# CD-DocDB #5495

# Measurement of

# Service Availability

# and

# Evaluation of Algorithms for Selection of

# Target Service Availability

Keith Chadwick
ITIL Availability and Service Continuity Manager
23-Feb-2015

Abstract:

Options for Service Availability measurements together with various algorithms for determination of Target Service Availability are examined. The impact of High Availability (HA) and/or Redundant Service (RS) deployments on Service Availability is analyzed. Specific advantages and disadvantages are discussed for the individual algorithms and recommended algorithms are identified.

## Table of Contents

## Document Revision History

| Version | Date | Author | Comments |
|---|---|---|---|
| V0.1 | 09-Dec-2014 | Keith Chadwick | Initial Draft. |
| V0.2 | 23-Dec-2014 | Keith Chadwick | Draft circulated for comments. |
| V0.3 | 12-Feb-2015 | Keith Chadwick | Add Executive Summary |
| V1.0 | 23-Feb-2015 | Keith Chadwick | Publish to CD-DocDB |
| | | | |
| | | | |

## Executive Summary

An analysis of various algorithms for calculation of target service availability has been performed. The calculations based on these algorithms show that the actual service availability is likely to be bounded by the range:

[ Algorithm #2 Availability : Algorithm #1 Availability ]

This would correspond to the range [ 97.92% : 99.40% ] for the specific example in Table #5 (below) and the hypothetical "*service*". The specific availability estimates are:

| Availability Algorithm | Availability Result | Downtime Estimate (Days per Year) |
|---|---|---|
| #1 | 99.40% | 2.19 |
| #2 | 97.92% | 7.60 |
| #3 | 98.51% | 5.45 |
| #4 | 99.62% | 1.39 |
| #5 | 99.30% | 2.56 |
| #6 | 99.30% | 2.57 |

Specific advantages and potential drawbacks for each considered algorithm have been identified. Based on these factors, the recommendation for non-Highly-Available (non-HA) services is to either use Algorithm #5 or Algorithm #6 above, both of which incorporate the following factors:

1. Underpinning service list;
2. Underpinning critical/non-critical service evaluation;
3. Underpinning service rank.

For services with multiple instances and a much more "interesting" service "Layer Cake", the situation is much more complex. There are additional failure modes to consider including the possibility of correlated failures. Despite these considerations, if a multiple instance service is properly engineered, then it is possible that the service ensemble will offer target service availabilities in excess of 99.90% (for the case of two service instances), and if the number of service instances is greater than or equal to 4, then the target service availability can effectively be 100.00% (for any reasonable values of the probability of correlated failures).

## Introduction

If five people are asked "how to measure availability", they will likely give at least five different answers. Their answers may differ depending on:

- The types of availability measurements that they are familiar with;
- The types of availability measurement that they customarily use;
- If they are talking about correlated or uncorrelated outages;
- As well as other factors.

This document is an attempt to formally analyze the various types of availability measurement and provide recommended practices within the Fermilab ITIL/ISO20K Availability Management processes, procedures, and policies.

## Availability Measurement

In the simplest form, the *availability* percentage of a *service* is defined and measured based on the value of the *uptime* of a system divided by the sum of the values of *uptime* and *downtime*:

$$availability = \frac{uptime}{(\, uptime + downtime \,)} * 100$$

While the above formula is a good formal definition of *availability*, in practice it needs further refinement in order to be a useful metric. Furthermore, definitions of *availability* can include *historical service availability*, *service support availability*, and *target service availability*, together with other *availability* definitions, that are discussed below.

One aspect of measuring *downtime* for multiple services is that outages can be classified as being *correlated* or *uncorrelated*.

Examples of *correlated* outages include:

- An unplanned failure of required electrical service – in this case, multiple services that depend on the steady flow of electricity would be impacted; furthermore the unplanned nature of the electrical fault may result in additional damage to the electrical distribution infrastructure or computing and network hardware.

- A failure of an underpinning NAS storage service – in this case, multiple services that depend on the availability of the underpinning NAS storage service would be impacted.

Examples of *uncorrelated* outages include:

- A failure of a server in one data center and a failure of a second server in a second data center.

- A failure of a disk drive in a server and the failure of a network switch in a single data center.

## Historical Service Availability

*Historical service availability* is the calculation of the *availability* value for a *service* based on previous (past) measurements of *service* specific *uptime* and *downtime*. A The analysis by Adam Walters and Kim Kasza for the availabilities of the various computer rooms that is published in the CCD Quarterly Reports[1] is an excellent example of historical service availability measurements.

However, just as in a stock market prospectus, "*past performance is no guarantee of future results*". A service may have an uptime of 365 days and a downtime of 0 days over the past year, which would correspond to a *historical service availability* of 100.00% for the *service*, but if an unplanned power outage were to occur tomorrow, then in the future the *historical service availability* would be less than 100.00%.

## Support Availability

While *service availability* corresponds to the availability of the *service*, *support availability* corresponds to the *availability* of the *service* support organization to provide technical support for the *service*. It should be noted that *support availability* could significantly impact the observed *service availability*.

An example if this impact would be a service that has a published target availability of 99.00%, but only has support available between 8 AM and 5 PM, Monday through Friday, exclusive of observed holidays. The published service target availability of 99.00% would correspond to an "allowed" downtime of 3.65 days per year. However if the service went down at 6 PM on Friday evening, it might not be restored until 9 AM on Monday morning for a downtime of 6+24+24+9 = 63 hours = 2.63 days. Thus a single incident could correspond to 72% of the total "allowed" downtime per year.

[1] https://cd-docdb.fnal.gov:440/cgi-bin/ShowDocument?docid=5477

The situation becomes worse if the service failure occurs at the beginning of a long weekend (a weekend with an observed holiday immediately preceding or following), in this case the downtime could be 87 hours = 3.63 days, or greater than 99% of the total "allowed" downtime per year consumed in a single incident.

## Target Service Availability

*Target service availability* is the determination of an *availability* value for a *service* to publish to the customers of a service. Publication of a *target service availability* value has several benefits:

1. It is a form of customer expectation management;
2. It should incorporate any expected downtimes of any *underpinning services*;
3. It should incorporate the impact of *service support availability* and any required *service* maintenance periods;
4. It should be consistent with other similar *services*.

### Customer Expectation Management

In general, customers will not complain (in fact they probably won't even notice) when the actual delivered (i.e. *historical*) *service availability* is greater than the published *target service availability*. On the other hand, if the actually delivered *service availability* is less than the published *target service availability*, then customers will notice and likely voice (fully justified) concerns about the service delivery.

The previous paragraph may appear to argue that a service owner should publish an artificially low value for their target service availability, in order to avoid conflicts with their customer base. However, if the target service availability is unreasonably low, then customers may rightfully conclude that the service in question is potentially unreliable and the use of the service should be avoided.

Based on the considerations in the above paragraphs, service owners are cautioned to carefully consider the methods that they use in order to selected their published target availability.

### Incorporation of Underpinning Services

The *target service availability* will need to incorporate *availability* contributions from the set of *underpinning services*.

### Incorporate Service Specific Support Availability and Maintenance Periods

The *target service availability* will need to incorporate *availability* contributions from any *maintenance* by the *service provider* to the *service* itself.

### Consistency with Similar Services

Customers have a reasonable expectation that similar services will have similar (but not necessarily identical) target service availabilities. Examples include:

- Authentication – MIT Kerberos, Windows Active Directory, Kerberos Certificate Authority
- Database Hosting - Oracle, Postgres, MySQL
- Storage – Network Attached Storage (NAS), Storage Area Network (NAS)

For the set of Authentication services, the service owner has deployed the infrastructure that is required to operate all of the various authentication methods in a highly available distributed infrastructure. Based on this infrastructure, together with operational experience, the customers have come to expect, and the service owners have committed to provide an extremely high level of target service availability across the suite of Authentication services.

For the set of Database Hosting services, depending on the business use case of the specific database, the service owner has deployed the infrastructure in a redundant infrastructure in the Feynman Computing Center building. Based on this deployment, there are still cases where a single event could result in the (near) simultaneous unavailability of both the primary and secondary infrastructure. The greatest support is offered for Oracle databases, the open source Postgres and MySQL databases are supported at a lower service level commitment.

For the set of Storage services, there are two offerings – the first is the fibre channel connection based Storage Area Network (SAN), and the second is the NFS over TCP/IP connection based Network Attached Storage (NAS). The NAS implementation uses redundant BlueArc storage servers to "front-end" SAN connected storage.

### Key Takeaways

The key takeaway for determination of *target service availability* is that the published value must be:

1. Based on the needs of the customer;
2. Based on the published *target service availabilities* of any underpinning *services*;
3. Incorporate any *service* specific maintenance periods;

4. Be consistent with other similar *services*.

## Target Service Availability Calculation Algorithms

Consider the trivial case of a service that does not have any underpinning service requirements, then based on a realistic estimate of the expected downtime for the forthcoming period, the simplest availability prediction would be:

$$availability = \frac{(\ 365\ days - expected\ downtime\ )}{365\ days} * 100$$

In the real world however, it is an extremely rare service that does not have any underpinning service requirements, other than the base data center service.

In Table 1 below, an *example* set of *underpinning services* is shown.

| Table 1 –Underpinning Services | | | |
|---|---|---|---|
| **Underpinning Service** | **Published Target Availability** | **Expected Downtime (Days/year)** | **Comments** |
| Data Center | 99.70% | 1.10 | Building, Electricity, HVAC, etc. |
| Network | 99.90%[2] | 0.37 | Estimate based on redundant Data Communications infrastructure |
| Authentication | 100.00% | 0.00 | Distributed redundant infrastructure |
| Network Storage | 99.50%[3] | 1.83 | Equipment on FCC2 |
| Database Hosting | 99.50%[4] | 1.83 | Equipment on FCC2 |

We will now explore the impact of various Algorithms to select the *published target availability*.

### Algorithm #1 – Minimum of Underpinning Target Service Availabilities

Algorithm #1 – Select the **minimum** *published target availability* from the list of *underpinning services*; incorporate an estimate for any *service* specific downtime availability to determine the *overall target availability* to publish for this specific service.

This *availability target calculation* algorithm determines the *availability* of a hypothetical "*service*" that depends on all of the *underpinning services*, listed in Table

[2] This target availability does not appear to incorporate contributions from the underpinning service(s).

[3] This target availability appears to incorporate contributions from the actual service as well as contributions from the underpinning services.

[4] Ibid.

#1 above, based on the underpinning service downtime corresponding to the minimum *published target availability* value in the list of *underpinning services* (in the above list that would correspond to a value of 99.50% or 1.83 days of downtime), and then add an estimate for the "*service*" specific downtime:

Target "service" downtime = MAX(set of underpinning service downtime estimates = 1.83) +
("*service*" specific expected downtime = 0.37).
= 2.20 days

Target "*service*" availability = (365.00 – 2.20 ) / 365.00 = 99.40%

Or an alternative calculation based on availability percentages:

Target "*service*" availability = MIN(set of underpinning service *availabilities* = 99.50%) *
( "*service*" specific expected *availability* = 99.90%).
= 99.40%

The downside of algorithm #1 is that it may result in a potential **overestimate** of the target "*service*" unavailability, since it does not take into account additional uncorrelated contributions to availability from the other underpinning services. Despite this issue, algorithm #1 is a good algorithm to determine a *maximum* value for the target "*service*" availability.

### Algorithm #2 – Multiplication of Underpinning Target Service Availabilities

Algorithm #2 – Calculate the product of the set of published *target service availabilities* from the list of *underpinning services*; multiply the result by an estimate for any *service* specific downtime availability.

This *availability target calculation* algorithm determines the *availability* of a hypothetical "*service*" that depends on all of the *underpinning services*, listed in Table #1 above, by multiplication of the individual *underpinning service availability targets*:

Target "*service*" availability = (Target availability of Data Center = 99.70%) *
(Target availability of Network = 99.90%) *
(Target availability of Authentication = 100.00%) *
(Target availability of Network Storage = 99.50%) *
(Target availability of Database Hosting = 99.50%) *
(100.00% – "*service*" specific downtime = 99.90%) .
= 97.92%

Again, the alternative calculation based on estimated downtime is:

Estimated "*service*" downtime = (Target downtime of Data Center = 1.10) +
(Target downtime of Network = 0.37) +
(Target downtime of Authentication = 0.00) +
(Target downtime of Network Storage = 1.83) +

(Target downtime of Database Hosting = 1.83) +
( "*service*" specific downtime = 0.37) .
= 5.50 days

Target "*service*" availability = ( 365.00 – 5.50 ) / 365.00 = 97.92%

The downside of algorithm #2 is that it may result in a significant **underestimate** of the target "*service*" unavailability since this algorithm does not account for any correlation in the downtimes. Furthermore, it has the drawback that the calculation result may be significantly lower that the customer (rightly or wrongly) expects. Despite these issues, algorithm #2 is a good algorithm to determine a *minimum* value for the target "*service*" availability.

## Additional Underpinning Service Considerations

In the specific example presented in Table #1 above, all of the underpinning services are required in order for the hypothetical "*service*" to be available to the consumer. There are cases where a "*service*" may have *non-critical underpinning services*. A *non-critical underpinning service* is an *underpinning service* that if it is not *available*, it may not *directly* impact upper level *service availability*.

Examples could include a "Backup and Restore" service or a "Printing" service as "non-critical" underpinning services:

**Table 2 – Incorporation of Non-Critical Underpinning Services**

| Underpinning Service | Published Target Availability | Critical Underpinning Service? | Comments |
|---|---|---|---|
| Data Center | 99.70% | Yes | Building, Electricity, HVAC, etc. |
| Network | 99.90%* | Yes | * = Estimate based on redundant core network infrastructure |
| Authentication | 100.00% | Yes | Distributed |
| Network Storage | 99.50% | Yes | Equipment on FCC2 |
| Database Hosting | 99.50% | Yes | Equipment on FCC2 |
| Backup and Restore | 99.50% | No | Equipment on FCC2 |
| Printing | 99.50% | No | Equipment on FCC1,2,3 |

In the specific example presented in Table #2, under most service continuity situations, the temporary unavailability of the "Backup and Restore" or the "Printing" services would not impact the availability of other services that have the "Backup and Restore" or "Printing" service listed as an underpinning service. The unavailability of the non-critical services would impact a "higher level" service only if the "higher level" service needed the particular capabilities of the non-critical service to restore the "higher level" service.

In actual practice, recovery of information from the "Backup and Restore" service is infrequently required, since most "higher level" services can be restored by just rebooting the servers that implement the "higher level" service once the set of required underpinning services are available. The “Backup and Restore” service would only be required if the servers that implement the “higher level” service had a disk or file system failure that occurred co-incident with the outage of the “Backup and Restore” service.

### Algorithm #3 – Calculation based on Product of Required Target Availabilities

Algorithm #3 – Filter on Critical Underpinning Service; calculate the product of the published underpinning service target availabilities.

Using the data from Table #2, this algorithm results in a target availability calculation of:

$$98.61\% * 99.90\% = 98.51\%$$

Depending on the method used to determine the published target availabilities for the underpinning services, algorithm #3 has the following potential issues:

1. Incorporation of redundant downtime estimates;
2. Does not address any correlation of potential sources of downtime.

### Algorithm #4 – Calculation based on Average of Required Target Availabilities

Algorithm #4 – Filter on Critical Underpinning Service; calculate the average of the published underpinning service target availabilities.

Using the data from Table #2, this algorithm results in a target availability calculation of:

$$99.72\% * 99.90\% = 99.62\%$$

Algorithm #4 has a significant issue – this algorithm can generate a target service availability estimate that is greater than the published target availability of the underpinning services.

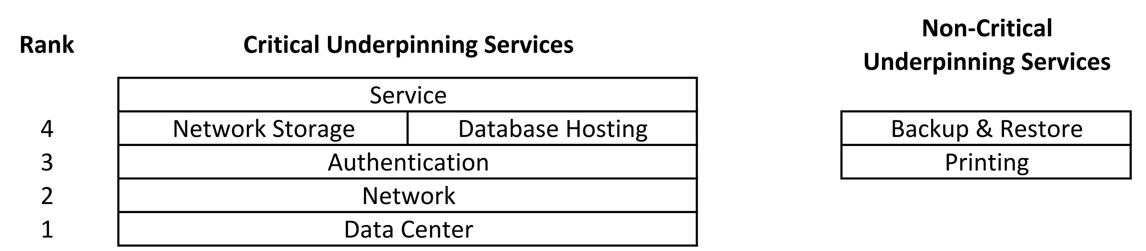
### Algorithm #5 – Calculation based on Service Rank and Requirements

Algorithm #5 – Filter on critical underpinning service; based on the service “rank”, calculate the underpinning service contribution to the total unavailability of the set of underpinning services.

| Table 3 – Ranking of Critical Services | | | | |
|---|---|---|---|---|
| **Underpinning Service** | **Published Target Availability** | **Critical Underpinning Service?** | **Service Rank** | **Comments** |
| Data Center | 99.70% | Yes | 1 | Building, Electricity, HVAC, etc. |
| Network | 99.90%* | Yes | 2 | Estimate based on redundant Data Communications infrastructure |
| Authentication | 100.00% | Yes | 3 | Distributed |
| Network Storage | 99.50% | Yes | 4 | Equipment on FCC2 |
| Database Hosting | 99.50% | Yes | 4 | Equipment on FCC2 |
| Backup and Restore | 99.50% | No | n/a | Equipment on FCC2 |
| Printing | 99.50% | No | n/a | Equipment on FCC1,2,3 |

The graphic below shows another way to present the information in the above table:

Image 1 – Service "Layer Cake" Model




The specific formula for algorithm #5 is:

Estimated "*service*" downtime = ((Estimated downtime of Data Center = 1.10 days) * (1/1)) +
((Estimated downtime of Network = 0.37 days ) * (1/2)) +
((Estimated downtime of Authentication = 0 days) * (1/3)) +
((Target availability of Network Storage = 1.83 days) * (1/4)) +
((Target availability of Database Hosting = 1.83 days) * (1/4)) +
("*service*" specific downtime = 0.37 days) .
= 2.56 days

Estimated "service" availability = ( 365.00 – 2.56 ) / 365.00 = 99.30%

The rank value is used in an attempt to compensate for any correlated downtimes.

### Algorithm #6 – Calculation based on Estimated Downtime per Year

Algorithm #6 – Filter on critical underpinning service; sum the underpinning service yearly downtime estimates, where there are multiple services with the same "rank" only add the largest value, add the individual service yearly downtime estimate, and then calculate the target service availability.

**Table 5 – Yearly Estimated Downtime**

| Underpinning Service | Published Target Availability | Critical Underpinning Service? | Rank | Yearly Estimated Downtime (Days/year) | Comments |
|---|---|---|---|---|---|
| Data Center | 99.70% | Yes | 1 | 1.07 | Building, Electricity, HVAC, etc. |
| Network | 99.90%* | Yes | 2 | 0.37 | Estimate based on redundant Data Communications infrastructure |
| Authentication | 100.00% | Yes | 3 | 0.00 | Distributed Redundant Infrastructure |
| Network Storage | 99.50% | Yes | 4 | 0.73 | Equipment on FCC2 |
| Database Hosting | 99.50% | Yes | 4 | 0.73 | Equipment on FCC2 |
| Backup and Restore | 99.50% | No | n/a | 0.73 | Equipment on FCC2 |
| Printing | 99.50% | No | n/a | 0.73 | Equipment on FCC1,2,3 |

Based on the data in Table 5, the specific formula for algorithm #6 is:

Estimated "*service*" downtime = (Estimated downtime of Data Center = 1.10 days) +
(Estimated downtime of Network = 0.37 days ) +
(Estimated downtime of Authentication = 0.00 days) +
(Target availability of Network Storage = 0.73 days) +
("*service*" specific downtime = 0.37 days) .
= 2.57 days

Estimated "service" availability = ( 365.00 – 2.57 ) / 365.00 = 99.30%

## The Impact of High Availability and/or Redundant Service Deployments on Availability Measurement

From Wikipedia[5], there are three principles of high availability engineering (or continual service delivery). These principles are:

[5] http://en.wikipedia.org/wiki/High_availability

1. Elimination of single points of failure. This means adding redundancy to the system so that failure of a component does not mean failure of the entire system.

2. Reliable crossover (failover). In multithreaded systems, the crossover point itself tends to become a single point of failure. High availability engineering must provide for reliable crossover.

3. Detection of failures as they occur. If the two principles above are observed, then a user may never see a failure. But the maintenance activity must.

Computing at Fermilab has many examples where high availability engineering has been used to deploy services. Here is a selected list:

1. There are multiple buildings and data centers;

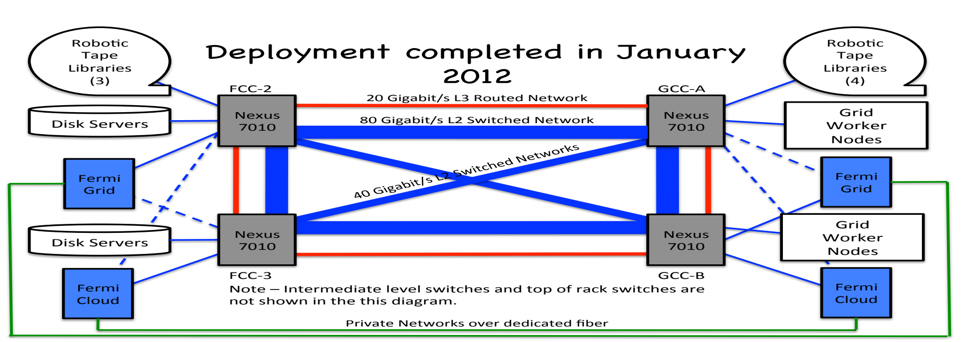
2. There is a distributed redundantly connected network infrastructure between the buildings and data centers, there are multiple Domain Name Service servers distributed across the network (refer to Image #2 below);

3. There are multiple offsite network connections using independent routes to independent ESnet and Internet 2 connection (refer to Image #3 below);

4. The Authentication services have been deployed with multiple servers across multiple buildings;

5. Other high availability services, such as FermiGrid-HA2[6] have been deployed using the above infrastructure.

Image 2 – Redundant Network Deployment




[6] http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=3739 and http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=4275

Image 3 – Redundant Offsite Network Connections

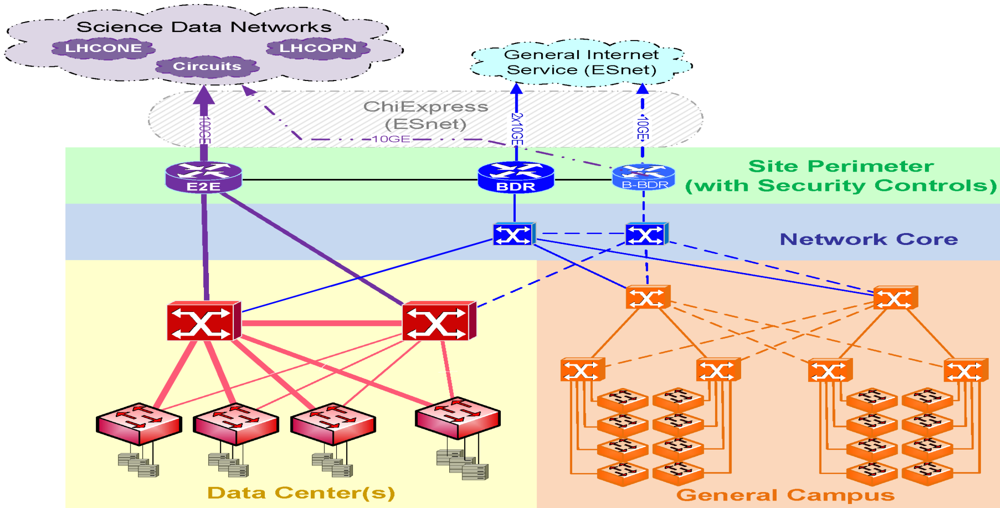



Image courtesy of Phil DeMar.

So for a "*service*" that uses the distributed and redundant capabilities of the infrastructure that is deployed at Fermilab, the resultant "Layer Cake" model for the service is shown in Image #4 below:

Image 4 – Revised Layer Cake Model for Redundant Services

| Rank | Service Area | Critical Underpinning Services | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Service | | | | | | | |
| 3 | Authentication | Authentication | | | | | | | |
| 2 | Distributed Network | FCC-2 | FCC-3 | GCC-NRA | | GCC-NRB | | LCC | WH8FC |
| 1 | Data Centers | FCC-2 | FCC-3 | GCC-CRA | GCC-TRR | GCC-CRB | GCC-CRC | LCC | WH8FC |

| Non-Critical Underpinning Services |
|---|
| Backup & Restore |
| Printing |
| Service Desk |

The question is how to best measure *service availability* and how to establish a *target service availability* across this distributed and redundant infrastructure?

As the Wikipedia article indicates, the best practice is to measure the availability of the high level redundant service and independently measure the availability of the individual services that are used to "assemble" the high level redundant service. Ideally this measurement should be performed independently of the service, and not solely rely on (any) service specific logs on the system that hosts the service.

In the example shown in Image #4 above, the ideal set of recommended measurements would include:

- Measurement of the physical availability of the individual data centers;
- Measurement of the network availability across the distributed network;
- Measurement of the authentication infrastructure availability;
- Measurement of the physical availability of the system(s) hosting the individual service instances;
- Measurement of the individual service instances that participate in the HA service;
- Measurement of the HA service instance.

In practice, some of the above measurements can be eliminated, since they may be inferred by other higher level measurements – a specific example is that the measurement of the physical availability of the systems hosting the service instance(s) via the network would be sufficient to measure the physical data center availability, the network availability, and potentially the authentication infrastructure availability (providing that the availability measurements use an authenticated method to access the remote service instance such as Kerberized ssh or http with SSL/TLS).

For the case of fully redundant HA services, the measured availability of the HA service instance at time "*t*" (providing that the HA service availability is not independently measured) can be derived from the Availability of the individual service instances [ 1, 2, … N ] at time "*t*":

$$\text{Availability (HA)}(t) = \text{MAX}(\ \text{Availability Service Instance 1}(t),\ \text{Availability Service Instance 2}(t), \ldots\ \text{Availability Service Instance N}(t)\ )$$

It is still best practice to independently measure the HA service availability in addition to the availability measurements of the individual service instances.

## Establishing Target Service Availabilities for Services with Multiple Instances

As the above section on Availability Measurement for HA services has shown, the measurement of service availability for HA services is significantly more complicated that the measurement of service availability for non-HA services. However establishing Target Service Availabilities for HA services can be significantly easier.

To begin with let us consider the case of a hypothetical HA service that is deployed at data centers “A” and “B” and only depends on the “data center” service (does not depend on network or authentication services). If the target service availability for the individual data centers “A” and “B” is 99.70, then the resultant combined data center target availability (AT) value for the set of the two data centers is:

AT = 99.70% + ( ( 100.00% - 99.70% ) * 99.70% )
= 99.70% + ( 0.30% * 99.70% )
= 99.70% + 0.2991%
= 99.99%

If rather than two data centers, there were “N” data centers, the formula for the availability target (AT) of any data center from the set of data centers without correlated failure modes would be:

AT = AT(DC1) + ( 100 – AT(DC1) ) * AT(DC2) +
( 100 – ( AT(DC1) + AT(DC2) ) * AT(DC3) + …
(100 – ( AT(DC1) + … + AT(DCN-1) * AT(DCN)

The results of these calculations are shown in Table #6 below:

**Table 6 – Impact of the Number of Data Centers on Data Center Availability**

| Number of Data Centers | Individual Data Center Availability Target | Combined Data Center Availability Target |
|---|---|---|
| 1 | 99.70% | 99.70% |
| 2 | 99.70% | 99.97% |
| 3 | 99.70% | Effectively 100.00%[7] |
| 4 | 99.70% | Effectively 100.00% |

The above calculations are valid providing that there are no correlated (common mode) failure mechanisms. If there were correlated failure mechanisms that could simultaneously impact data centers “A” and “B”, then the formula would become:

AT = 99.70% + ( ( 100.00% - 99.70% ) * ( 1 – correlated failure probability ) * 99.70%

So, if the correlated failure probability were 0.100 (1 in 10 instances or 10.0%), the resultant value would be:

AT = 99.70% + ( ( ( 100.00% - 99.70% ) * ( 1 – correlated failure probability ) * 99.70% )
= 99.70% + ( 0.30% * ( 1 - correlated failure probability ) * 99.70% )
= 99.70% + ( 0.30% * 0.90 * 99.70% )
= 99.70% + 0.26919%
= 99.97%

[7] To the precision of the calculations in a Microsoft Excel for the Mac 2011 spreadsheet.

And the results for N data centers with a correlation probability of 0.10 are shown in Table #7 (below):

| Table 7 – Impact of the Number of Data Centers on Data Center Availability with a Correlation Probability of 0.100 | | |
|---|---|---|
| **Number of Data Centers** | **Individual Data Center Availability Target** | **Combined Data Center Availability Target** |
| 1 | 99.70% | 99.70% |
| 2 | 99.70% | 99.97% |
| 3 | 99.70% | Effectively 100.00% |
| 4 | 99.70% | Effectively 100.00% |

In the case where the probability of correlated failures between any two data centers "x" and "y" is P(DCx,DCy), the formula becomes:

AT = AT(DC1) + ( ( 100 – AT(DC1) )* ( 1 - P(DC1,DC2) ) ) * AT(DC2) +
( ( 100 – ( AT(DC1) + AT(DC2) ) * ( 1- ( P(DC1,DC2) * P(DC2,DC3) ) * AT(DC3) + …
( 100 – ( AT(DC1) + … + AT(DCN-1) ) * ( 1 – (P(DC1,DC2) *… * P(DCN-1,DCN) ) * AT(DCN)

The results for N data centers with an Individual Data Center Availability Target of 99.00% ( 3.65 days downtime per year!) and a absurdly large correlation probability of 0.250 (25% chance of any given issue involving multiple data centers) are shown in Table #8 (below):

| Table 8 – Impact of the Number of Data Centers on Data Center Availability with Data Center Availability = 99.00% and Correlation Probability = 0.250 | | |
|---|---|---|
| **Number of Data Centers** | **Individual Data Center Availability Target** | **Combined Data Center Availability Target** |
| 1 | 99.00% | 99.00% |
| 2 | 99.00% | 99.74% |
| 3 | 99.00% | 99.94% |
| 4 | 99.00% | 99.98% |
| 5 | 99.00% | 100.00% |
| 6 | 99.00% | 100.00% |

From the above calculations, it is clear that based on any *reasonable* estimate of the correlated failure factor, that once the number of instances of a service is greater than or equal to 3, and definitely by 5, then the combined availability target of the ensemble of such services is *effectively* 100.00%.

Table #9 below lists the number of service instances of selected computing services and the effective target service availability using a correlation probability of 0.100 that is a consequence of the number of service instances:

| Table 9 – Effective Target Service Availabilities, Correlation Probability = 0.100 | | | |
|---|---|---|---|
| **Service** | **# of Service "Instances"** | **Individual Instance Target Service Availability** | **Effective Target Service Availability due to Multiple Instances** |
| Data Center | 6 | 99.70% | 100.00% |
| Network Core | 4 | 99.90% | 100.00% |
| Network Offsite Access | 2 | 99.90% | 99.97% |
| Authentication (Kerberos) | 6 | 99.90% | 100.00% |
| Authentication (KCA) | 2 | 99.90% | 99.97% |
| Authentication (Windows) | 6 | 99.90% | 100.00% |
| Authentication (LDAP) | 4 | 99.90% | 100.00% |

## Availability and Service-Now Outage Records

Within Service-Now, service availability is measured through the use of Outage Records to determine the uptime percentage during the expected service availability window. There are (currently) three types of Outage records that may impact service availability measurements:

1. **Outage** - Any issue that results in service unavailability during the expected service availability window is an Outage and will negatively impact the service availability measurements.

2. **Planned Outage** - Any issue that results in service unavailability during an "agreed to maintenance window" is a Planned Outage and will not impact the service availability measurements.

3. **Degradation** - Indicates an issue affecting a service that does not result in a disruption of that service, or normal service operations from the customers' point of view. A degradation type outage record will not impact the service availability measurements. **NB:** *If an issue results in **any** service disruption, then it should be entered as a type = Outage rather than type = Degradation*.

Note: It is possible that a single issue may result in more than one outage record. An example of this would be the result of a failure of an individual LDAP Domain Controller. For the period between the failure and the time that Network Services performed the required DNS update, the LDAP service would be declared to be in an "Outage", and a corresponding "Outage" type Outage record would be created; following the DNS update and until the failed LDAP Domain Controller was fully repaired, the LDAP service would be declared to be in a degraded state, and a corresponding "Degradation" type Outage record would be created.

## Summary and Recommendations

An analysis of various algorithms for calculation of target service availability has been performed. The calculations above show that the actual service availability is likely to be bounded by the range:

[ Algorithm #2 Availability : Algorithm #1 Availability ]

This would correspond to the range [ 97.92% : 99.40% ] for the specific example in Table #5 above and the hypothetical "*service*". The specific availability estimates are:

| Availability Algorithm | Availability Result | Downtime Estimate (Days per Year) |
|---|---|---|
| #1 | 99.40% | 2.19 |
| #2 | 97.92% | 7.60 |
| #3 | 98.51% | 5.45 |
| #4 | 99.62% | 1.39 |
| #5 | 99.30% | 2.56 |
| #6 | 99.30% | 2.57 |

Specific advantages and potential drawbacks for each considered algorithm have been identified. Based on these factors, the recommendation for non-HA services is to either use Algorithm #5 or Algorithm #6 above, both of which incorporate the following factors:

4. Underpinning service list;
5. Underpinning critical/non-critical service evaluation;
6. Underpinning service rank.

For services with multiple instances and a much more "interesting" service "Layer Cake", the situation is much more complex. There are additional failure modes to consider including the possibility of correlated failures. Despite these considerations, if a multiple instance service is properly engineered, then it is possible that the service ensemble will offer target service availabilities in excess of 99.90% (for the case of two service instances), and if the number of service instances is greater than or equal to 4, then the target service availability can effectively be 100.00% (for any reasonable values of the probability of correlated failures).