Surviving Attacks and Intrusions: What can we Learn from Fault Models

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

Zhanshan (Sam) Ma
University of Idaho
Moscow, ID USA
ma@vandals.uidaho.edu

Abstract

When designing or analyzing applications or infrastructures with high reliability, safety, security, or survivability demands, the fundamental questions are: what is required of the application and can the infrastructure support these requirements. In the design and analysis of fault-tolerant systems, fault models have served us well to describe the theoretical limits. But with the inclusion of malicious acts, the direct application of fault models has exposed limited applicability. However, we can take advantage of the powerful fault models if we defer their direct application from the events that lead to faults, that is, the fault causes, and instead focus on the effects. This way one can avoid questions referring to the meaning of fault models in the context of previously unsuitable faults like Trojan horses or Denial of Service (DoS) attacks. Instead, we can use fault models at the level of abstraction where the application maps on the infrastructure.

In this paper fault models are discussed in the context of system survivability and malicious act. It is shown that these models can be used to balance the demands put on the application and the capabilities of the underlying infrastructure. Active *and* imposed *fault descriptions are defined that allow to match the mechanisms that provide survivability to the application with the infrastructure-imposed limitations. By defining a system as a collection of functionalities, individual functionalities and their associated fault descriptions can be analyzed in isolation.*

1 Introduction

In the design of systems that are subjected to security and survivability threats, one cannot directly apply many of the models and approaches used in fault tolerant systems design. The main reason is that for these models and approaches that have proven so successful in the design of dependable systems, the fault causes and their distributions do not translate. For example, it is straightforward to derive the fail rates of a component as the result of mechanical stress of breakdown over time. However, it is very difficult, if not impossible, to determine fail rates when a system is subjected to malicious act resulting in uncertain, unobserved, or unobservable events. This means that designing secure and survivable systems is much harder than designing fault-tolerant systems, a fact that has been acknowledged by the security and dependability community at large. Fault-tolerant systems are at the core of modern societies and these systems were largely designed for fault-tolerance and not with malicious act in mind. Thus, given the ever-increasing dependence of society on computerized systems, e.g., control of our critical infrastructures or environmental monitoring, understanding 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, as reviewed in [8, 12]. However, there have only been modest advances related to analyzing systems subjected to malicious acts or unpredictable events. At the core of the problem is the function representing the fail rate, often also called hazard function. This function ranges from a simple constant to complex functions of variables and covariates, with vast differences in the complexity of computability.

2 Background

Since the research discussed here addresses different aspects of security, survivability and fault-tolerance, some background is necessary to understand the main issues. Some of the preliminary information is drawn from [9].

2.1 Design for Testability and Survivability

Why is it so difficult to design systems that are fault-tolerant, secure, and survivable? Let's consider the evolution of integrated circuits and an approach that was called *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, rather than relying on testing circuits that were not designed with testability in mind.

Systems with high security and survivability demands suffer from the same phenomena. 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 [8]. If one could fully test and verify a system, then this would not be an issue. However, if completely testing integrated circuits is already NP hard, then there is no hope for complex systems. For example, whereas integrated circuits have as complex building blocks flip-flops, software has many complex mechanisms for branching, the simplest being a branch based on the state of a single variable. Thus, for complex software and hardware systems design for survivability is not a choice, but should be a requirement. 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".

2.2 Fault Models

Faults have been classified by their behavior. Many different taxonomies exist and typical terms include crash fault, benign fault, value fault, or malicious fault. The faults may be transient, intermittent, or permanent. 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. For example consider b benign faults in a set of N components. As long as a functionality f_i requires only one component to function properly, then f_i can function as long as one component survives, i.e., f_i needs at least $N = b + 1$ components. The assumptions on a benign fault are very restrictive, as benign faults imply that they are globally diagnosable and do not affect any other components. The fault model described in [10] made no assumptions about the fault behavior, and it was shown that to tolerate m malicious faults f_i needed at least $N = 3m + 1$ components. In the design of dependable systems, this fault model however seemed overly conservative since it assumed that every fault is malicious. In order to introduce more flexibility and to combine different fault types into one fault model, so-called hybrid fault models were derived. Hybrid fault models consider multiple fault behavior and cardinality. The mix of faults may range from benign, symmetric to asymmetric faults [20], and include potential transmissive and omissive behaviors [2]. The basic idea behind these models was to (1) specify the expected fault types and their count and (2) to derive the algorithms and redundancy management system to tolerate the fault mix. For example, using the model of [20] one may have a system specification that needs to tolerate a specific mix of faults, e.g., 3 benign faults, 1 symmetric fault and 1 asymmetric fault. Symmetric faults are value faults, where a wrong value is equally perceived by all other modules. Asymmetric faults make no restrictions on the values received by other nodes.

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, 11]. Thus, when considering survivability, one may (1) 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, or (2) focus on the effect of a fault, rather than the fault cause. For some faults this may create philosophical arguments, e.g., should a trojan horse that has not been triggered be ignored, since no fault has been introduced yet? In this research faults are always viewed in the context of what they are capable of producing, regardless of time.

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.

3 Survivable System Definition

In the definition of a survivable system using the notion of fault models, the reader should not make the mistake of viewing this definition as an attempt to simply argue that a survivable system can be expressed as a traditional fault-tolerant system. As indicated above, this view does not work due to the fact that survivable systems are expected to deal with unpredictable, unobserved, or unobservable events. In fact, the hazard function for survivable systems is much more complex.

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 with respect to the specific functionality. Thus, F_i indicates the fault types that f_i is designed to tolerate. However, F_i says nothing about how many faults of a specific type can or should be tolerated. For example, consider a communication service f_i that supports authentication. First, let's consider the case in which we assume that authentication is unbreakable or otherwise resilient to compromise. For this case the resulting fault description is $F_i = (b)$. The fault description indicates that attacks against authentication always result in benign faults, as indicated by parameter b, and since authentication is viewed as non-compromisable, *all* faulty values can be expelled. In this particular case there is thus no bound on the number of benign faults that can be tolerated. Note that this interpretation of a benign fault is very different from that usually used in dependability, where benign faults are typically associated with a component that fails in a benign manner, and as long as there is one functioning component, all is fine. However, as indicated before, in this research we look at the effects of the fault, not what caused it. This presents the departure from the definition of a fault in the dependability community, where a fault is a physical flaw that may produce an error that in turn may cause failure. Under that view a denied authentication is not seen as a "fault", it is simply a feature of the authentication mechanism. A non-value-based example of a benign fault is the case where the authentication system crashes in a benign fashion.

Next consider a second case where authentication has been compromised, e.g., due to a security breach. Now the previous model for F_i does not hold anymore, i.e. $F_i = (b)$ is unsuitable. Therefore we need a model capable of addressing the faults possibly resulting from an attack against a compromisable authentication system. Under the fault model described in [20], multiple fault types are considered, i.e., benign, symmetric and asymmetric faults. The resulting fault description for our compromisable authentication example is $F_i = (b, s, a)$, where b, s, and a indicate the potential of benign, symmetric, and asymmetric fault-behavior respectively. Failed authentication attempts result in (1) benign "faults" if the attack is recognized, (2) symmetric (value) faults if the same faulty value is sent to all redundant authentication modules, and (3) and asymmetric (value) faults, if different faulty values are send to different redundant authentication modules. Note that the only failed authentication attempts that are recognized are those resulting in benign faults. Furthermore note that both symmetric and asymmetric faults are value faults. However, as indicated before, the difference is that in the first case the faulty value is consistently perceived by all redundant modules. Value faults are not detected or corrected unless the correct redundancy management mechanisms are in place. In the case of symmetric faults it has been shown that a redundancy level of $N \geq 2s + 1$ can tolerate the s value faults, and in the case of asymmetric faults Byzantine majority may be necessary, i.e., $N \geq 3a + 1$.

In both of the two cases discussed above authentication is assumed, but the assumptions on the effectiveness of this functionality have changed. It should be obvious that wrong assumptions, and thus infeasible fault descriptions F_i, will affect the survivability of the system. For example, the system with $F = (b)$ that relies entirely on its authentication mechanism will not be able to tolerate a value fault, should authentication be compromised. It is thus important that system designers carefully evaluate the risk to system survivability and the consequences of making false assumptions.

In order to manage the complexity of a system S we view the system as a collection of k functionalities, i.e.,

$$S = \sum_{i=1}^{k} f_i,$$

where each f_i has an associated fault description F_i. System survivability is a function of all F_i. Note that S is not a scalar resulting from the sum of values, it is a collection of functionalities similar to [5, 7]. The view of the system as a collection of functionalities allows for the evaluation and analysis of each functionality in isolation or in groups of functionalities. It should be noted that this isolation-based view is very similar to the determination of essential services and their associated essential components, which were determined in [19] by tracing scenarios through the system architecture. Whereas in their case this was extracted (or traced) from the system, in our case this is the actual definition of the system.

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 through many functionalities. Specifically, if $f_i = f_p \circ f_q$, i.e., if f_i is composed of f_p and f_q, then the resulting F_i depends on the fault types of its components, i.e., $F_i =$

$g(F_p, F_q)$. The composition simply indicates that f_i has components that assume F_p and others that consider F_q. However, it is important to see that the weaker fault description may be dominant. Thus, at this point there are no assumptions made about the composition function g. Typical compositions include series, parallel, or series-parallel constructs. Furthermore, no assumptions are made about the significance, or importance, of different fault descriptions. If there is a need to analyze the significance, then weights or weight functions can be associated with F_i. Here however, the focus is simply on realizing that a system definition based on functionalities allows us to take advantage in the choice of mechanisms and their theoretical susceptibility to diverse faults, as will be described later.

Fault description 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.

4 System Analysis

System analysis of complex systems has its roots in dependability analysis, or more specifically: reliability analysis. In the dependability community, the reliability $R(t)$ is the probability that the system works as specified, without failure, during the entire time period from $[0, t]$. Thus, system analysis boils down to the quantification of reliability under consideration of the fault model and the assumptions about the fault environment. Similarly, survivability analysis is an attempt to quantifying system survivability under the same considerations. 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 or hazard function, and the independence or interdependence of faults.

Perhaps the simplest 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. It is mostly suitable for many problems in the dependability community where no malicious act is assumed and faults are considered to be independent. Therefore, $R(t) = e^{-\lambda t}$ should only be used for the determination of the reliability of those components that are feasibly modeled by a constant fail rate, e.g., hardware reliability.

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., the aforementioned constant fail rates, independence-of-failures assumption, and the inability of effective censoring. Some research has therefore attempted to eliminate the fail rates altogether. For example in [12] a survivability analysis was presented that was based on the definition of survivability of the T1A1.2 group. A survivability analysis was shown that was not dependent on the fail rate induced by malicious act. Specifically, a steady state solution was used up to (but not including) some time t_f, the time at which the fault ocurred. Then a transient solution showed the dynamic recovery of the system. However, the time of the event that lead to the fault was not important and the steady state solution simply served as the starting probabilities of the second part, which was initiated by the malicious act at time t_f. The approach presented in [12] is very dependent on the application, which in this case was a telephone system. Furthermore the Markov model of the system was very regular.

The alternative of eliminating the fail rates is dealing with them. However, the mathematical models associated with non-constant hazard functions quickly become very complex, as will be addressed in Section 4.1.

No matter what models are used for the analysis itself, at the basis of any model is the fault description, i.e., the fault model considered. This will hopefully lead to an analysis approach based on functionalities f_i and the associated F_i, with potentially different assumptions about the hazard functions themselves, as they apply to the functionalities and their underlying infrastructure. This decomposition of the overall (combined) model is thus an attempt to *Design for Analyzability*, with the goal of using the lowest feasible complexity model for each functionality f_i.

The choices in the design specifications of functionalities have direct implications on other system parameters. For example, increasing reliability of a functionality may in turn reduce security or performance. Thus, 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 at CMU [22] 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.

4.1 Dealing with Complex Hazard Functions

As indicated before, one of the main problems dealing with malicious acts such as intrusions or other attacks are the different degrees of unpredictability. In [13] this was captured with the introduction of so-called UUUR events (Unpredictable, latent, Unobserved and Unobservable Risks). In order to assess the consequences of UUUR events, survival analysis with its "sister" fields competing risks analysis and multivariate survival analysis were introduced in [15] and [16] respectively. A three-layer survivability analysis architecture was introduced that consisted of tactical, strategic, and operational levels. This architecture allows to integrate reliability, hybrid fault models and survivability under a unified paradigm.

A comprehensive introduction of three-layer survivability analysis is beyond the scope of this paper. Here, we only briefly discuss one aspect of the tactical level, dealing with complex hazard functions. We completely skip the strategic and operational level modeling, which can be found in [13] and [17].

In the absence of UUUR events, the tactical level is largely equivalent to traditional reliability analysis. Indeed, the most fundamental definition in survival analysis is the survivor function, $S(t) = Pr(T > t)$, which has the exact same definition as the reliability function. The hazard function $h(t)$ and the cumulative hazard functions $H(t)$ even use the same terminology besides the common mathematical definitions. However, there are additional advantages from introducing survival analysis over the traditional reliability analysis. Major advantages of survival analysis are: (i) more flexible, time-variant and covariates-dependent hazard functions; (ii) built-in procedures to deal with censored events; (iii) multivariate failure beyond binary failure; and (iv) more effective modeling of dependent failure events through competing risks and shared frailty modeling. Details can be found in [13, 14, 15, 16].

Our interest is with regard to the hazard functions listed in the advantages of (i). Therefore we now briefly introduce several forms of hazard functions from survival analysis to show one of the advantages of adopting survival analysis.

The simplest type of hazard function is the constant hazard function, which is when the failure time follows exponential distribution. It takes the form

$$h(t) = \lambda. \tag{1}$$

With the original Cox Proportional Hazards Model [4], (PHM), the hazard function becomes time and covariates dependent:

$$\lambda(t, z) = \lambda_0(t)e^{Z\beta} \tag{2}$$

where Z is the vector of covariates, such as environment factors that influence the hazard function, and $\lambda_o(t)$ is the *baseline* hazard function.

The Cox PHM has been extended numerously, e.g., [18, 21]. Two simple extensions, the stratified Cox PHM and the Cox PHM with time-dependent covariates, are of particular interest and their applicability will be discussed below.

First, we introduce the stratified Cox PHM. Suppose there is a factor that occurs on q levels and for which the so-called proportionality assumption of PHM may be violated. The hazard function for an individual in the j-th stratum or level of this factor is

$$\lambda_j(t, z) = \lambda_{0j}(t)e^{Z\beta} \tag{3}$$

for $j = 1, 2, ..., q$. The baseline hazard function $\lambda_{01}(\cdot), ..., \lambda_{0q}(\cdot)$ for the q strata are permitted to be arbitrary and are completely unrelated.

The second generalization to the PHM is the Cox PHM with time-dependent covariates. Here the covariates Z depend on time t, i.e., $Z = Z(t)$. For unstratified PHM, the hazard function is

$$\lambda[t; z(t)] = \lambda_0(t)e^{Z(t)\beta} \tag{4}$$

and for stratified PHM it is

$$\lambda_j[t; z(t)] = \lambda_{0j}(t)e^{Z(t)\beta}, \quad j = 1, 2, ..., q. \tag{5}$$

With the increase of the hazard function complexity from Equation (1) to (5), their descriptive power and flexibility also increase. For example, the stratified PHM [Equations (3) and (5)] may be used to formulate a unified hazard function for various levels of security alerts, e.g., low, medium, and high.

The choice of the hazard function that can be used has significant implications on the complexity of the system analysis.

4.2 Model Changes and State Changes

The view of the functionality-based analysis model described above has several advantages (in theory):

1. Different functionalities can have different fault descriptions.

2. Different functionalities can utilize different hazard functions.

3. Each functionality may change its fault description and/or hazard function in time.

The first advantage is the flexibility to view the fault description of each functionality in isolation. This *design for analyzability* feature allows for ease of analysis.

The second advantage is significant since the type of hazard function has huge implications on the complexity of the analysis. Note that the term "complexity" is not to be interpreted as computational complexity, but as the complexity of the approach. Having the flexibility of selecting an appropriate hazard function for individual functionalities rather than for the entire complex system allows to study the effects of specific assumptions about failing rates. This way one can move in the trade-off space of simplicity of analysis versus accuracy of the hazard function. For example, the simple hazard function of Equation (1) is suitable for the analysis of certain functionalities, but it is not generally suitable for malicious act. From an analysis point of view it is desirable to select the least complicated hazard function suitable for each functionality. Thus, just because certain aspects of the systems may be subjected to UUUR events, the entire system does not have to use the complicated hazard model.

The third advantage is the flexibility to consider the analysis in different phases with potentially different hazard functions at different stages. This can be modeled using a state machine. Consider the three different threat levels, e.g., low, medium, and high, one may use the stratified Cox PHM model expressed with Equations (3) and (5). Such system may adapt to threat levels announced by some authority. The system hazard function may transition from one stratum to another when the threat levels change. For example, the sequence $h_i \mapsto h_k \mapsto h_j \mapsto h_i$ would represent a system that transitions through various states (strata), each with their respective hazard functions. The sequence could be from the state machine shown in Figure 1.

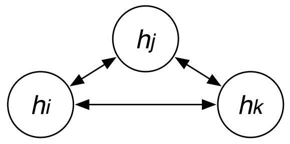

Figure 1. Thread Model State Machine

5 Fault Model Adaptation

As information becomes available that may change the system's landscape, changes to the system model or parameters need to be considered. Such adaptation is considered to be an integrated feature in any design for survivability. For example, in [6, 19] survivability was described in terms of Resistance, Recognition, Recover, and Adaptation. Adaptation implemented the mechanism to adapt the system to knowledge gained in the prior three phases.

Our interest is primarily in adaptation related to the fault model. Adaptation may be the result of diverse scenarios such as:

1. The fault description F_i is no longer valid due to a specific event. For example, intelligence suggests that authentication may be compromised. As a result fault types that are not reflected in F_i may occur.

2. The fault description of f_i should be strengthened by design. This may be the result of analysis indicating that a functionality may be the weakest link.

3. The infrastructure that f_i relied on has changed. The implications of this change on the fault description need to be evaluated. Of special interest is the case where the infrastructure may not be able to support tolerance of certain fault types.

In general, adaptation is viewed to address the dynamics of changes in fault descriptions F_i, which however has to be addressed in the context of the capabilities of the system or infrastructure that supports functionality f_i. Thus, we want to be able to point out if there is a mismatch between (1) what is assumed or implemented by f_i and (2) inherently theoretically possible in the system.

We define the *active* fault description as the fault model that the system currently subscribes to, i.e., the faults that f_i assumes to be able to tolerate or deal with. Thus for functionality f_i the fault description F_i is the active fault description, The specification of F_i is determined by the system designer or more specifically, by the designer of f_i. The real question that remains is whether the system's infrastructure can support this F_i.

In contrast to the active fault description, the *imposed* fault description, denoted by $\hat{F_i}$, is the fault model that the infrastructure of the system (or application) imposes on f_i. It encompasses those fault types that the system has to explicitly deal with by distinct mechanisms. For example, $\hat{F_i} = (b, s)$ indicates that for the given infrastructure benign and symmetric faults are possible and theoretically unavoidable. However, note that $\hat{F_i}$ does not list asymmetric faults. The reason is that the infrastructure is assumed to be capable of theoretically eliminating this fault type. An infrastructure that has an imposed fault description $\hat{F_i} = (b, s)$ is a broadcast network. In such a network asymmetric faults are not possible due to the fact that every module in the broadcast domain can see every message.

The active and imposed fault descriptions can serve the system designers to analyze the survivability of f_i. Let's consider the authentication example in a general network environment under the fault model of [20] and assume that messages are signed. First we assume that point-to-point communication, e.g., TCP/IP, is used. Furthermore, consider the two cases where 1) TCP/IP provides reliable transmission, and 2) when TCP may time out. With respect to the infrastructure, in case 1) this leads to an imposed fault description of $\hat{F_i} = (s, a)$, meaning that there are no benign faults. However, value faults, both symmetric and asymmetric, cannot be resolved without explicit mechanisms. If we consider case 2) in which TCP may time out, then $\hat{F}_i = (b, s, a)$. Next, let's consider the active fault descriptions. If our authentication scheme is assumed to be uncompromisable then $F_i = (b)$, otherwise it is $F_i = (b, s, a)$.

The interesting case in the example above is when authentication is compromised. Value faults cannot be dealt with unless the authentication mechanism is implemented to provide redundancy levels of $N \geq 2s+1$ and $N \geq 3a+1$ for symmetric and asymmetric behavior respectively. Note that in order to avoid common mode faults the redundant modules should be dissimilar. In order to deal with symmetric faults one needs a simple majority of unaffected modules. However, asymmetric faults do not only require a higher degree of redundancy, but also require that agreement algorithms be used. These algorithms typically work in rounds of message exchanges. The result is high message overhead in addition to the high component count. However, since in our example the imposed and active fault descriptions both contain s and a, there is no easy way around having to deal with these faults explicitly. For the system designer the choices seem clear: 1) one lives with the risk of authentication compromises, or 2) one pays the cost of module and message overhead. But how high is that cost? This depends on how many faults of type s and a one wants to tolerate. In addition, common mode faults need to be addresses and thus the cost of dissimilar components needs to be considered.

Design Changes: The advantage of working with imposed fault description is that it gives insight about what the infrastructure cannot inherently deal with. This allows for adaptation that can bring significant simplifications to the application. Consider the example above and assume that authentication may be compromised, i.e., assume that $F_i = \hat{F}_i = (b, s, a)$. The largest challenge is to avoid having to deal with costly asymmetric faults. However, the infrastructure cannot tolerate such faults implicitly and thus explicit mechanisms such as agreement algorithms must be used. Therefore, let's consider what changes can be made to the infrastructure in order to avoid asymmetric faults. With respect to networking this is actually quite simple. As indicated before, a broadcast environment cannot exhibit asymmetric behavior. Therefore, assume that point-to-point networking in authentication is eliminated and that broadcasting is used instead. Under the broadcast paradigm every node can "see" the same messages, so that Byzantine faults can be immediately detected. Now $\hat{F}_i = (b, s)$, and thus the application can provide simple mechanisms to take advantage of the imposed tolerance to asymmetric faults. Thus by observing the limitations of the imposed fault description an infrastructure-related change can make large improvements.

Adaptive Policies In the previous example the design of the authentication mechanism was motivated by low cost, which resulted in an active model considering only benign faults. If value faults are suspected, then the high cost of dealing with value faults, most significantly asymmetric faults, was required. However, in most applications the worse case behavior, e.g., broken authentication, may be only of importance during times of high threat levels. This suggest a security policy that is flexible and sensitive to the threat level. Such a policy would select the lowest overhead solution possible under a given threat level. In our authentication example this could mean using the benign model under normal situations and augmenting value faults if the threat level is high. Such "gear shifting" is not new and has been used in the context of agreement algorithms to reduce overhead [3].

Infrastructure Changes Lastly, if the infrastructure used by functionalities f_i changes, then one should consider if these changes have implications on the imposed fault description. If they do, then perhaps one can take advantage of this change. On the other hand it may mean that now the limitations of the infrastructure need to be compensated by more sophisticated solutions. An example of such "degeneration" is when a network changes from a broadcast to a point-to-point communication primitive.

In all the cases above, the careful analysis of F_i and $\hat{F}_i$ should be undertaken. Misjudging the fault model can render the application non-survivable.

6 Conclusions

A new view of fault models was presented that shifted away from the fault cause and instead focused on the effect of faults. This allows for the use of fault models in the analysis of systems operating in hostile environments. A general view of design for survivability was adopted. This implied that the application and infrastructure were viewed in concert in order to determine which fault models they required and supplied. By viewing a system as a collection of functionalities, each functionality could be separately analyzed. This simplified the determination of the active and imposed fault description, which was then used to determine a mapping between what the application functionality required and what the infrastructure could or could not support. In the latter case explicit solutions must be used to overcome the infrastructure induced limitations.

With respect to system analysis, the functionality-based view of the system allowed flexibility in the choice of hazard functions. Rather than using one model for the entire system, now each functionality can be analyzed using its appropriate hazard functions. The flexibility was then extended to the time domain, thus allowing to change the model over time in response to external changes.

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] Bar-Noy, A., D. Dolev, C. Dwork, and H. R. Strong, *Shifting Gears: Changing Algorithms on the Fly to Expedite Byzantine Agreement*, Information and Computation, Vol. 97, pp. 205-233, 1992.

[4] D. R. Cox, *Regression models and life tables*, Journal of the Royal Statistical Society, Series B (Methodological), Vol. 34, No. 2., pp. 187-220, 1972.

[5] S. Elbaum, and J. Munson, *Intrusion Detection Through Dynamic Software Measurement*, Proceedings of the Eighth USENIX Security Symposium, 1999.

[6] 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.

[7] A. Krings et. al., *A Two-Layer Approach to Survivability of Networked Computing Systems*, Proc. International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet, (SSGRR'2001), L'Aquila, Italy, Aug 06 - Aug 12, pp. 1-12, 2001.

[8] 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.

[9] Axel Krings, *Design for Survivability: A Tradeoff Space*, Proc. 4th Cyber Security and Information Intelligence Research Workshop, Oak Ridge National Laboratory, May 12-14, 2008.

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

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

[12] 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.

[13] Z.S. Ma, *New Approaches to Reliability and Survivability with Survival Analysis, Dynamic Hybrid Fault Models, and Evolutionary Game Theory*, Ph.D. dissertation, University of Idaho, Computer Science Department, 177pp., 2008.

[14] 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.

[15] 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.

[16] 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.

[17] Z. S. Ma, and A. W. Krings, *Dynamic Hybrid Fault Models and the Applications to Wireless Sensor Networks (WSNs)*, to appear in, The 11-th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, (ACM MSWiM 2008), 2008.

[18] T. Martinussen, and T. H. Scheike, *Dynamic Regression Models for Survival Data*, Springer Verlag, 466pp., 2006.

[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] 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.

[21] T. Therneau, and P. Grambsch, *Modeling Survival Data: Extending the Cox Model*, Springer Verlag, 2000.

[22] Jay J. Wylie, et.al., *Selecting the Right Data Distribution Scheme for a Survivable Storage System*, Technical Report, CMU-CS-01-120, Carnegie Mellon University, May 2001.