A Case Study in Survivable Network System Analysis

R. J. Ellison
Software Engineering Institute
Carnegie Mellon University

R. C. Linger
Software Engineering Institute
Carnegie Mellon University

T. Longstaff
Software Engineering Institute
Carnegie Mellon University

N. R. Mead
Software Engineering Institute
Carnegie Mellon University

ABSTRACT

Survivability is receiving increasing attention as a key property of critical systems. Survivability is the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents. We present a method for analyzing survivability of distributed network systems and an example of its application. Survivability requires system capabilities for intrusion resistance, recognition, and recovery. The Survivable Network Analysis (SNA) method permits assessment of survivability at the architecture level. Steps in the method include system mission and architecture definition, essential capability definition, compromisable capability definition, and survivability analysis of architectural softspots that are both essential and compromisable. Intrusion scenarios play a key role in the analysis. SNA results are summarized in a Survivability Map that links recommended survivability strategies to the system architecture. The case study summarizes application of the SNA method to a subsystem of a large-scale, distributed healthcare system.

Keywords
Survivability, network systems, essential services, Survivability Map, architecture analysis, intrusion scenarios

1 NETWORK SYSTEM SURVIVABILITY

Survivability Concepts

Modern society is increasingly dependent on large-scale networked systems for the conduct of business, government, and defense. Survivability of these systems is receiving increasing attention, particularly in the area of critical infrastructure protection [1]. As part of its Survivable Systems Initiative, the CERT® Coordination Center of the Software Engineering Institute (SEI)[1] at Carnegie Mellon University is developing technologies for analyzing and designing survivable network systems [2,3,4], and helping to foster a survivability research community [5]. Survivability analysis helps answer the following key questions:

- What essential functions must survive attacks and failures?
- What effects can attacks and failures have?
- What survivability risks exist?
- What architecture changes could improve survivability?

Survivability is defined as the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents. Unlike traditional security measures that require central control and administration, survivability addresses highly distributed, unbounded network environments with no central control or unified security policy. Survivability focuses on delivery of essential services and preservation of essential assets, even when systems are penetrated and compromised. As an emerging discipline, survivability builds on existing disciplines, including security [6], fault tolerance [7], and reliability [8], and introduces new concepts and principles. Survivability addresses a spectrum of adverse conditions. In this paper we focus exclusively on attacks, in the knowledge that our trace-based, compositional methods are applicable to analysis of failures and accidents as well.

® CERT and CERT Coordination Center are registered in the U.S. Patent and Trademark Office.

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

The focus of our survivability research is on delivery of *essential services* and preservation of *essential assets* during attack and compromise, and timely recovery of full services and assets following attack. Essential services and assets are defined as those system capabilities that are critical to fulfilling mission objectives. Survivability in the presence of attacks depends on three key system capabilities: *resistance, recognition*, and *recovery.* Resistance is the capability of a system to repel attacks. Recognition is the capability to detect attacks as they occur, and to evaluate the extent of damage and compromise. Recovery, a hallmark of survivability, is the capability to maintain essential services and assets during attack, limit the extent of damage, and restore full services following attack.

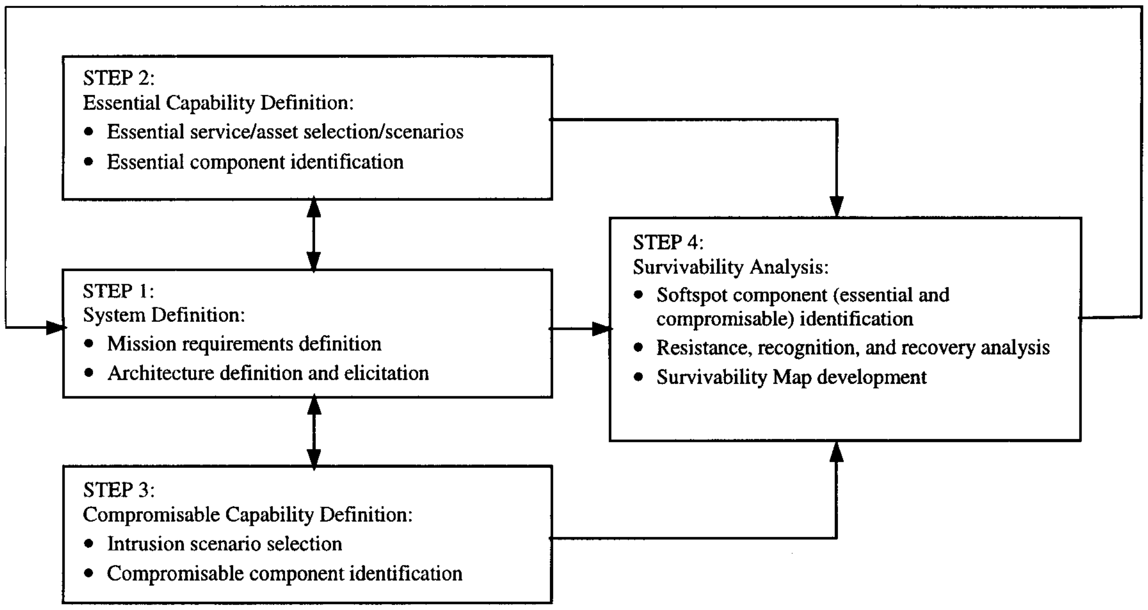
STEP 2:
Essential Capability Definition:
• Essential service/asset selection/scenarios
• Essential component identification
STEP 1:
System Definition:
• Mission requirements definition
• Architecture definition and elicitation
STEP 3:
Compromisable Capability Definition:
• Intrusion scenario selection
• Compromisable component identification
STEP 4:
Survivability Analysis:
• Softspot component (essential and compromisable) identification
• Resistance, recognition, and recovery analysis
• Survivability Map development

Figure 1. The Survivable Network Analysis Method

The Survivable Network Analysis Method

The Survivable Network Analysis method is depicted in Figure 1. In contrast to ad hoc, runtime patching in reaction to survivability problems, the SNA method focuses on systematically designing survivability into systems during development and evolution. The method can be applied to an existing or proposed system by a small team of evaluators, typically on the order of three or four people, through a structured interaction with system personnel composed of several meetings and working sessions. Evaluators must be knowledgeable in a number of disciplines, including architecture analysis, intrusion techniques, and survivability strategies.

The method is composed of four steps, as follows. In step 1, mission objectives and requirements for a current or candidate system are reviewed, and the structure and properties of its architecture are elicited. In step 2, essential services (services that must be maintained during attack) and essential assets (assets whose integrity, confidentiality, availability, and other properties must be maintained during attack) are identified, based on mission objectives and the consequences of failure. Essential service and asset uses are characterized by *usage scenarios.* These scenarios are mapped onto the architecture as execution traces to identify the composition of corresponding *essential components* (components that must be available to deliver essential services and maintain essential assets). In step 3, *intrusion scenarios* are selected based on the system environment and an assessment of risks and intruder capabilities. Selections are also influenced by the extensive CERT knowledge base of intrusion strategies. These scenarios are likewise mapped onto the architecture as execution traces to identify corresponding compositions of *compromisable components* (components that could be penetrated and damaged by intrusion). The SNA method takes into account strengths and weaknesses of COTS components, as well as any known security and reliability flaws. In step 4, *softspot components* of the architecture are identified as components that are both essential and compromisable, based on the results of steps 2 and 3. The softspot components and their supporting architectures are then analyzed for the key survivability properties of resistance, recognition, and recovery. The analysis of the "three Rs" is summarized in a *Survivability Map.* The map is a matrix that enumerates, for every intrusion scenario and its corresponding softspot effects, the current and recommended *architecture strategies* for resistance, recognition, and recovery. The Survivability Map provides feedback regarding the original architecture and system requirements, and often results in an iterative process of cost/benefit analysis and survivability improvement. While the SNA method was developed for use with large-scale distributed network systems, it is equally applicable to other architectures, including host-based and client–server systems. The scenario-based approach in SNA is a generalization of operation sequence [9] and usage scenario methods [10,11].

2 SENTINEL: THE CASE STUDY SUBSYSTEM

Management of mental health treatment is often performed as a manual process using hand-written forms and informal communication. As a result, substantial time and effort are consumed in the coordination of various treatment providers, including physicians, social service agencies, and healthcare facilities. CarnegieWorks, Inc. (CWI) is developing a large-scale, comprehensive management system to automate, systematize, and integrate multiple aspects of regional mental health care. The CWI system, named Vigilant, will ultimately be composed of some 22 subsystems operating on a distributed network of client and server computers, and will maintain a large and complex database of patient and provider records. A vital part of the Vigilant system is development and management of *treatment plans.* A treatment plan is developed for a *patient* by a *provider.* The *problems* of each patient are identified, together with a set of *goals* and *actions,* including medication and therapy, to achieve those goals. Each treatment plan is carried out by an interdisciplinary and interorganizational *action team* composed of providers. An *affiliation* is an organization that provides healthcare services, possibly to many patients. The development and management of treatment plans and the definition and coordination of action teams are key functions of the Sentinel subsystem, which resides on a node in the network architecture and communicates with other nodes through network protocols. As a subsystem of Vigilant, Sentinel interacts with providers, affiliations, and other subsystems. It maintains the action teams and treatment plans as part of the Vigilant patient database and applies regulatory and business rules for treatment plan development and validation. Because of the critical nature of mental health treatment, the need to conform to regulatory requirements, and the severe consequences of system failure, survivability of key Sentinel capabilities was identified by CWI personnel as extremely important.

3 APPLYING THE SURVIVABLE NETWORK ANALYSIS METHOD TO SENTINEL

The SNA method was applied to the Sentinel subsystem design through a structured series of meetings between the SNA analysis team and Sentinel project personnel (the customer and development teams), interleaved with analysis team working sessions. Specific results are summarized below in terms of the four SNA steps and their corresponding artifacts:

Step 1: System Definition

Mission Requirements Definition

The following normal usage scenarios (NUS) elicited from Sentinel requirements documentation characterize principal mission objectives of the subsystem. Each scenario includes a statement of the primary Sentinel responsibility with respect to the scenario:

- NUS1: Enter a new treatment plan. A provider assigned to a patient admitted into an affiliation performs an initial assessment and defines a treatment plan, specifying problems, goals, and actions. Sentinel must apply business rules to the treatment plan definition and validation.
- NUS2: Update a treatment plan. A provider reviews a treatment plan, possibly adding or changing problems, goals, or actions, and possibly updating the status of these items. Sentinel must apply business rules to the treatment plan update and validation.
- NUS3: View a treatment plan. A provider treating a patient views a treatment plan to learn the status of problems, goals, and actions. Sentinel must ensure that the plan being displayed is current and valid.
- NUS4: Create or modify an action team. A provider defines or changes the membership of a treatment team in an affiliation for a patient. Sentinel must ensure that the treatment team definition is current and correct.
- NUS5: Report the current treatment plans in an affiliation. An administrator views the current state of her affiliation's treatment of a patient or set of patients. Sentinel must ensure that the treatment plan summaries are current and correct.
- NUS6: Change patient medication. A provider changes the medication protocol in a treatment plan for a patient, possibly

in response to unforeseen complications or side effects. Sentinel must ensure that the treatment plan is current and valid.

Architecture Definition and Elicitation

The original Sentinel architecture obtained from design documentation is depicted in simplified form in Figure 2. Execution traces of the normal usage scenarios identified in Step 1 were used by the evaluation team to illuminate and understand architectural properties. The traces revealed component sequencing within the architecture, as well as referencing and updating of database artifacts.

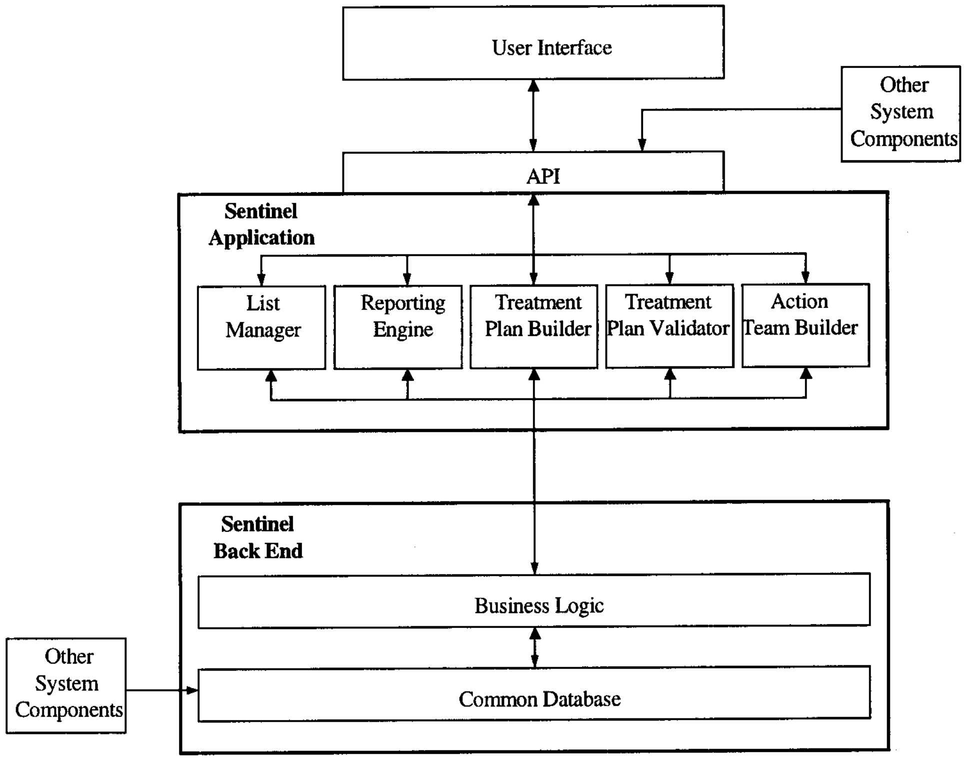

Figure 2. Original Sentinel Architecture

Architecture component functions are summarized as follows:

- User Interface: Resides outside of Sentinel to allow a single User Interface to serve multiple subsystems and components.
- Application Programming Interface (API): Provides synchronous remote procedure call (RPC) and asynchronous messaging facilities for use by the User Interface and other system components.
- List Manager: Maintains lists including patients, affiliations, providers, and action teams, and the relationships among them.
- Reporting Engine: Provides read-only viewing and reporting of Sentinel artifacts, including current treatment plans and their histories.
- Treatment Plan Builder: Creates treatment plans for patients, including problems, goals, and actions.
- Treatment Plan Validator: Checks the completeness and consistency of treatment plan development and modification.
- Action Team Builder: Provides capability to define and modify action team membership.
- Business Logic: Contains enterprise-defined business rules, including validation checks for treatment plan development

and logging triggers that manage change control of sensitive data.

- Common Database: Sentinel shares access to a database with other subsystems and components.

Step 2: Essential Capability Definition

Essential Service/Asset Selection/Scenarios

The essential service/asset analysis was based on the normal usage scenarios identified in Step 1. The analysis resulted in selection of a single essential service, namely, NUS3, the capability to view treatment plans. This service, more than any other, was deemed essential to delivery of mental health treatment because providers depend on real-time, on-demand access to treatment plans in clinical situations, particularly in cases involving medication or therapy problems in emergency situations. The other normal usage scenarios could be postponed for hours or even days in the event of system intrusion and compromise. The analysis also identified a single essential asset, namely, the treatment plans themselves. Preservation of treatment plan integrity and confidentiality was deemed essential to meeting Sentinel mission objectives. The other Sentinel artifacts, such as action teams, affiliations, and providers, could all be reconstructed or updated hours or days after intrusion with no irreversible consequences.

Essential Component Identification

Essential system components are those components that participate in delivery of essential services and preservation of essential assets. The execution trace of the NUS3 scenario revealed that the reporting engine and the database components, as well as their supporting components and artifacts, are essential to maintaining the capability to perform the scenario. As essential assets, the integrity and confidentiality of treatment plans depends on database components for security and validation.

Step 3: Compromisable Capability Definition

Intrusion Scenario Selection

Based on the system environment and assessment of intruder objectives and capabilities, the following five intrusion usage scenarios (IUS) were selected as representative of the types of attacks to which Sentinel could be subjected. The intrusions were selected based on customer priorities as well as on our judgment and experience for their ability to illuminate the risks and vulnerabilities that the essential services could experience. The scenarios were judged sufficient to cover the exposures, and the additional scenarios considered did not add significantly to the findings. The description for each scenario consists of an IUS number, the type of attack (shown in parentheses), and a brief explanation:

- IUS1 (data integrity and spoofing attack): An intruder swaps the patient identifications for two validated treatment plans.
- IUS2 (data integrity and insider attack): An insider uses other legitimate database clients to modify or view treatment plans controlled by Sentinel..
- IUS3 (spoofing attack): An unauthorized user employs Sentinel to modify or view treatment plans by spoofing a legitimate user.
- IUS4 (data integrity and recovery attack): An intruder corrupts major portions of the database, leading to a loss of trust in validated treatment plans.
- IUS5 (insider and availability attack): An intruder destroys or limits access to Sentinel's software so that it cannot be used to retrieve treatment plans.

Compromisable Component Identification

Compromisable system components are those components that can be accessed and damaged as shown in intrusion scenarios. The execution traces of the five IUS scenarios revealed the following component vulnerabilities:

- IUS1: This scenario compromises the treatment plan component. There were no validity checks made on treatment plans after the initial entry.
- IUS2: This scenario compromises the treatment plan component. The treatment plan changes might be consistent even though they were made by an improper agent.
- IUS3: This scenario compromises the treatment plan component. The majority of system users would object to the need for logging into the system repeatedly as a way of continually monitoring the authenticity of a user. While that approach could be tolerated for terminals in relatively secure locations, the system design had not considered terminals in open areas that are easily accessible by unauthorized users.
- IUS4: This scenario compromises the treatment plan component. Database recovery required higher priority with respect to operations.

- IUS5: This scenario affects all software components, including the reporting engine. While there were implicit user requirements for availability, those requirements had not been considered in the architecture.

Step 4: Survivability Analysis

Softspot Component Identification

As noted earlier, softspot components are those components that are both essential and compromisable. The foregoing analysis shows that the (essential service) reporting engine component and the (essential asset) database treatment plan component can both be compromised in a variety of ways.

Resistance, Recognition, and Recovery Analysis

Analysis of the three Rs resulted in the Survivability Map depicted in Table 1(ID stands for identification, TP for treatment plan, and DB for database). Development of the table began with the matching of each intrusion scenario trace (created in Step 3 above) to softspot components. Each trace was first checked to determine if any current resistance components in the architecture could increase the difficulty an intruder would confront in reaching the softspots referenced in the trace. Because no detailed implementation information was available to identify specific vulnerabilities in these resistance components, an assumption was made that any vulnerabilities in the components would be found and corrected over time. The greater the resources available to an intruder, however, the less time a resistance component will be completely effective. The current resistance components are described in the resistance column of the Survivability Map for each scenario.

For the recognition column, a similar process was followed. To assess the effectiveness of current recognition components, a number of assumptions were made and listed in the Survivability Map. For example, in scenario IUS3 in Table 1, there is a documented assumption that a provider will become suspicious when there are a large number of denied accesses to treatment plans reported to some party. If this assumption is not valid, then there are no current recognition strategies associated with this scenario.

For the recovery column, assumptions were made regarding standard practice in distributed database management facilities (standard backup and recovery of the database itself and version control of the Sentinel software). Table entries for current recovery strategies included these assumptions. If they are not satisfied in the final system, the recovery strategies will be less effective than those described in the Survivability Map.

Once the current resistance, recognition, and recovery strategies were identified, gaps and weaknesses were analyzed with the goal of locating common points in the architecture where a particular survivability improvement could address multiple scenarios or strategies. These high-leverage recommendations are listed in a consistent form and identified as a common recommendation. Other gaps for which there is no existing strategy in the resistance, recognition, or recovery columns were also addressed. For the resistance column, recommendations were made even where an existing resistance mechanism existed, as this mechanism can be expected to degrade over time. Ultimately, it is up to the system architect to determine the costs and benefits of implementing these recommendations. The Survivability Map can help an architect determine the impact of accepting risks associated with weaknesses in the resistance, recognition, or recovery columns, as these have a correlation to the intrusion scenarios and can affect the essential services or assets of the system. In Table 1, a number of gaps and assumptions are identified in the current strategies. Of particular interest to an architect are those recommendations that deal with multiple intrusion scenarios. For example, adding a crypto-checksum to the validation of a treatment plan can minimize the risks identified in several scenarios.

The modified architecture resulting from the Survivability Map analysis is depicted in Figure 3, with additions and changes shown with dashed lines and shading. Many of the recommendations call for alterations to the same architectural component. To further illustrate the overlaps, the recommendations are annotated with a reference number (in this case {1} through {6}) that is associated with the modified architecture. This makes it easy to determine which recommendations would alleviate risks in multiple intrusion scenarios. This view of the recommendations can help the architect allocate limited resources to high-impact modifications of the architecture. As the modified architecture was formed to address the recommendations in the Survivability Map, several natural locations emerged where changes could be implemented with minimal impact to the overall system. This beneficial localization was primarily due to the functional decomposition used in the original architecture. It is also likely that the scenario evaluations led to recommendations that were natural to the architecture, since their impact on individual modules was evident.

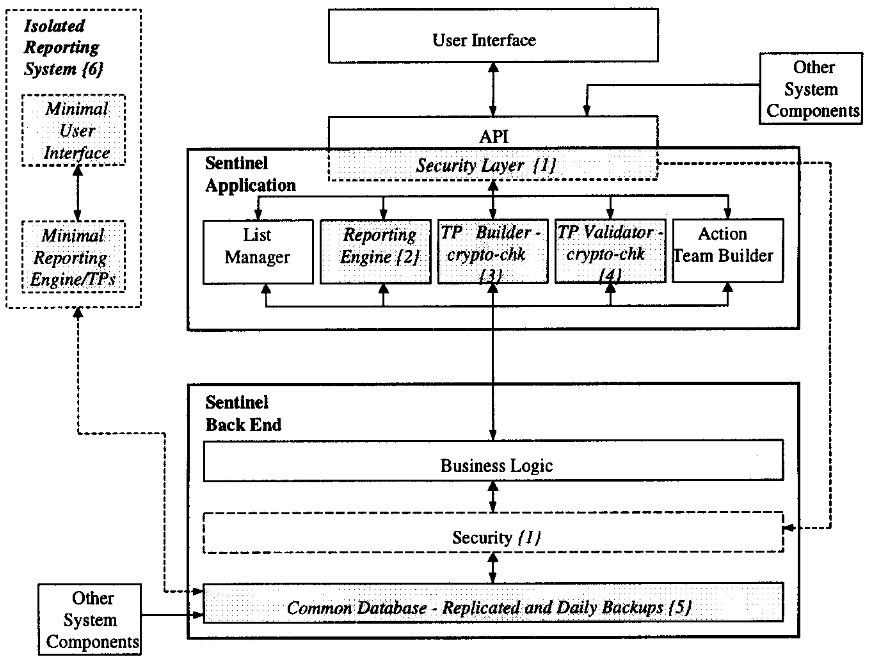

Figure 3. Sentinel Architecture with Survivability Modifications

Intrusion Scenario	Resistance Strategy	Recognition Strategy	Recovery Strategy
IUS1: Intruder swaps the ID of two validated TPs.	**Current:** Two passwords are required for TP access.	**Current:** Logging of changes made to DB. Provider may recognize an incorrect TP.	**Current:** Built-in recovery in commercial DB. Backup and recovery scheme defined.
	Recommended: Implement strong authentication supported in a security API layer. {1}	**Recommended:** Add crypto-checksum when TP is validated.{3} Verify crypto-checksum when TP is retrieved. {4}	**Recommended:** Implement a recovery mode in the user interface to support searching for and recovering incorrect TPs. {1}
IUS2: Outside agents exercise (legitimate) access to DB fields controlled by Sentinel.	**Current:** Security model for DB field access.	**Current:** None.	**Current:** Scrap data and start over, or find an early backup and verify each entry.
	Recommended: Need to verify the security model with respect to the integration of additional system components.	**Recommended:** Perform a validation on access of a TP for verification. {2} Add crypto-checksum when TP is validated.{3} Verify this checksum when TP is retrieved. {4}	**Recommended:** Scan DB for invalid crypto-checksums and/or invalid TPs and recover to last known correct TP. {4}
IUS3: An unauthorized user employs Sentinel to modify or view TPs by spoofing a legitimate user.	**Current:** None. No timeout is specified so that anyone can use a logged in but vacated terminal. However, intruder only has access to logged in user's TPs	**Current:** None, except for unusual number of denied accesses to TPs as an intruder attempts to locate particular TPs.	**Current:** Can get list of modified TPs through the spoofed users transaction history. Manually recover each modified record.

	Recommended: Add a short logout timeout for any terminals in uncontrolled areas (not physician's offices). {1}	**Recommended:** Add logging, access control, and illegal access thresholds to the security API. {1}	**Recommended:** Develop a recovery procedure and support it in the UI. {1}
IUS4: Intruder corrupts DB leading to loss of trust in validated TPs.	**Current:** Security model in the DB protects data against corruption.	**Current:** None, except when provider happens to recognize a corrupted TP.	**Current:** Locate an uncorrupted backup or reconstruct TPs from scratch.
	Recommended: Implement live replicated DB systems that cross check for validity (supported in many commercial DB systems). {5}	**Recommended:** Add and check crypto-checksums on records in the DB. {3} {4}	**Recommended:** Reduce the backup cycle to quickly rebuild once a corrupted DB is detected. {5}
IUS5: Intruder destroys the Sentinel software so it cannot be used to retrieve TPs	**Current:** Keep originals available.	**Current:** System doesn't work.	**Current:** Reload the system from originals.
	Recommended: Keep a spare CD available for quick recovery	**Recommended:** None. Easy to detect this one.	**Recommended:** Fast recovery from CD. Create a small sub-system that can retrieve TPs while Sentinel is down or being upgraded. {6}

Table 1. Sentinel Subsystem Survivability Map

In addition to using these findings for architectural analysis, Sentinel could also use the findings to make modifications to its requirements. The Mission Requirements Definition of Step 1 revealed that few specific survivability requirements had been specified, other than requiring 1) validation of treatment plan data, 2) utilization of some security features built into the standard login process and the database, and 3) a development strategy that would permit easy modification so that security features could be added. Changes are thus needed at the highest level to two areas of the requirements:

1) Under survivability conditions, there is a critical need for providers to view treatment plans within a reasonable time.

2) There is a need to protect the integrity of the treatment plans in the database.

4 LESSONS LEARNED

The SNA method is under continuing development and additional case studies are underway. The lessons learned at this stage focus on the validity of the initial assumptions and objectives of the method, as well as on refinements that can be explored in the case studies. The Sentinel case study began with three assumptions:

1. Survivability strategies could be organized in terms of resistance, recognition, and recovery.
2. The analysis should focus on early phases of the life cycle—specifically on the mission requirements, as they represent the essential services and assets of the system—and on the architecture, as it represents the components that must be survivable and the strategies for achieving survivability.
3. The application logic rather than the system infrastructure should bear a significant portion of the responsibility for implementation of survivability strategies [12].

The case study supported these assumptions. Organization of survivability strategies in terms of resistance, recognition, and recovery was straightforward and could be easily communicated to the customer. Identification of essential services and assets was critical in limiting the scope of the analysis and reducing the number of architecture modifications the customer should consider. It was indeed the case that most of the recommendations focused on the application level.

Success of the SNA method depends on the effectiveness of the recommendations, that is, on achievement of a modified system that exhibits improved survivability. Specifically, effectiveness depends on how well the system meets the survivability requirements selected by the customer in terms of essential services and assets, given the extent to which the SNA recommendations are implemented. It is important to note that the effectiveness of survivability requirements can be assessed and measured during implementation testing by intentionally causing compromisable components to fail. This will simulate the effects of successful intrusions. Such testing can provide objective feedback on the value of survivability strategies.

Of equal importance is whether the customer can readily incorporate the SNA recommendations into the existing software development process, and thus be able to adopt the suggested changes. Because the survivability recommendations for Sentinel concentrated on refining an existing architecture, rather than on requiring a redesign, they did satisfy this criterion.

While most of the recommendations focused on revisions to the application architecture, several called for changes in design and implementation, or in operations and procedures. The study also raised questions about the system's extensibility: Could the proposed architecture support, from a survivability perspective, the functionality desired in later versions. The recommendations were able to take advantage of existing system features to support reliability and fault tolerance, such as the transaction support and recovery mechanisms provided by the relational database.

The SNA process raises questions about all of the design choices that are embodied in a system architecture. Of special interest are choices that could impact survivability in a networked environment. For example, a networked application might include requirements for supporting disconnected operations by clients, and thus exhibit an architecture based on a messaging communications model that could impact post-intrusion recovery. Future studies will explore how to leverage that type of architectural choice to better support survivability, in the same way that this study leveraged the survivability capabilities of the relational database infrastructure.

5 FUTURE PLANS

The SNA method is an effort to present the first word, not the last word, on the complex problem of assessing system survivability. Additional case studies are in progress on large-scale network systems to validate and refine the method. Much work remains to be done, including:

- development of theory and practice for rigorous definition of network system behavior and architecture.
- development of canonical intrusion scenarios that embody the current knowledge of attack strategies.
- integration of risk analysis and management techniques into the SNA method.
- development and quantification of survivability metrics and measures of success.

6 ACKNOWLEDGEMENTS

The authors are pleased to acknowledge Howard Lipson and David Fisher of the SEI CERT/CC for their contributions to the definition of the SNA method. We thank Cindi Carbine of CarnegieWorks, Inc and Carnegie Mellon University's Master Of Software Engineering student developers Melissa Cheok, Ryan Soulier, Grama Srinivasan, Shinji Yamato, and Isabel Yebra for their support and cooperation in conducting this pilot.

7 REFERENCES

1. *Critical Foundations: Protecting America's Infrastructures: The Report of the President's Commission on Critical Infrastructure Protection*, Government Printing Office, Washington, DC, 1997.

2. Ellison, R. J.; Fisher, D. A.; Linger, R. C.; Lipson, H. F.; Longstaff, T.; and Mead, N. R. *Survivable Network Systems: An Emerging Discipline*. Technical Report CMU/SEI-97-TR-013, Pittsburgh, PA, Software Engineering Institute, Carnegie Mellon University, November 1997 (revised May 1999).

3. Linger, R. C.; Mead, N. R.; and Lipson, H. F. Requirements Definition for Survivable Network Systems. *Proceedings of International Conference on Requirements Engineering*. Colorado Springs, CO, IEEE Computer Society Press, Los Alamitos, CA., 1998.

4. Ellison, R. J.; Linger, R. C.; Longstaff, T.; and Mead, N. R. *A Case Study in Survivable Network System Analysis.* Technical Report CMU/SEI-98-TR-014, Pittsburgh, PA, Software Engineering Institute, Carnegie Mellon University, September 1998.

5. *Proceedings of 1998 Information Survivability Workshop*, Orlando, Florida, October 28-30, 1998, IEEE Computer Society. On-line at: <http://www.cert.org/research/isw98.html>, 1998.

6. Summers, Rita C. *Secure Computing: Threats and Safeguards.* New York, NY, McGraw-Hill, 1997.

7. Mendiratta, V. Assessing the Reliability Impacts of Software Fault-Tolerance Mechanisms. *Proceedings of Seventh*

International Symposium on Software Reliability Engineering, White Plains, NY, IEEE Computer Society Press, Los Alamitos, CA, 1992.

8. Musa, J.; Iannino, A.; and Okumoto, K. *Software Reliability: Measurement, Prediction, and Application.* New York, NY, McGraw-Hill, 1987.

9. Kemmerer, R.A.; Porras, P.A. Covert Flow Trees: A Visual Approach to Analyzing Covert Storage Channels. *IEEE Transactions on Software Engineering*, Vol.17, No.11, November, 1991, p. 1166-85.

10. Carrol, J.M. Five reasons for Scenario-Based Design. *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences.* Maui, HI, January 5-8, 1999, IEEE Computer Society Press; Los Alamitos, CA.

11. Prowell, S.J.; Trammell, C.J.; Linger, R.C.; and Poore, J.H. *Cleanroom Software Engineering: Technology and Process*, Addison-Wesley, Reading MA, 1999.

12. Saltzer, J. H.; Reed, D.P.; and Clark, D.D. End-to-End Arguments in System Design. *ACM Transactions in Computer Systems* 2, 4 (November, 1984), 277-288.

Robert Ellison is a Senior Member of the Technical Staff in the Networked Systems Survivability Program at the Software Engineering Institute. His research interests have included system survivability, software development environments, and CASE tools. He has a PhD in Mathematics from Purdue University. He is a member of the ACM and the IEEE Computer Society.

Richard Linger is a Visiting Scientist in the Networked Systems Survivability Program at the Software Engineering Institute and an adjunct faculty member at the Heinz School of Public Policy and Management, Carnegie Mellon University. His research interests include survivable systems, semantics of network architectures, large-scale system development, Cleanroom software engineering, software project management, and process improvement. He holds a BSEE from Duke University and is a member of the ACM and IEEE.

Thomas Longstaff is currently managing research and development in network security for the Networked Systems Survivability Program at the Software Engineering Institute. His research interests include information survivability and critical national infrastructure protection. He has a PhD in Computer Science from the University of California, Davis.

Nancy Mead is Senior Member of the Technical Staff in the Networked Systems Survivability Program of the Software Engineering Institute, and a faculty member in the Master of Software Engineering, Carnegie Mellon University. She is currently involved in the study of survivable systems architectures, and the development of professional infrastructure for software engineers. She has also served as Director of Education for the SEI from 1991-1994. Her research interests are in the areas of software requirements engineering, software architectures, software metrics, and real-time systems. She has a PhD in Mathematics from Polytechnic Institute of New York. She is a Senior Member of IEEE and IEEE Computer Society, and a member of ACM. She has served on numerous professional boards and committees, and is an international speaker.