

Availability - The 9's Game

**Bill Highleyman
The Sombers Group, Inc.
billh@sombers.com**

Caveat

This is a 50,000 foot view of availability.

Real world situations require getting much lower.

But for today, we want to see the forest, not the trees.

$$A = 1 - f(1 - a)^{s+1}$$

**If you're algebraically challenged,
ignore the math and listen for the concepts.**

Question:

If you add processors to your system, does it become more reliable?

Availability

Availability is the proportion of time that the system is up:

$$a = \frac{\text{MTBF}}{\text{MTBF} + \text{MTR}}$$

a is availability

MTBF is mean time between failure (average time system is up)

MTR is mean time to repair (average time system is down)

Example: System fails on average every 4000 hours
Repair takes an average of 2 hours

$$a = 4000/4002 = .9995 = 99.95\%$$

Reliability

a is the probability that the system will be up

(1-a) is the probability that the system will be down

Judge **reliability** by the probability of failure:

| | System A | System B |
|-----|----------|----------|
| a | .9995 | .9999 |
| 1-a | .0005 | .0001 |

System A will be down 5 times as much as System B

The 9's Game -

| <u>Nines</u> | <u>% Available</u> | <u>Hours/Year</u> | <u>Minutes/Month</u> |
|--------------|--------------------|-------------------|----------------------|
| 2 | 99% | 87.60 | 438. |
| 3 | 99.9% | 8.76 | 44. |
| 4 | 99.99% | .88 | 4.4 |
| 5 | 99.999% | .09 | .44 |
| 6 | 99.9999% | .01 | .04 |

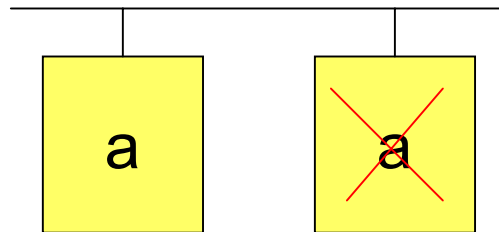
The 9's Game -

| | |
|------------|-------|
| Himalaya | .9999 |
| Mainframe | .999 |
| OpenVMS | .998 |
| AS400 | .998 |
| HPUX | .996 |
| Tru64 | .996 |
| Solaris | .995 |
| NT Cluster | .992 |

(Thanks to Gartner Group)

Providing a Backup

When one sub-system can be down:
(that is, both must fail for the system to fail)



$$A \sim 1 - (1 - a)^2$$

Example:

$$a = .99$$

$$A \sim 1 - .01 \times .01 = .9999$$

100 times as reliable as a single system!

Providing a Backup

Doubling the 9's

Adding a second redundant system doubles the nines:

| | | Availability | |
|-------------------------|---|--------------------------|---------------------------|
| | | <u>one system</u> | <u>two systems</u> |
| Typical system today | → | .9 | .99 |
| | | .99 | .9999 |
| | | .995 | .999975 |
| | | .999 | .999999 |

The Real NSK World

- multiple processors
- software distributed randomly across all processors (usually based on load balancing)
- each process has one backup (checkpointed or Pathway persistent process)

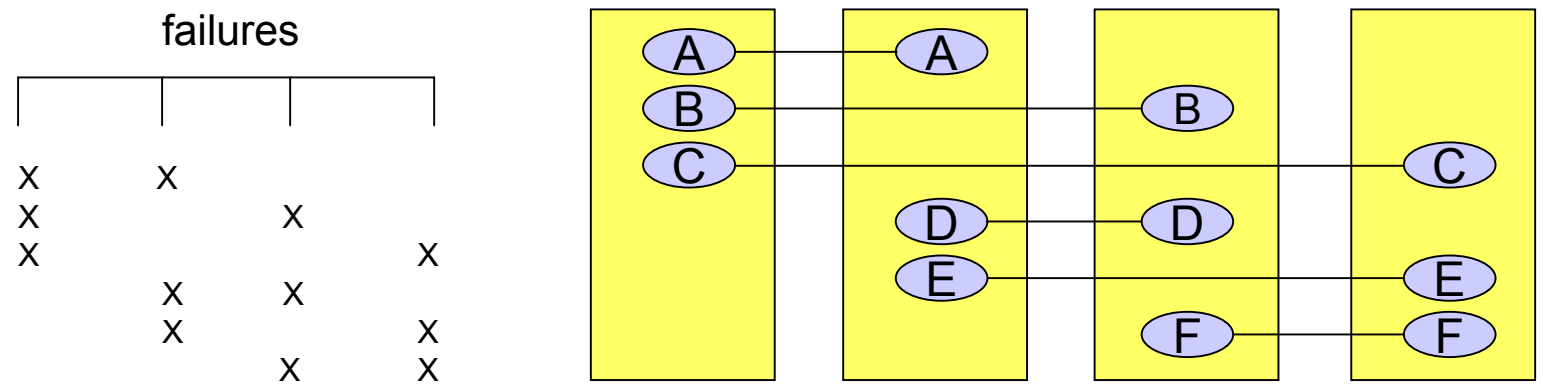
$$A \sim 1-f(1-a)^2$$

f = number of 2-processor failure modes

What are failure modes? Some examples →

The Real NSK World

Randomly distributed process pairs:



Any pair of processor failures causes a system failure

$$A \sim 1 - 6(1-a)^2$$

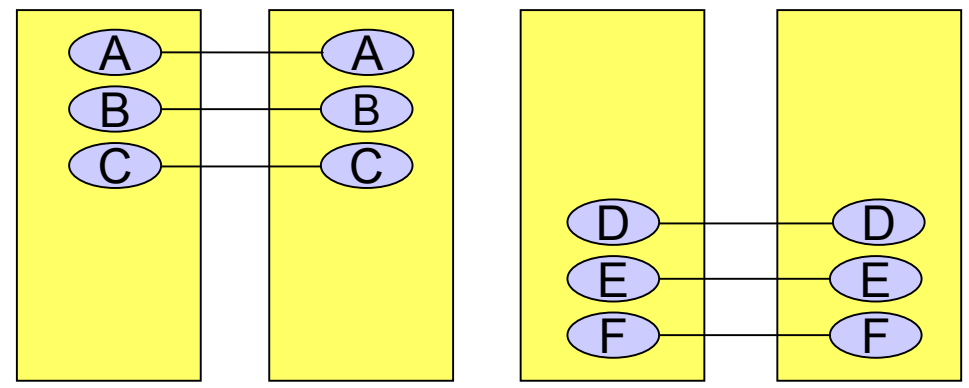
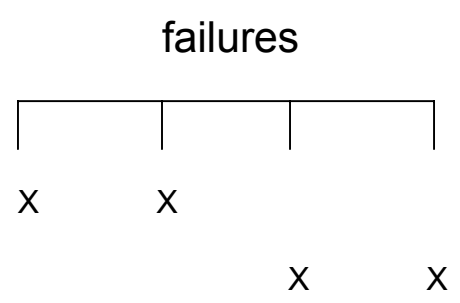
proc (n) # fail modes (f)

| | |
|---|----|
| 2 | 1 |
| 4 | 6 |
| 6 | 15 |
| 8 | 28 |

$$f = n(n-1)/2$$

The Real NSK World

Processors organized into pairs:



Only certain pairs of processor failures cause a system failure

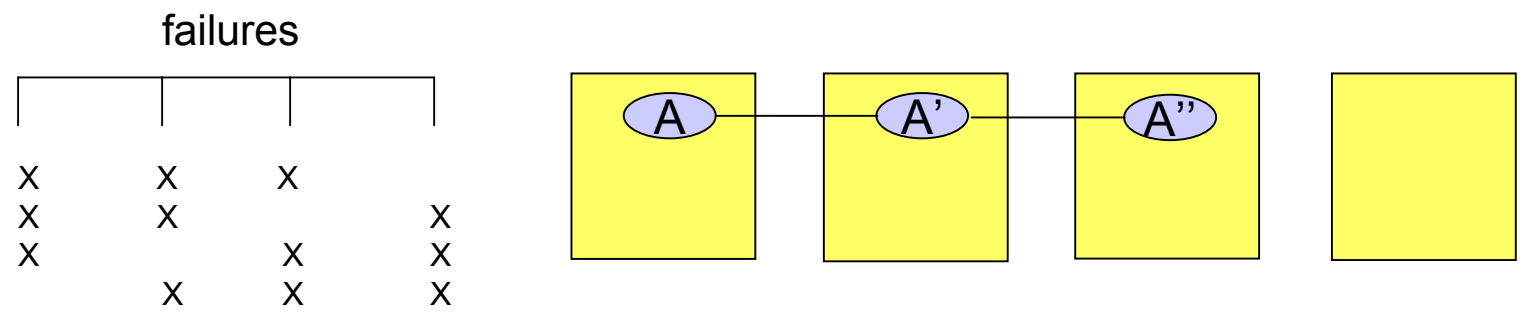
$$A \sim 1 - 2(1-a)^2$$

| <u># proc (n)</u> | <u># fail modes (f)</u> |
|-------------------|-------------------------|
| 2 | 1 |
| 4 | 2 |
| 6 | 3 |
| 8 | 4 |

$$f = n/2$$

The Real NSK World

Triple redundancy:



Any three processors must fail to cause a system failure

$$A \sim 1 - 4(1-a)^3$$

| <u># proc (n)</u> | <u># fail modes (f)</u> |
|-------------------|-------------------------|
| 4 | 4 |
| 6 | 20 |
| 8 | 56 |

$$f = n(n-1)(n-2)/6$$

The Real NSK World

Let's compare these approaches for 8 processors each with an availability of .995:

| | <u>A</u> | <u>f</u> | <u>downtime/year</u> |
|-------------------------------------|----------|----------|----------------------|
| random distribution | .9993 | 28 | 6.13 hours |
| paired distribution | .9999 | 4 | .88 hours |
| triple redundancy (random dist.) | .999993 | 56 | .06 hours |

Availability

The general availability equation for NSK systems

We usually design NSK systems with one spare.

For s spares:

$$A \sim 1 - (1 - a)^{s+1}$$

f is the number of $s+1$ processor failures that will take the system down

a is the availability of a processor subsystem

s is the number of spares such that one more failure *may* take the system down

Failure Modes (f)

For math nuts:

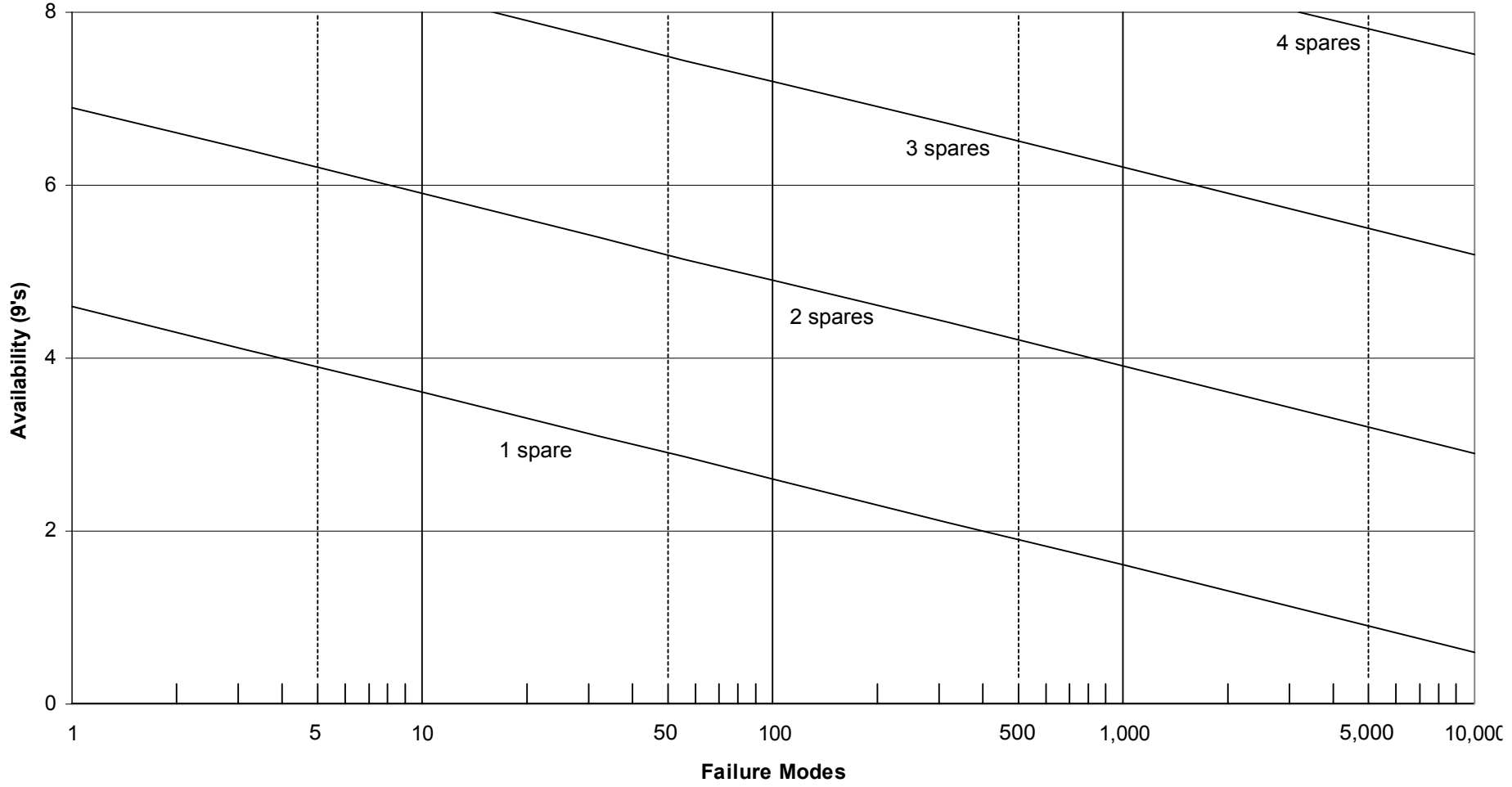
$$f = \binom{n}{s+1} = \frac{n!}{(n-s-1)!(s+1)!}$$

Maximum Failure Modes (f)

| | Processors (n) | | | | | | | |
|------------|----------------|---|----|----|-----|-----|------|-------|
| | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 |
| Spares (s) | | | | | | | | |
| 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 |
| 1 | 1 | 6 | 15 | 28 | 45 | 66 | 91 | 120 |
| 2 | | 4 | 20 | 56 | 120 | 220 | 364 | 560 |
| 3 | | 1 | 15 | 70 | 210 | 495 | 1001 | 1820 |
| 4 | | | 6 | 56 | 252 | 792 | 2002 | 4368 |
| 5 | | | 1 | 28 | 210 | 924 | 3003 | 8008 |
| 6 | | | | 8 | 120 | 792 | 3432 | 11440 |
| 7 | | | | 1 | 45 | 495 | 3003 | 12870 |
| 8 | | | | | 10 | 220 | 2002 | 11440 |
| 9 | | | | | 1 | 66 | 1001 | 8008 |
| 10 | | | | | | 12 | 364 | 4368 |
| 11 | | | | | | 1 | 91 | 1820 |
| 12 | | | | | | | 14 | 560 |
| 13 | | | | | | | 1 | 120 |
| 14 | | | | | | | | 16 |
| 15 | | | | | | | | 1 |

System Availability

versus failure modes (a = .995)



Test Question 1

**If you add processors to your system,
does it become more reliable?**

Test Question 1

If you add processors to your system, does it become more reliable?

Answer

It all depends:

- if spares remain the same, then it is less reliable.
- if more spares are added, then it is more reliable.

(Since we generally don't change the design, the typical answer is **NO**. Adding processors reduces reliability!)

Test Question 2

A 16 processor system with 1 spare and randomly distributed processes has the reliability of:

- the space shuttle**
- a UNIX box**
- a '37 automobile**

Test Question 2

A 16 processor system with 1 spare and randomly distributed processes has the reliability of:

- the space shuttle**
- a UNIX box**
- a '37 automobile**

Answer

A UNIX box.

Test Question 3

True or False:

A NonStop processor board is less than half as reliable as a UNIX processor board.

Test Question 3

True or False:

A NonStop processor board is less than half as reliable as a UNIX processor board.

Answer

True, but so what?

Given the same quality of components and similar component counts, a NonStop processor will fail if either lock-step processor fails. Therefore, it has two failure modes and will fail twice as often.

But this is what it takes to be fault tolerant and get four 9s.

Test Question 4

Should you upgrade your 4 processor S74K to an 8 processor S74K or to a 4 processor S86K?

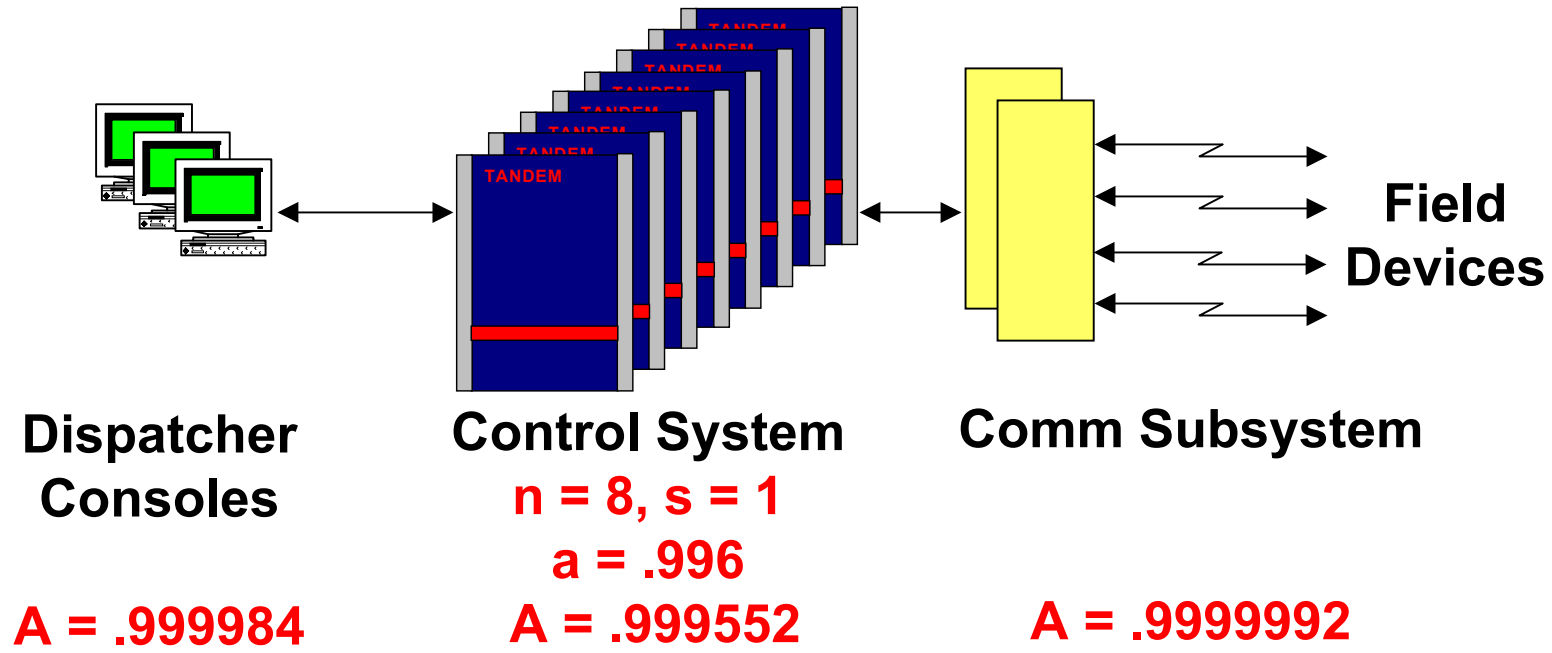
Test Question 4

Should you upgrade your 4 processor S74K to an 8 processor S74K or to a 4 processor S86K?

Answer

A 4 processor S86K will be almost five times more reliable. As an added plus, its response time will be almost twice as fast.

Case Study - Amtrak



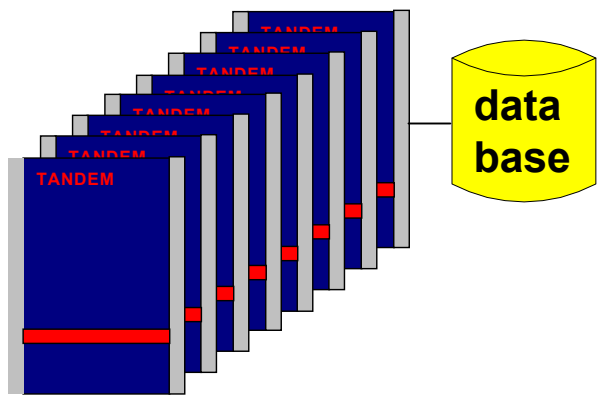
Required Availability = .9998

System Availability = .999984 x .999552 x .9999992 = .99950
Failure Rate is .0005/.0002 = 2.5 times worse than required.
Solution: Reduce failure modes from 28 to 12 through software configuration.

Case Study – Split System

Full System

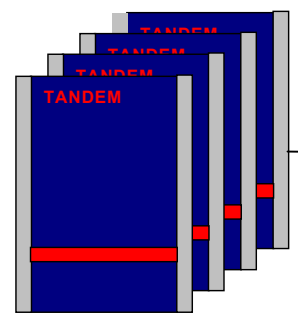
$n = 8$
 $s = 1$



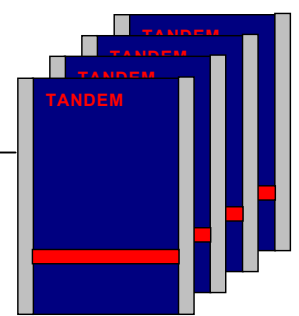
$$A_{100} = .9993 \text{ (3 9s)}$$

Split System

$n = 4$
 $s = 1$



$n = 4$
 $s = 1$



$$A_{100} = .9997 \text{ (3+ 9s)}$$

$$A_{50} = .99999998 \text{ (8 9s)}$$

Reminder – Full redundancy doubles the 9s

Is 4 9s Enough?

The Standish Group Says:

| | | |
|--------------------------|-----------------------|-------------|
| Non-Critical | 90 hours/year | 2 9s |
| Task Critical | 9 hours/year | 3 9s |
| Business Critical | 1 hour/year | 4 9s |
| Mission Critical | 5 minutes/year | 5 9s |
| Safety Critical | 1 minute/year | 6 9s |

Breaking the 4 9s Barrier

$$A \sim 1 - f(1 - a)^{s+1} \sim 1 - f \left(\frac{mtr}{mtbf} \right)^{s+1}$$

- f** Easiest. Minimize via processor pairing.
(maintain 4 9s)
- s** Done for hardware. Do it for software
(at least triple redundancy (s=2)).
(pick up 2 9s – but this requires HP action)
- mtbf** This is given to us.
- mtr** A subject in itself ...

Breaking the 4 9s Barrier

$$A \sim 1 - f(1 - a)^{s+1} \sim 1 - f \left(\frac{\text{mtr}}{\text{mtbf}} \right)^{s+1}$$

mtr is the time to recover the system.

Understanding how to reduce mtr depends on understanding what causes outages:

1% caused by dual hardware failures.

99% caused by software bugs or human errors:

Generally recover with a cold load.

Typically 4 hours, mostly application time.

Design applications for recovery.

Cut mtr (recovery) by three, add a 9!

So ...

where do you want to be?

Higher availability doesn't come for free.

But it doesn't cost a whole lot either:

- minimize failure modes**
- design for fast recovery**
- pick up a couple of 9s!**