A Case Study in Survivable Network System Analysis

R. J. Ellison
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
+1 412 268 7705
ellison@sei.cmu.edu

R. C. Linger
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
+1 412 268 7710
rlinger@sei.cmu.edu

T. Longstaff
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
+1 412 268 7074
tal@cert.org

N. R. Mead
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
+1 412 268 5756
nrm@sei.cmu.edu

ABSTRACT
This paper presents a method for analyzing the survivability of distributed network systems and an example of its application. Survivability is the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents. Survivability requires capabilities for intrusion resistance, recognition, and recovery. The Survivable Network Analysis (SNA) method permits assessment of survivability strategies at the architecture level. Steps in the SNA 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 method. SNA results are summarized in a Survivability Map which links recommended survivability strategies for resistance, recognition, and recovery to the system architecture and requirements. This case study summarizes the application and results of applying the SNA method to a subsystem of a large-scale, distributed healthcare system. The study recommended specific modifications to the subsystem architecture to support survivability objectives. Positive client response to study recommendations suggests that the method can provide significant added value for ensuring survivability.

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

1 NETWORK SYSTEM SURVIVABILITY

Survivability Concepts

As part of its Survivable Systems Initiative, the CERT[®] Coordination Center of the Software Engineering Institute (SEI) at Carnegie Mellon University is developing technologies and methods for analyzing and designing survivable network systems [1, 2, 3]. 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 [4], fault tolerance [5], and reliability [6], 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.

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.

[®] CERT is registered in the U.S. Patent and Trademark Office.

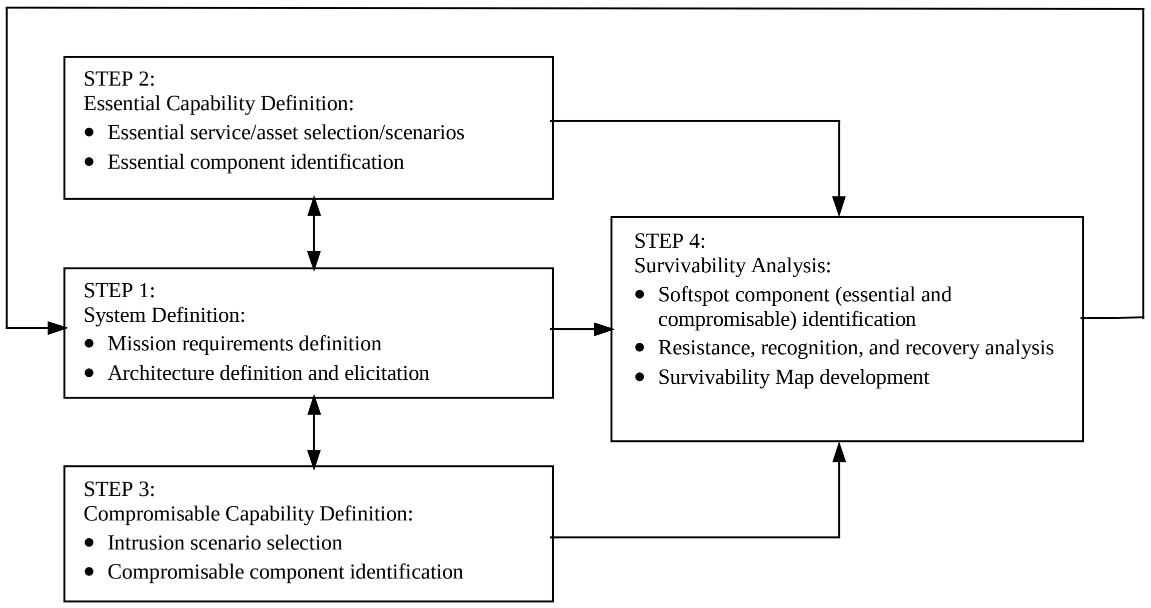
STEP 2:
Essential Capability Definition:
• Essential service/asset selection/scenarios
• Essential component identification
STEP 1:
System Definition:
• Mission requirements definition
• Architecture definition and elicitation
STEP 3:
Compromisable Capability Definition:
• Intrusion scenario selection
• 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 (SNA) method for assessing and improving the survivability of network architectures is depicted in Figure 1. The method can be applied to an existing or proposed system by a small team of trained evaluators through a structured interaction with system personnel composed of several meetings and working sessions.

The method is composed of four principal 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 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 assessment of risks and intruder capabilities. 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). 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 the supporting architecture are then analyzed for the three key survivability properties of resistance, recognition, and recovery. The analysis of the “three R’s” is summarized in a *Survivability Map.* The map is a two-dimensional matrix that enumerates, for every intrusion scenario and corresponding softspot effects, the current and recommended *architecture strategies* for resistance, recognition, and recovery. The Survivability Map provides feedback to the original architecture and system requirements, and may result in an iterative process of survivability evaluation and improvement.

2 SENTINEL: THE CASE STUDY SUBSYSTEM

Management of mental health treatment is often performed as a manual process based on hand-written forms and informal communication. As a result, substantial time and effort are consumed in 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. Treatment plan development and management and action team definition and coordination are key functions of the Sentinel subsystem. 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 has been 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 through a structured series of meetings between the SNA analysis team and Sentinel project personnel (customer and development team), interleaved with analysis team working sessions. Specific results of these meetings and working sessions are summarized below in terms of the four SNA steps and their 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 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 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 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 reference and update of database artifacts.

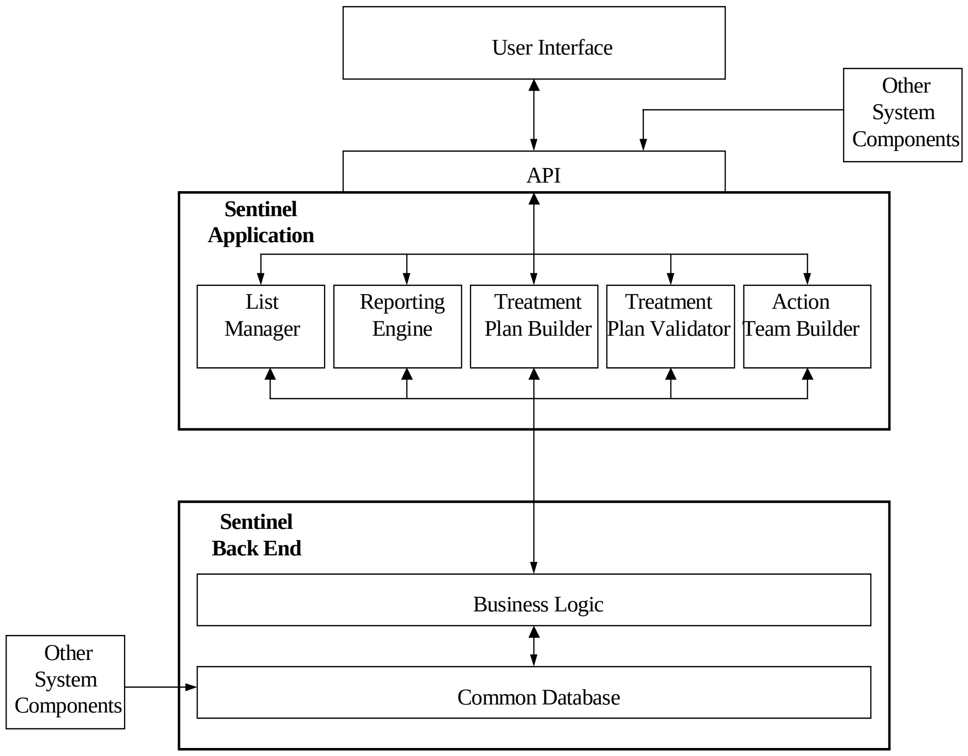

Figure 2. Original Sentinel Architecture

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.
- API: Provides synchronous RPC and asynchronous messaging facilities for use by the User Interface and other system components.
- List Manager: Maintains lists including patients, affiliations, providers, action teams, and relations 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.
- Database: Sentinel shares access to a common database with other subsystems and components.

Step 2: Essential Capability Definition

Essential Service/Asset Selection/Scenarios

Essential services and assets represent critical system capabilities that must survive and be available during intrusions. Criticality is based on analysis of mission objectives, risks and consequences of failure, and availability of alternatives. Such an analysis may result in selection of any number of essential services and assets, and may stratify them into survivability classes of varying criticality. The survivability analysis of the Sentinel subsystem was carried out together with CWI personnel, and 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 of medication or therapeutic problems of an emergency or life-critical nature. 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 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 judgement and experience to illuminate the risks and vulnerabilities that the essential services could experience. In this case, five scenarios were judged sufficient to cover the exposures; analysis of other systems could require more or fewer scenarios. Each scenario is preceded by an IUS number and type of attack (shown in parentheses), and followed by a brief explanation:

- IUS1 (Data Integrity and Spoofing Attack): An intruder swaps the patient identification of 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 loss of trust in validated treatment plans.
- IUS5 (Insider and Availability Attack): Intruder destroys or limits access to the Sentinel software so it cannot be used to retrieve treatment plans.

Compromisable Component Identification
Compromisable system components are those components that can be accessed and potentially damaged by 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 but made by an improper agent.
- IUS3: This scenario compromises the treatment plan component. The majority of system users would object to logging into the system repeatedly as a way to continually monitor the validity of the user. That assumption could be tolerated for those terminals in relatively secure locations. The system had not considered those terminals which were in open areas easily accessible by unauthorized users.
- IUS4: This scenario compromises the treatment plan component. Database recovery required higher priority with respect to operations.
- IUS5: All software components of the Sentinel subsystem are affected by this scenario. While there were implicit user requirements on availability, it 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. The survivability analysis focuses on the essential services and assets that these components provide in fulfilling the mission objectives of the system.

Resistance, Recognition, and Recovery Analysis
Analysis of the three R's 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 by matching each intrusion scenario trace (created in step 3 above) to the softspot components. Each trace was first checked for all current resistance (protection) components in the architecture that would increase the difficulty experienced by an intruder 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 them 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 process similar to the resistance analysis 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 common 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, so that if in fact they are not satisfied in the final system, the recovery strategy will be less effective than that described in the Survivability Map. However, the assumptions for the current recovery strategies take into account standard practice with regard to distributed database systems.

Once all of the current resistance, recognition, and recovery strategies were identified, gaps and weaknesses were analyzed for common points in the architecture where a particular survivability improvement could address multiple scenarios or multiple strategies. These high-leverage recommendations are listed in a consistent form and identified as a common recommendation. Other gaps identified by a lack of an existing strategy in any of 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 cost-benefit 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 are correlated to the intrusion scenarios that affect the essential services or assets of the system. In Table 1, a number of gaps and assumptions are identified in the current resistance, recognition, and recovery 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 addresses 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 in the survivability map affect the same architectural component. To further illustrate the overlaps, the recommendations are annotated with a reference number (in this case, {1} to {6}) associated with the modified architecture. In this way, it was easy to determine which of the recommendations addressed multiple intrusion scenarios. With limited resources to mitigate these risks, this view of the recommendations can help the architect allocate 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 in the existing architecture where implementation of the recommendations could be localized with minimal impact to the overall system. This was primarily due to the functional decomposition used in the original architecture. It is also likely that the evaluation of the scenarios led to the formation of recommendations that were natural to the architecture, since in executing the scenarios over the architecture, the impact on individual modules was evident.

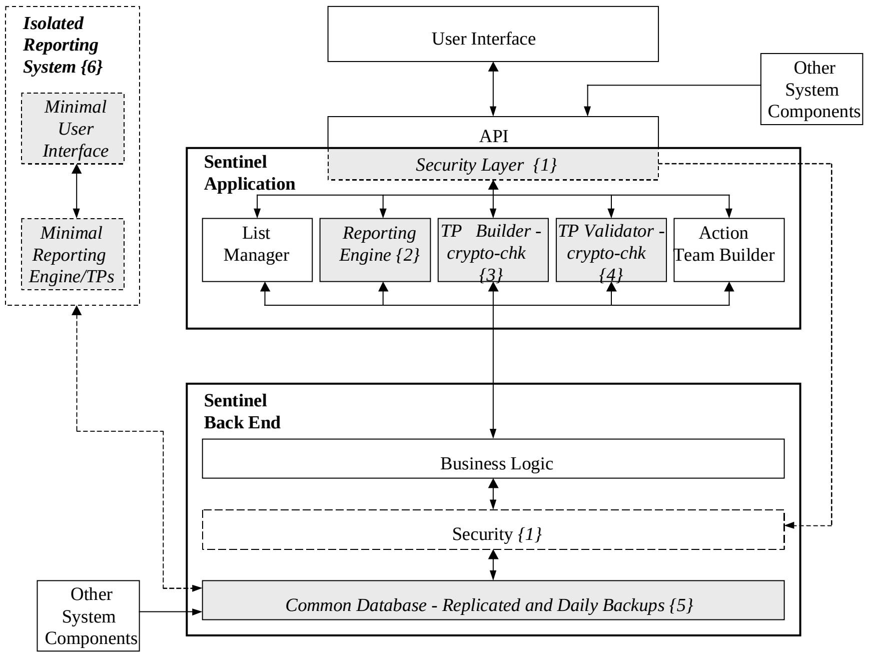

Figure 3. Sentinel Architecture with Survivability Modifications

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 in light of module addition and integration.	**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 architectural analysis, these survivability findings can also be reflected in modifications to Sentinel requirements. The Mission Requirements Definition of step 1 revealed few specific survivability or security requirements for the Sentinel subsystem, 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 to add security features. Changes are needed at the highest level to two areas of the requirements. Under survivability conditions, there is a critical need for providers to view treatment plans within a reasonable time. In addition, 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 planned. Lessons learned at this stage focus on the validity of the initial assumptions and objectives of the method, as well as refinements that can be explored in future case studies. The Sentinel case study began with three assumptions:

- Survivability strategies could be organized in terms of resistance, recognition, and recovery.
- 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.
- The application logic rather than the system infrastructure should bear a significant portion of the responsibility for implementation of survivability strategies [7].

The case study supported these assumptions. Organization of survivability strategies in terms of resistance, recognition, and recovery was straightforward and easily communicated to the customer. Identification of essential services and assets was a critical step in limiting the scope of the analysis, as well as in reducing the number and scope of architectural revisions which the customer should consider. The Sentinel subsystem examined in this study was just entering its implementation phase; future studies should include a need to reengineer existing systems.

The success of the SNA method depends on the effectiveness of the recommendations, that is, achievement of a modified system that exhibits improved survivability. More 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 failing compromisable components to simulate the effects of successful intrusions. Such testing can provide objective feedback on the effectiveness 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 requiring a redesign, they did satisfy this criterion.

While most of the recommendations focused on revisions to the application architecture, several suggested changes in design and implementation or in operations and procedures to support survivability in the existing architecture. The study did raise some issues of extensibility, that is, could the proposed architecture support the functionality desired in later versions from a survivability perspective. Analysis of extensibility could be an important aspect of future studies. The recommendations produced in this study 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 questions all design choices 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. A future study might explore how to leverage that type of architectural choice to better support survivability, in the same way that this study leveraged survivability capabilities of the relational database infrastructure.

5 FUTURE PLANS

The SNA method is an effort to say the first word, not the last word, about the complex problem of assessing network system survivability. Much work remains to be done, including:

- Development of theory and practice for rigorous definition of network system behavior and architecture.
- Addition of rigor to the SNA process through more formal analysis techniques.
- Development of a canonical set of intrusion scenarios that embody current knowledge of attack strategies.

- Integration of risk analysis and management techniques into the SNA method.
- Development of survivability metrics and measures of success.

6 ACKNOWLEDGEMENTS

The authors are pleased to acknowledge the contributions of David Fisher and Howard Lipson of the SEI CERT/CC, Cindi Carbine of CWI, and the CMU MSE student developers Melissa Cheok, Ryan Soulier, Grama Srinivasan, Shinji Yamato, and Isabel Yebra.

7 REFERENCES

1. 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.
2. 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. On-line at: <http://www.cert.org/research>.
3. Lipson, H. F. and Longstaff, T. *Proceedings of 1997 Information Survivability Workshop*. Pittsburgh, PA, Software Engineering Institute, Carnegie Mellon University. On-line at: <http://www.cert.org/research/isw97_hypertext/front_page.html>, 1997.
4. Summers, Rita C. *Secure Computing: Threats and Safeguards*. New York, NY, McGraw-Hill, 1997.
5. 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.
6. Musa, J.; Iannino, A.; and Okumoto, K. *Software Reliability: Measurement, Prediction, and Application*. New York, NY, McGraw-Hill, 1987.
7. 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.