



## Preventing Failure: The Value of Performing a Single Point of Failure Analysis for Critical Applications and Systems

Laurence J. Wolf

To cite this article: Laurence J. Wolf (2004) Preventing Failure: The Value of Performing a Single Point of Failure Analysis for Critical Applications and Systems, Information Systems Security, 13:1, 51-54, DOI: [10.1201/1086/44119.13.1.20040301/80434.7](https://doi.org/10.1201/1086/44119.13.1.20040301/80434.7)

To link to this article: <https://doi.org/10.1201/1086/44119.13.1.20040301/80434.7>



Published online: 21 Dec 2006.



Submit your article to this journal [↗](#)



Article views: 67



View related articles [↗](#)



Citing articles: 1 View citing articles [↗](#)

# *Preventing Failure: The Value of Performing a Single Point of Failure Analysis for Critical Applications and Systems*

Laurence J. Wolf

**E**nsuring that a system is resilient requires not only good design and implementation, but also an independent engineering review in the form of a single point of failure analysis. This article presents guidelines for testing prior to any implementation and can prove very useful in meeting the requirements of the users.

In one wild week in June 2001, the New York Stock Exchange, NASDAQ Stock Exchange, and Orbitz — a major new travel site backed by the major airlines — all had very public miserable failures. The exchanges are the engine of capitalism, and they stalled out. These operations had spent untold millions to ensure high availability during their stated hours of operations. Yet somewhere in their systems, they still were able to find that magical point that could take everything out if it failed — and that point failed. This point can be small, like a

LAN hub or a server, or it can be large, like an entire data center. In the case of Orbitz, one of its failures happened to be a single fiber cable that was cut. At NASDAQ, it was a piece of network hardware; a few years ago, it was a squirrel chewing a power line that disabled NASDAQ's entire data center and shut it down. In any data system, there can be many points on which the system is dependent. Sometimes the point can be huge; the 9/11 attack on the World Trade Center (WTC) demonstrated that in a very sad way. Not only did many companies have critical data centers and trading operations located in or near the WTC, but a major telephone point-of-presence was wrecked as a result of the collapse of WTC 1 and WTC 7. Many firms in the area lost telephone service, as a result hurting physically untouched operations dramatically. A number of Wall Street firms were hurt

---

*LAURENCE J. WOLF is senior manger in Deloitte & Touche's Infrastructure & Security Systems Practice. He has 18 years of experience in systems, networks, software development, and resilient design, implementation and testing, as well as business continuity planning. He has spent the past 12 years working at and consulting for Wall Street financial firms and has a degree in engineering from the University of Pennsylvania.*

*In systems requiring resilience, that is, self-healing or fault-tolerant capabilities, there must in all cases be multiple parallel points where the system flow can exist.*

badly, three commodity exchanges lost their homes, and the American Stock Exchange lost its ability to operate. All four were forced to move to temporary locations.

In a high-availability system, there must never be a single point of criticality. In systems requiring resilience, that is, self-healing or fault-tolerant capabilities, there must in all cases be multiple parallel points where the system flow can exist. There can be no single point of dependency for the system from an engineering perspective; if there is, then a failure of that point will take down the system. This means that from the user's viewpoint, a single point of failure (SPOF) is defined as having a single point fail, whatever its size, that results in a user's inability to access a system with high-availability requirements. If this point exists, we have a potential single point of failure scenario.

When looking at a high-availability system, it can be seen that it has certain critical uptime requirements. It may need to run 24/7 like a retail Web site such as Orbitz, or in the case of the exchanges, 6½/5. Any system needing to meet an uptime goal must have resilience built into its design to enable failure to occur without losing the user application. The needed points of resilience vary in size and can be viewed in a hierarchical manner as follows:

- Element
- System
- Network
- System management
- Intrasite (room, floor, data center, etc.)
- Site-intersite

Take, as an example, a company that has located its mainframe operation in a single data center. The users of the system need data from that mainframe. If that site becomes unreachable for any reason (e.g., power outage or fire), then from the users' viewpoint, they see that site as a single point of failure. On the other end of the scale, if an application runs on a few servers that are all connected to the same eight port LAN switch, and that switch failed, the application is again

dead. A data center is a big point and a LAN switch is a much smaller point, but users only know that they cannot get what they want, when they want it — and that is the real point.

When a system is designed to be highly available, there is the presumption that the extra time and money spent have brought some value to the provider and the users of the system — that value being uptime in the face of failure. If it turned out that the system indeed failed to be as available as the requirements stated, then the value invested was wasted. Two things should be remembered about resilient systems:

1. Element failure ≠ system failure
2. Good system design ≠ good system implementation

This means that if a resilient system is properly designed and built, then single failures internal to the system do not affect the users. System engineers and administrators can detect and repair the issue and restore the system to full resilience without interfering with user functions. This also means that if the system is not built to specifications, then the unexpected will occur. Murphy's Law is fairly clear in that if a failure occurs, it will occur at the worst possible time and with the greatest of consequences. To be placed in a position such as this is not an ideal spot for any service provider. Confidence is lost, customers leave, users are unhappy, and current revenue is under threat.

Ensuring that a system is resilient requires not just good design and implementation, but also an independent engineering review in the form of a single point of failure (SPOF) analysis. This can take two forms: (1) a design analysis, and (2) an implementation analysis. Both of these must be viewed as engineering studies and treated as such with all of the proper talent, experience, and skills brought to bear to ensure it is properly done. In the first, the design and the requirements for the system are analyzed and design failures that yield an SPOF are determined. This can be a paper exercise that can be performed prior

to any implementation and can prove very useful in meeting the requirements of the users. In the second, the actual implementation of the system is reviewed. This is broader in scope and involves not just the system in question, but all of the things that support the system in the real world. An implementation review would physically examine the system from a number of viewpoints:

- Data network
- Management systems
- Low voltage wiring plants
- Timing services
- Servers and storage
- Telecommunications
- Power allocation
- Software resilience

As can be seen from the above pressure points, implementation issues can often occur that are not typically a part of system design. Rather, they are *inferred* to be done correctly. For example, if a server or router has dual power supplies, a systems designer would assume that both of these power supplies are connected and that they are each connected to separate power circuits. This maximizes the value gained by buying a dual power supply machine — both loss of a particular circuit and loss of a power supply are protected against. However, in the real world of implementation, things often go awry. At midnight, during the usual window of IT change times, the lowest-paid, least-trained, and most-tired individuals are performing the installations. They are given tasks to perform and deadlines to meet. So if the rack the new server is to be placed in has only one circuit, or the second circuit has no free outlets, or there is a lack of proper training or procedures, then both power supplies end up on one circuit or only one is plugged in.

There are many possible mistakes to be made and only a few correct ways to do the work. For example, let us look at a multiserver application. What if these servers are all on the same LAN switch, or what if they are distributed over multiple LAN switches but

these switches all connect to a single upstream LAN switch? For software resilience, have all the cases that could result in a software switchover from the hot machine to the standby system been tested — has it been configured according the design specifications? All of these cases are SPOF scenarios. These are the scenarios that put your business at risk.

Often, these are scenarios with simple resolutions. The difficult part is determining the requirements of the system, and then using those as the seed for the analytical methodology used to determine the best approach to an SPOF analysis. The best answer is not always obvious. Thinking out-of-the-box is critical in achieving the maximum resilience in systems. For example, take the case of two servers that make up an application. If both perform distinct functions and they are on separate power circuits, a quick look would say that is good. Upon further review, however, consolidating their power to one circuit would actually be better. If either server fails, the application is dead; so having two power circuits involved actually doubles the chance that a power failure could take down the application. On the other hand, if the machines were mirroring each other in a hot standby or load balancing situation, then the optimal solution is separate power circuits. Thus, a proper understanding of the system and its functionality is key to a proper SPOF analysis. This applies across the board for all system pressure points. It includes network and storage elements and other devices involved — the review must be holistic in nature.

So how is such an analysis actually performed? The key steps involve the following:

- Determination of the system(s) to be reviewed
- Discerning the locations of all machines that are part of the system
- Understanding the architecture and system goals

*Implementation issues can often occur that are not typically a part of system design. Rather, they are inferred to be done correctly.*

*The desired results might not ever be achieved if the people making the errors are the same people finding the errors.*

In a physical review of all the above pressure points, power and network are most critical; and if the application is a single-site application, then telecommunications may not apply. Other issues may be involved, so understanding the construct of the system is critical.

- For a software resilience analysis, create a test plan to test various cases in differing circumstances and apply it.
- Make careful notes of all findings; create a database if the data volume is large.
- Be willing to perform much of the work during off-hours if the system is critical.
- Have a plan to recover in the event of problems during the review; keep everyone informed.
- Think about the size of the points to be considered.
- Discover the SPOF scenarios in each of the pressure point areas or with any special circumstances by a careful study of the data.

Consider employing outside, independent consultants experienced in the field. Common repeated problems can indicate procedural issues. Follow up the study with

a root cause analysis study if common issues are present. This type of follow-up study can be a useful exercise in increasing the level of quality in the data center. The desired results might not ever be achieved if the people making the errors are the some people finding the errors. The next step is to go back and carefully fix the problems in a controlled manner. Then move on to procedural and training fixes. There can be many points of process failure, from requirements definition to design to implementation to maintenance.

The result will be superior system reliability in many ways. Not only the application, but also the surrounding systems will be improved, and system confidence will be at a high level. Think about reduction of risk to your operations and the value that brings to your operation.

Make sure that when failures do occur that the means (network and system management) are in place to detect, isolate, and restore them. Remember that when you have an unresolved point of failure in a critical system, until you repair it, you have an automatic SPOF scenario. ■

The views presented in this journal are those of the authors and do not necessarily reflect the views of the publisher, the (ISC)<sup>2</sup>, or of the journal's board of advisers.

*Information Systems Security* (ISSN 1065-898X) is published bi-monthly by Auerbach Publications, CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton, FL 33431. Editorial offices: Auerbach Publications, 29 W. 35th Street, 7th Floor, New York, NY 10001. Subscription rates: \$175/year in the U.S., U.S. possessions, and Canada. For prices elsewhere, please inquire. Periodicals postage paid at Boca Raton and other mailing offices. Printed in U.S. Copyright © 2004 CRC Press LLC. All rights, including translation into other languages, reserved by the publisher in the U.S., Great Britain, Mexico, and all countries participating in the International Copyright Convention and Pan American Copyright Convention. Product or corporate names may be trademarks or registered trademarks, and are only used for identification and explanation, without intent to infringe. This journal contains information obtained from authentic and highly regarded sources. Reprinted material is quoted with permission and sources are indicated. A wide variety of references are listed. Reasonable efforts have been made to publish reliable data and information, but the author and the publisher cannot assume responsibility for the validity of all materials or the consequences of their use. No part of this journal may be reproduced in any form—by microfilm, xerography, or otherwise—or incorporated in any information retrieval systems without the written permission of the copyright owner. Authorization to photocopy items for internal or personal use, or the personal or internal use of specific clients, may be granted by CRC Press LLC, provided that \$20.00 per article photocopied is paid directly to Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923 USA. The fee code for users of the Transactional Reporting Service is ISSN 1065-898X/04/\$20.00+\$0.00. The fee is subject to change without notice. For organizations that have been granted a photocopy license by the CCC, a separate system of payment has been arranged. Postmaster: Send address changes to *Information Systems Security*, 2000 N.W. Corporate Blvd., Boca Raton, FL 33431.