

# Survivability as Quality

Thomas A. Longstaff  
Network Systems Survivability Program  
Software Engineering Institute, Carnegie Mellon University  
tal@cert.org

## Abstract

Since 1997, there has been a small but dedicated community that has claimed that survivability was an area of research and development worthy of standing along side other more established areas such as security and dependable systems. In this position statement, I will focus on the question of why the survivability community has had such a difficult time finding a set of problems and techniques that are clearly identified with survivable systems, and propose a possible approach that places survivable systems clearly in the area of system quality.

## Position Statement

This will be the 4<sup>th</sup> event to bring together professionals with a common interest in survivability. Since 1997, there has been a small but dedicated community that has claimed that survivability was an area of research and development worthy of standing along side other more established areas such as security and dependable systems. At each event, the argument still rages: what is survivability *really* and are the problems associated with survivability unique or simply a subset of problems in other well established areas. In this position statement, I will focus on the question of why the survivability community has had such a difficult time finding a set of problems and techniques that are clearly identified with survivable systems, and propose a possible approach that places survivable systems clearly in the area of system quality.

Survivability is an attribute of physical devices that relates the ability of the device to continue to operate in a stressed environment. This is true of fighter jets under attack and microwave ovens on shake tables. In the world beyond software systems, survivability is enhanced through engineering solutions that allow a device to expand its range of environmental parameters in a measurable and controlled way. For example, a temperature probe can be made more survivable by extending its temperature extremes where a valid temperature will be signaled from the probe. A simple household thermometer will survive the extremes of hot days in the summer and cold days in the winter, but will not survive long in a bath of

molten lead or liquid nitrogen. It is simply not designed to survive under those conditions.

The above definition of survivability matches closely with the definition of dependability, i.e., the threats, attributes, and means by which dependability is attained<sup>1</sup>. Threats are further broken down as faults, errors, and failures, means through prevention, tolerance, removal, and forecasting, and attributes through availability, reliability, confidentiality, integrity, and maintainability. As described above, survivability is primarily concerned with external threats (from the environment), means that would expand the ability to respond to these external threats (prevention, and tolerance), and attributes related to performing the desired function (availability, reliability, confidentiality, and integrity). Excluded would be the definition of internal threats and means related to the development of the device, since these have little bearing on the operational nature of survivability.

Survivability has also been closely aligned with the area of security. The reason is that the most difficult environmental extreme to which to respond is the

---

<sup>1</sup> This definition is adapted from Avizienis, Laprie, and Randell, "*Fundamental concepts of computer system dependability.*" IARP/IEEE-RAS workshop on robot dependability: Technological challenge of dependable robots in human environments, Seoul, Korea, May 21-22 2001.

intelligent adversary. The intelligent adversary attacks a system primarily to reduce or eliminate one of the three traditional security attributes: confidentiality, integrity, or availability. While these are also included in the above description of dependability, it is commonly accepted that a specialization in security is helpful in the development of specific engineering solutions and measurement techniques focused on vulnerabilities that affect the security attributes. In addition, much of the research in other attributes in dependability has relied on statistical independence of faults (the assumption that there is no causal relationship between two independent faults), while security has always assumed that an intelligent adversary could and will coordinate seemingly unrelated faults in order to achieve a reduction in one of the security attributes.

One way of viewing much of the research in the dependable systems and security research to date is to observe that the bulk of the research in dependable systems has been *internal* to the defined system, while security has largely been associated with the interface between a protected resource and an external threat<sup>2</sup>, that is, *external* to the system under protection. Neither area is *exclusively* devoted to this, of course, but the preponderance of literature supports these foci as the main solutions proposed and implemented. Under this view of the main thrusts of each area, survivability appears to take a third position.

In thinking back to the results of the original examples, survivability of physical devices are measured in terms of both the system behavior and the environment under which the survivability measurement was made. It would seem that the survivability of these devices uses an internal measure of performance (similar to those generated for dependable systems) from a malicious interface attack (similar to those with a security focus). Engineering solutions that are effective in countering

---

<sup>2</sup> Note that even in the case of “inside threats” for security, the solutions proposed for these threats separate a protected set of resources from the “insider,” thus still making the adversary external to the resource under protection.

these attacks are both protection similar to security solutions and fail-over solutions such as those popular with dependable systems.

So if we are to use attacks, ala security breaches, to define the environment, how can these attacks increase the range of environmental measures we engineer our systems to accept as part of the normal operational environment?

Although we use the word “attack” to describe the activity of intelligent adversaries, it is difficult to explicitly and completely characterize all of the activity of an adversary into a single category. Dozens of vulnerability and attack taxonomy publications have addressed this problem with no clearly definitive result. There are many reasons why this is difficult. First of all, much of adversarial activity is focused on gathering information and to a potential victim site, this activity cannot be distinguished from normal activity. Frequently, this information gathering takes place with 3<sup>rd</sup> parties that have nothing at all to do with direct access to the potential victim site. Yet this collection of information is crucial in planning a successful attack and often distinguishes the intelligent adversary from the simple “hacker<sup>3</sup>.”

Secondly, there are many possible motives, methods, and results of an attack. An adversary may choose to simply exhaust the publicly available resources (such as bandwidth or buffer size), exploit a vulnerability that crashes the system, or use a back door to gain unauthorized access to the system. There are many more of these variations, some of which are described in the Sandia report, “A common language for computer security incidents”<sup>4</sup>. Finally, it is unclear whether to characterize these attacks as normal or abnormal behavior. In expanding the environmental

---

<sup>3</sup> I use the term “hacker” in this context to describe the uncoordinated activities of the malicious computer user that attacks targets of opportunity without significant preparation or forethought using tools and techniques readily available rather than custom designed for the victim.

<sup>4</sup> Howard J., Longstaff, T., [A Common Language for Computer Security Incidents](#). Sandia Report prepared by Sandia National Laboratories. October 1998 (pdf)

conditions of a device for its normal operation (such as the temperature probe as described above), we would expect that this expansion would be part of the normal operation of the device once it was deployed. For military aircraft, we expect that these aircraft must survive attack as part of their expected operation. However, we seem to believe that software intensive system attacks will be in some respect anomalous to our normal operating environment, which leads to systems that are not engineered to withstand these attacks as part of their normal operation. We should not be surprised, therefore, when these systems fail when deployed in an environment hostile to their construction.

A key difficulty here is that the environment of software intensive systems is changing rapidly and in ways that are unanticipated by their designers. This is very much like designing a physical device for an environment where we have no idea how this environment will change over time. There are two main engineering techniques used to address this difficulty. The first is to over-engineer the system to hopefully cope with the most extreme conditions we can imagine. In security, this was attempted through closely defining the operation of the system, then showing through formal methods that the system could not behave in any other way regardless of the input to the system.

The other method is to provide a flexible method of modifying the behavior of the system to cope with

new threats as they are discovered. This is very similar to anti-virus scanners that have dynamic configuration files that are augmented whenever a new virus is discovered.

In both cases, we need to know in detail what threats (or at least inputs from threats) will be included in the definition of the environment. Once the environment and functionality have been clearly defined, the real task at hand to achieve survivability is to improve the quality of the software intensive system so that we have some reasonable assurance that the appropriate behavior will be achieved.

So how will survivability be achieved in practice? Not through an endless review of whether this is a security problem or a dependable systems problem, but through the careful definition of the environment in which our systems will run and careful design and test of those system in the extremes of that environment. It will be achieved through additional tools and laboratories that can replicate a complex and malicious network environment to show that the behavior of our systems remains consistent by design, not by luck or by some mysterious “good engineering practice” that expects to design systems for undefined environments. In the end, the definition of quality in software remains the suitability of that software to operate in the *expected and anticipated* operational environment.