Design for Survivability: A Tradeoff Space

Axel Krings
University of Idaho
Moscow, ID, USA
krings@uidaho.edu

ABSTRACT

When designing a system that has strong reliability, security, or survivability requirements one moves in a trade-off space with a delicate balance between causes and effects that have implications on various objective functions such as cost, performance, availability, analyzability, predictability, or feasibility. The key issues are: 1) given an existing system or application, what are the impacts of adjustments in the fault assumptions, 2) given an existing system or application, what are the impacts of adding or subtracting security features, and 3) given performance, availability, security, or survivability requirements, how can one determine feasibility based on the infrastructure- or application-induced limitations.

This research promotes design for survivability and analyzability to allow for effective assessment of the trade-off space from the view of dynamically changing fault models and the analyzability of a system. It gives pointers to new research directions and presents solutions that aid in making operational decisions or assessing impacts of design decisions.

Categories and Subject Descriptors

B.8 [**Performance and Reliability**]: Reliability, Testing, and Fault-Tolerance; D.4.6 [**Security and Protection**]: Invasive software; D.4.5 [**Reliability**]: Fault-tolerance; B.1.3 [**Control Structure Reliability, Testing, and Fault-Tolerance**]: Redundant design

General Terms

Design, Reliability, Security

Keywords

Survivability, Design for Survivability, Fault Models, Security, Result Certification

1. INTRODUCTION

Given the ever-increasing dependence of society on computerized systems, e.g., control of our critical infrastructures or environmental monitoring, assessing the dependability, security, and survivability of systems is imperative. Dependability is a general term that typically encompasses many definitions related to fault tolerance, such as reliability, safety, availability, performability, and maintainability. These terms have formal definitions and their quantification can be reviewed in any general text on fault tolerance. Quantifying security or survivability on the other hand is a different story. For example, to date there is not even a single agreed-upon definition for system survivability. Many qualitative and quantitative definitions have been suggested [7, 10]. However, there have only been modest advances related to analyzing and making systems more tolerant towards malicious acts or unpredictable events. This paper attempts to clarify some interrelated issues by combining them in the specification of a new approach to "design for survivability".

1.1 Design for Survivability

Design for Survivability is an approach that has much in common with *Design for Testability.* As integrated circuits became larger, exhaustive testing became infeasible, i.e., the set of test vectors needed to test circuits became intractable. This was formalized in the *test vector generation problem* and it was realized that one had to design circuits for testability. As systems are becoming increasingly complex and difficult to analyze the notion of designing for survivability, i.e., integrating the mechanisms that aid survivability into the system (rather than as an add-on feature), became a natural extension analogous to design for testability [7]. It should be noted that the concept of design for survivability (using diverse terminology) has been suggested by many researchers over the years.

As the need for verification and certification of systems has been increasing, the concept of design for survivability should be extended to require *design for analyzability*, as it is analysis that allows for the determination and quantification of the "health of a system".

1.2 Fault Models

The diversity of faults and their consequences on a system have been the primary motivator for the definition of fault models. Fault models have played a major role in reliability analysis, as well as agreement and consensus algorithms. A fault model addresses the behavior of the faults and the redundancy levels required to tolerate a single fault type or perhaps a mix of fault types. Many different fault models have been proposed over the years ranging from the simple models that make no assumptions about the fault behavior [8], to hybrid fault models considering multiple fault behavior. The latter consider a mix of faults ranging from benign, symmetric to asymmetric faults [22], with potential transmissive and omissive behaviors [2].

The causes of the faults considered in the dependability community have been attributed mainly to failing components, e.g., due to material fatigue, breakdown of physical or electronic components, accidents or environmental influences. The impact of malicious behavior, e.g., as the result of hacking, may it be from external sources or even insiders, viruses or Trojan horses, denial of service etc., have traditionally not been addressed. It was only later that accidental and intentional faults were discussed side by side [1, 9]. Thus, when considering survivability, one may take the approach of using the standard notions from fault tolerance and extending the definition of a fault to include those attributable to malicious act. It is of extreme importance to know what faults a system should be able to tolerate, and to know exactly what this requires of the system, since technologies may be inherently capable or incapable of tolerating certain faults.

2. SYSTEM DEFINITION

Our view of the systems is based on its survivability capabilities with respect to fault models. Specifically, every system functionality f_i can be mapped to a fault description F_i, which defines the fault model. For example, consider a communication service f_i that supports authentication. If authentication is assumed to be unbreakable or otherwise susceptible to compromise, then $F_i = (b)$, which indicates that attacks against authentication result in benign faults, as indicated by parameter b. If however authentication has been compromised, e.g., due to a security breach, then under the fault model in [22] $F_i = (b, s, a)$, where b, s, and a indicate possible benign, symmetric, and asymmetric fault-behavior respectively. In both cases authentication is assumed, but the assumption on the effectiveness of this functionality has changed. It should be obvious that wrong assumptions, and thus different F_i, will affect the survivability of the system.

We view a system as a collection of k functionalities, i.e., $\sum_{i=1}^{k} f_i$, and system survivability is a function of all F_i. Just as functionalities are not necessarily independent, e.g., functionality f_i may utilize functionality f_j, neither are their respective fault descriptions F_i and F_j. Thus any wrong assumptions about a fault description may propagate though many functionalities. Specifically, if $f_i = f_p \circ f_q$, i.e., of f_i is composed of f_p and f_q, then the resulting F_i encompasses all fault types of is components, i.e., $F_i = F_p \cup F_q$. The F_i needs to be analyzed for its potential impacts on the survivability requirements. Thus, the question about the impact of changes in fault assumptions, impact of security features availability, and their failure, boils down to the analysis of the functionalities in the context of the fault descriptions F_i. Conversely, given survivability requirements one can determine feasibility under infrastructure- or application-induced limitations. In this case F_i needs to be mapped onto the infrastructure, i.e., it needs to be determined if the infrastructure is inherently suitable to support the fault model described by F_i. It should be noted that the view of a system in terms of f_i and F_i allows to assess key issues 1) and 2) stated in the abstract.

Given the description of a system in terms of f_i and F_i we view a survivable system as comprised of three components: 1) Model Analysis, 2) Result Certification and 3) Adaptation.

2.1 Model Analysis

Model analysis addresses the quantification of survivability under consideration of the fault model and the assumptions on the fault environment. The fault model has been defined above as the partitioning of the fault space. The fault environment however is much more complex as it addresses the statistical assumption about the faults themselves, e.g., the fail rates, hazard function, and the independence or interdependence of faults.

Perhaps the simples analysis model is the traditional reliability model based on constant fail rates, which is specified as $R(t) = e^{-\lambda t}$, where $R(t)$ is the probability that the lifetime T of the system exceeds time t and λ is the fail rate. This model is used extensively in reliability block diagrams, fault tree analysis, Markov chain models and Petri net analysis. However, the assumption of a constant fail rate (exponential failure distribution) is very limited and mostly suitable for many problems in the dependability community if no malicious act and the independence of faults are assumed.

Given an analysis model one can determine the tradeoff space the system is operating in. The tradeoff space of a system describes the dependencies between different variables. For example, in the context of survivable storage in the PASIS project [20] the tradeoff space of security, availability, and performance was investigated with respect to data storage across distributed storage units. Survivability was addressed by spreading information among independent storage nodes. As the number of storage shares increased, so did performance (data bandwidth), however, system availability and security were negatively affected.

The reliability model $R(t) = e^{-\lambda t}$ is not generally suitable to analyze systems subjected to malicious act due to its limitations, i.e., constant fail rates, independence of failures, and the inability for effective censoring. Recent research inspired by biological systems has show that these limitations can be overcome. Specifically, diverse powerful new approaches to analyze systems subjected to unpredictable environments with interdependencies and uncertainty in parameter assessment were introduced in [11, 12, 13, 14, 15, 16, 17, 18]. Whereas much of the work was conducted in the context of wireless ad hoc and sensor networks, many methods are suitable for a much larger application domain. The flexibility of these new models can be used to assess uncertainty in the fault descriptions as well as in environments with complicated hazard functions, interdependencies, and environmental dependencies.

2.2 Result Certification

If a system cannot be analyzed, or if insufficient guarantees can be made about the outcome of the analysis, runtime certification can be useful. Result certification in distributed systems has been mainly based on mechanisms such as voting, spot-checking, credibility-based approaches, partial execution on reliable resources, and re-execution on reliable resources [21].

In [4, 5, 6] result certification was related to the notion of massive attacks. Specifically, fault-tolerant algorithms were considered in decentralized systems. Note that a fundamental survivability assumption is decentralization of critical functionalities. The motivation is typically the elimination of single points of failure.

If one considers a distributed application, then fault-tolerant algorithms can be used to mask faults up to a prescribed fault ratio q, with no assumptions on the fault behavior. Then probabilistic certification can be used to detect attacks that surpass the design-in threshold q of the algorithm.

The approaches in [4, 5, 6] are based on probabilistic Monte Carlo certification and indicate whether a result was either 1) CORRECT or 2) FAILED, together with a proof that an execution had failed. The probabilistic certification was said to be with error ϵ if the probability of the answer CORRECT, when the execution actually failed, was less than or equal to ϵ. Probabilistic result certification was shown to be particularly effective as the magnitude of an attack, i.e., the fraction of components that had been affected by the massive attack, was large. This also included large scale attacks that experienced common mode behavior. Note that common mode behavior is included in *arbitrary* faults, which are even more complicated than Byzantine, i.e., asymmetric, faults.

Whereas result certification based on probabilistic certification has been mainly used in grid applications, the general principle is suitable for distributed applications at any level of granularity. Since survivable systems should be designed distributed and decentralized [7], design for survivability can be extended to include the mechanisms necessary to allow for result certification. The addition of this concept would introduce a new dimension in design for survivability.

2.3 Adaptation

Adaptation is considered to be an integrated feature in any design for survivability. In [3, 19] survivability was described in terms of Resistance, Recognition, Recover, and Adaptation. Adaption implemented the mechanism to adapt the system to knowledge gained in the prior three phases.

Adaptation, in general, also encompasses movements in the tradeoff space. Recall that this research considers a system definition based on model analysis, result certification and adaptation. Adaptation is viewed to address the dynamics of changing fault descriptions F_i, which however has to be addressed in the context of the capabilities of the system to support functionality f_i. We define an *active* fault model as the fault model that the system currently subscribes to. Thus for functionality f_i the fault description F_i is the active fault model. In contrast, the *available* fault model, denoted by $\hat{F_i}$, is the hybrid fault model of the infrastructure of the system (or application). This is the mix of faults that the infrastructure is theoretically capable of producing. Note that this is seen strictly from the theoretical point of view. For example, a broadcast environment such as a bus is theoretically not capable of producing an asymmetric fault since any node on the bus can see all messages. This should however not be confused with the problems arising from weak solutions that do not take advantage of the infrastructure, e.g., a simple logical point-to-point communication on a bus rather than using utilizing the broadcast paradigm. The available fault model puts a bound on the active fault model. Thus what need to be concerned about is the impact of $\hat{F_i}$ on F_i. Note that if the infrastructure does not support the mechanisms necessary for restricting F_i to meet its survivability specifications, survivability cannot be achieved. In such a case result certification may be the last resort.

Fault descriptions can change over time. Let's consider again the authentication example at the beginning of Section 2. It may be reasonable to assume that authentication f_i is effective and any violation will be detected, i.e., the active fault model is $F_i = (b)$. Now, if intelligence suggests that authentication may be compromised, then the active fault model has to be changed, e.g., to $F_i = (b, s, a)$. If the available fault model $\hat{F_i}$ does not restrict the new F_i, survivability is compromised. Otherwise, survivability can theoretically be maintained, however, a shift in the tradeoff space will occur. In this example, this will likely result in decreased performance and increased number of resources, resulting in reduced availability as now more components can theoretically fail. Just as in the case of PASIS increased security demands in the example change performance and availability.

The analysis of the active and the available fault models relates directly to key issue 3) stated in the abstract, which is concerned with the feasibility of survivability requirements for f_i based on the infrastructure- or application-induced limitations. If infeasibility arises, i.e., if F_i exceeds the survivability specifications and $\hat{F_i}$ cannot restrict F_i which "is" the fault model for f_i, then this indication of the inability to satisfy survivability can be used to steer the application towards fail-safe behavior. Alternatively, f_i can be re-evaluated and infrastructure or application support can be adapted, e.g., via real-time reconfiguration, to adjust $\hat{F_i}$ to get a desired F_i. For example, a functionality f_i that utilizes point-to-point communication and transitions from $F_i = (b)$ to $F_i = (b, s, a)$ may be reconfigured to use broadcast (or multicast) communication that is inherently capable of avoiding asymmetric faults. Thus by adapting the infrastructure of f_i to a broadcast environment, F_i can be downgraded from $F_i = (b, s, a)$ to $F_i = (b, s)$, thus eliminating asymmetric fault behavior. Note that this can be achieved by using appropriate multicast protocols.

3. CONCLUSIONS

A survivability architecture was outlined that defines a system in terms of its functionalities and their temporal fault model requirements. The system was defined in terms of Model Analysis, Result Certification, and Adaptation. It was suggested that the deterministic methods from fault tolerance have limited effectiveness in the survivability environment which is assumed to be hostile. Therefore references to new biologically inspired non-deterministic work were given that have the potential to assess survivability under diverse assumptions much more suitable for malicious environments. The support for probabilistic result certification in design for survivability was suggested as it can help overcome limitations in the model analysis. Lastly, adaptation addressed the ability to react to changes in the operational environment, may they be perceived or experienced. Adaptation however could only operate within the available fault model. Nevertheless, the understanding of the active and available fault model can be used to introduce changes in the functionality or infrastructure in order to reduce the active fault model.

4. REFERENCES

[1] A. Avizienis, et.al., *Fundamental Concepts of*

Dependability, Information Survivability Workshop (ISW-2000), Boston, Massachusetts, Oct. 24-26, 2000.

[2] M.H. Azadmanesh, and R.M. Kieckhafer, *Exploiting Omissive Faults in Synchronous Approximate Agreement*, IEEE Trans. Computers, 49(10), pp. 1031-1042, Oct. 2000.

[3] R. J. Ellison, D. A. Fisher, R. C. Linger, H. F. Lipson, T. Longstaff and N. R. Mead, *Survivable Network Systems: An Emerging Discipline*, Technical Report CMU/SEI-97-TR-013, November 1997, Revised: May 1999.

[4] S. Jafar, A. Krings and T. Gautier, *Flexible Rollback Recovery in Dynamic Heterogeneous Grid Computing*, IEEE Transactions on Dependable and Secure Computing, (TDSC), in print.

[5] A. Krings, J-L. Roch, and S. Jafar, *Certification of Large Distributed Computations with Task Dependencies in Hostile Environments*, IEEE Electro/Information Technology Conference , (EIT 2005), May 22-25, Lincoln, Nebraska, 2005.

[6] A. Krings, J.-L. Roch, S. Jafar and S. Varrette, *A Probabilistic Approach for Task and Result Certification of Large-scale Distributed Applications in Hostile Environments*, Proc. European Grid Conference (EGC2005), in LNCS 3470, Springer Verlag, February 14-16, Amsterdam, Netherlands, 2005.

[7] A. Krings, *Survivable Systems*, Chapter 5 in: Information Assurance: Dependability and Security in Networked Systems. Morgan Kaufmann Publishers, Yi Qian, James Joshi, David Tipper, and Prashant Krishnamurthy Editors), in press, 2008.

[8] L. Lamport, et.al., *The Byzantine Generals Problem*, ACM Transactions on Programming Languages and Systems, Vol. 4, No. 3, pp. 382-401, July 1982.

[9] J.C. Laprie, editor, *Dependability: Basic Concepts and Terminology*, Springer-Verlag, 1992.

[10] Y. Liu, and K. S. Trivedi, *Survivability Quantification: The Analytical Modeling Approach*, International Journal of Performability Engineering, Vol. 2, No 1, Jan. 2006, pp. 29-44.

[11] Z.S. Ma, A. W. Krings, and R. E. Hiromoto, *Insect Sensory Systems Inspired Communication and Computing (II): An Engineering Perspective*, IEEE-ACM International Conference on Bio-inspired Systems and Signal Processing, (BioSignals 2008), Funchal, Madeira, Portugal, 28 - 31 January, 2008.

[12] Z.S. Ma, and A. W. Krings, *Survival Analysis Approach to Reliability Analysis and Prognostics and Health Management (PHM)*, Proc. IEEE AeroSpace Conference, March 1-8, Big Sky, MT, 2008.

[13] Z.S. Ma, and A. W. Krings, *Competing Risks Analysis of Reliability, Survivability, and Prognostics and Health Management (PHM)*, Proc. IEEE AeroSpace Conference, March 1-8, Big Sky, MT, 2008.

[14] Z.S. Ma, and A. W. Krings, *Multivariate Survival Analysis (I): Shared Frailty Approaches to Reliability and Dependence Modeling*, Proc. IEEE AeroSpace Conference, March 1-8, Big Sky, MT, 2008.

[15] Z.S. Ma, A. W. Krings, and R. E. Hiromoto, *Multivariate Survival Analysis (II): An Overview of Multi-State Models in Biomedicine and Engineering Reliability*, IEEE International Conference of Biomedical Engineering and Informatics, (BMEI 2008), 27 - 30 May, Sanya, Hainan, China, 2008.

[16] Z.S. Ma, and A. W. Krings, *Bio-Robustness and Fault Tolerance: A New Perspective on Reliable, Survivable and Evolvable Network Systems*, Proc. IEEE AeroSpace Conference, March 1-8, Big Sky, MT, 2008.

[17] Z.A. Ma, and A. W. Krings, *Spatial Distribution Patterns, Power Law, and the Agent-based Directed Diffusion Sensor Networks*, Sixth Annual IEEE International Conference on Pervasive Computing and Communications, (PerCom 2008), March 17-21, Hong Kong, 2008.

[18] Z.S. Ma, and A. W. Krings, *Insect Population Inspired Wireless Sensor Networks: A Unified Architecture with Survival Analysis, Evolutionary Game Theory, and Hybrid Fault Models*, IEEE International Conference of Biomedical Engineering and Informatics, (BMEI 2008), 27 - 30 May, Sanya, Hainan, China, 2008.

[19] N. R. Mead, R. J. Ellison, R. C. Linger, T. Longstaff, and J. McHugh, *Survivable Network Analysis Method*, Technical Report CMU/SEI-2000-TR-013, Software Engineering Institute, Carnegie Mellon, 2000.

[20] The PASIS project, Engineering Survivable Storage, Carnegie Mellon University, http://www.pdl.cmu.edu/Pasis/

[21] L.F.G. Sarmenta, *Sabotage-Tolerance Mechanisms for Volunteer Computing Systems*, Future Generation Computer Systems, Elsevier Publishing, No. 4, Vol. 18, 2002.

[22] P. Thambidurai, and Y.-K. Park, *Interactive Consistency with Multiple Failure Modes*, Proc. 7th Symp. on Reliable Distributed Systems, Columbus, OH, pp. 93-100, Oct. 1988.