather than making a rational assessment of how to address the attacks that are likely to compromise the overall mission. This report proposes an intrusion-aware design model called trustworthy refinement through intrusion-aware design (TRIAD). TRIAD helps information system decision makers formulate and maintain a coherent, justifiable, and affordable survivability strategy that addresses mission-compromising threats for their organization. TRIAD also helps in evaluating and maintaining an information system design in terms of its ability to implement a survivability strategy. This report demonstrates the application of TRIAD to the refinement of a survivability strategy for a business that sells products over the Internet.

TRIAD provides a solid foundation for the further refinement, experimentation, and validation of an approach to exploit knowledge of intruder behavior to improve system architecture design and operations. Ultimately, with effective tool support and evidence of its efficacy, TRIAD will be integrated with more comprehensive life-cycle models for the development and maintenance of high-confidence systems.

14. SUBJECT TERMS	15. NUMBER OF PAGES
survivability, intrusion-aware design, survivable systems development, system architecting	96

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL