This article is part of the IEEE Reliability Society 2008 Annual Technology Report.

Today's Perspective of Network Reliability

Mod Marathe (mmarathe)
Email: mmarathe@cisco.com

Network Availability Basics

In this note, we assume that there will be failures in a network and we are mainly concerned with how network components like routers, switches and trunks react to these failures to minimize their impact on end customers. So we should start with what we consider to be a failure and how often failures happen in network components.

What is a failure?

The formal definitions of fault and failure are as follows.

Failure – an event when a component of the system deviates from the required functionality
Fault - an incorrect state of the hardware or software that may cause a component or the system to fail
Failure Detection – a process performed in a certain interval of time to discover the failure

In essence, a fault is an incorrect state that causes a failure to happen. Since both these terms are commonly used interchangeably, we will not make any distinction between them in the rest of the note.

Among all the hardware and software failures that can happen in a network, this note is only concerned about service affecting failures. So we will not deal with bugs in the features and functionality or in network management as long as they don't affect service. Moreover, we will focus on service disruptions that exceed a threshold. Clearly, different network applications have different thresholds for determining that service has been affected. To put it in perspective, in general for IP (Internet Protocol) data traffic we are only concerned about failures that affect service for several minutes or longer. Small transient failures, losses of a few packets or momentary outages that are automatically resolved either through higher level protocols (retransmissions) or switchovers to redundant hardware or software are not considered here.

For voice applications, an interesting threshold that needs to be discussed is the length of an interruption in a voice call. From common sense, it would appear that if a voice conversation is interrupted for 3 to 5 [1] seconds, the participants will notice it but may not hang up. So this would be the requirement for restoring voice connections on any switchover. If the end users were using this connection for a modem call or a fax call, the requirement would have to be much smaller – 750 ms or so because the modems will drop the call once they lose the carrier for this long. The Bellcore standards do not specify any such threshold other than the Sonet APS switchover outage requirement of 60 milliseconds.

One distinction that may be important is "planned" versus "unplanned" service outage – planned outage is generally for software or hardware upgrades or other maintenance activities. Since planned outages can be scheduled in advance the impact on the end customers can be reduced. Unfortunately, the reality of most large networks is such that there is not really a "good" time to schedule an outage. Most of these networks are running on a 7X24 schedule and any outage affects some customers. Unplanned outages are all the other outages that were not anticipated and happen due to some failure in the network.

Mean time between failures (MTBF)

MTBF (mean-time-between-failures) is the average expected time between failures of a product, assuming the product goes through repeated periods of failure and repair. MTBF applies when a product is in its steady-state random-failure life stage (that is, after the infant mortality and before the wear-out periods), and is equal to the reciprocal of the corresponding constant failure rate. A useful interpretation of MTBF is that within the period of MTBF, 63% of the product population is expected to have failed at least once [2].

Network components consist of hardware and software so we need to consider their failure rates separately. Further, hardware consists of many field-replaceable-units (FRUs) and since they can have different failure rates, it makes sense to consider their individual MTBF. Similarly for software, different versions or types of software could be running on different cards so we need to consider the MTBF of each of these different software images.

MTBF is generally specified in the units of hours. One year has 24*365 = 8760 hours. In general, hardware MTBFs are in the range of 100,000 hours or more and software MTBFs are in the range of 10,000 (new versions) to 100,000 (mature versions) hours.

Hardware MTBF

How do we estimate or measure all these different MTBF values? For hardware, Quality Engineers calculates a predicted MTBF number that is based on the Telcordia (formerly Bellcore) "Parts Count Method [TR-332]." Individual device failure rates are obtained from the suppliers or taken from the latest issue of the Telcordia specification TR-332 tables. The individual failure rates of all the components on the board as well as those for connectors on the board are added together to get the overall failure rate of the board. It is difficult to estimate the failure rate of a new ASIC so if the board contains many ASICs, the MTBF estimate may be inaccurate. In any case, it turns out that the failure rates in TR-332 are pessimistic and actual experience in the field is usually better than these predictions. Also, an important contributor to component failure rates is the temperature the component will be running at. As we build products that push the state of the art and capacity, we tend to run these products at high temperatures (60°C or more)[3]. This needs to be accounted for when running the TR-332 calculations. So the bottom line is that the predicted MTBF values for boards are not necessarily accurate and should be validated against actual field experience.

FIT or failure in time

FIT (failures in time) is a measure of failure rate and is defined as the number of failures per 10^9 operating unit-hours. FIT rate is typically used to describe the reliability of components and circuit cards. Failure rate and MTBF are reciprocally related as in the following equation.

$$\text{MTBF} = 1 \div \text{Failure rate} = 10^9 \div \text{FIT} \qquad \text{in hours}$$

Software MTBF

The MTBF values for software are very difficult to predict – if there were a TR-332 like technique for software, it would have to depend on the lines of code, the number of "if" statements, the number of files, the number of developers that have touched the code, the test coverage and such factors. Some people look at the rate of finding new bugs just before the software is released and extrapolate that to predict the MTBF. A good reference for these techniques is "Software Reliability Engineering" by John Musa (McGraw-Hill 1999). These techniques are still experimental so the only way of determining software MTBF is to measure it in the field.

Since software failures are usually triggered by some unusual combination of events and usually depend on the particular hardware it is running on, even if we were to measure the software MTBF in the field for one network, we will have difficulty applying it to another network. Also, the field measurements can only be made on shipping versions of the software and we are generally interested in predicting the MTBF of new versions of software. Newer versions of software will generally have more failures in the beginning.

All these caveats are necessary to keep in mind when using software MTBF values. Nevertheless, it is still useful to measure software MTBF values so we can compare one release to another or evaluate the effects of various quality improvement initiatives in software development. Also, even an approximate value for the software MTBF is useful when comparing the availability improvements possible if different availability features were to be implemented.

Software upgrade MTBF

It turns out that for many network components, the rate of software upgrades is higher than the rate of crashes. For example, if the software is upgraded three times a year and if a software upgrade requires an outage, its MTBF will be at most 8760/3 = 2920 hours. A lot of software is upgraded at this rate due to new features or bug fixes. If the MTBF of crashes is say 20,000 hours, we gain more by reducing the outages caused by upgrades than those caused by crashes.

Mean time to repair (MTTR)

MTTR is the average expected time to restore a product to service after a failure. It represents the period that the product is out of service because of a failure, and is measured from the time that the failure occurs until the time the product is restored to full operation. Therefore, MTTR includes the times for failure detection, craft dispatch (if at an unmanned site), fault diagnosis, fault isolation, the actual repair, and any software and protocol resynchronization time needed to restore the entire service.

Generally, recovery from software crashes is accomplished by switching to a redundant processor or if that is not available, just restarting the crashed software. Recovery from hardware failures requires switching to redundant hardware (if available) and/or replacing the failed hardware.

In general, we want to automate recovery from failures because the service outage time is much longer if manual operations are needed. Also, manual recovery can sometimes lead to other accidental failures while repairs are being made. Typical numbers for MTTR are 0 to 30 minutes for automated recovery and 1 to 3 hours for manual recovery. One reason why the manual recovery takes so much longer is that in a large network the network operations center needs to remotely diagnose the problem and instruct the local operators to execute the correct recovery procedure. Also, in a large network, protocol resynchronization can take a long time as the knowledge that the network has been repaired gets propagated throughout the network. Obviously, the better job we can do of isolating and identifying a failed field-replaceable-unit, the shorter the MTTR will be.

Availability

Availability is traditionally defined as the probability that a product will operate when needed. For mature communications equipment at steady state, it is the average fraction of time that the product is expected to be in operating condition. For a simplex system (that is, without redundancy), availability is defined as follows:

$$Availability = \frac{MTBF}{MTBF + MTTR} \quad for\ a\ simplex\ system \qquad \textbf{FORMULA 1}$$

Availability is normally expressed as a percentage and in fact, sometime expressed as some number of '9's. For example, availability of 99.999% is called "five nines". Since there are roughly half a million [4] minutes in a year, 99.999% availability translates into an unavailability of 10 per million or 5 minutes per half-million minutes or 5 minutes per year. An availability of 99.99% will be equivalent to a downtime of about 50 minutes per year and an availability of 99.9999% will be equivalent to 30 seconds per year.

Don't sweat the milliseconds:

Note that even if availability is expressed in minutes of downtime per year, it is important to remember that this is an average over many deployed units. The annual downtime is not accumulated on each unit by adding up a few seconds here and there because such small outages do not exceed the measurement threshold. Rather, most of the units work without failure but some of them fail with a very long repair time extending into tens of minutes or hours.

Such a definition [5] for availability is good for a system consisting of a single box such as a workstation. However, in a network that consists of a number of routers and trunks, most failures are partial failures – that is, only some customers do not receive service but others are not affected. Moreover, even within a router only one line card may be down – so if you only consider the entire router, it is neither "up" nor "down". How do we then use the availability definition given above?

Availability of a network is therefore defined with respect to a customer of the network – it is the probability that a customer of the network will find the network to be in an operating condition.

To compute availability, we only need to consider the components along the data path needed to provide service to a single customer. If we choose a typical customer, the availability experienced by that customer will be the typical availability experienced by all the customers.

Unavailability

Unavailability is simply defined as

$$\begin{aligned} Unavailability &= 1 - Availability \\ &= 1 - \frac{MTTF}{MTTR + MTTF} \\ &= \frac{MTTR}{MTTR + MTTF} \\ &= \frac{MTTR}{MTBF} \end{aligned}$$

$$Unavailability = \frac{MTTR}{MTBF} \qquad \textbf{FORMULA 2}$$

Defects per Million (DPM)

DPM (*defects per million)* experience of a typical customer of the network is the preferred metric for availability. It is adaptable to the particular application being run on a network. There are different formulas for the typical applications that run on a network (permanent virtual circuits, switched virtual circuits, voice calls, connectionless data service). In each case, the DPM values are independent of the size of the network and of the number of users on each subscriber side trunk or card. Another property of these definitions is that they can be used to calculate the availability of an existing network as well as calculate the theoretical availability prediction of any network. Finally, calculations are simpler because the DPM values are additive along the data path.

References

[1]The reason for saying 3 to 5 seconds instead of specifying a single number is that the users' expectations may be different for different types of voice calls. For a landline phone, we may have to use 3 seconds but for wireless, which the users don't expect to have high reliability/quality, we may be able to stretch it to 5 seconds.

[2]This is based on an assumption that the failures are exponentially distributed so the distribution function is F(t) = (1 – e-t/a) where a is the MTBF. This gives F(a) = (1 – e-1) = 0.63

[3]Many products' MTBF calculation is based on operating at 40°C.

[4]365 days × 24 hours × 60 minutes = 525600 minutes per year

[5]There is a slight inaccuracy in this definition. Availability is actually defined as the percentage of time that a device is operational relative to the total time it is in service. During the repair, the device is not operational; so the availability is MTTF/(MTTF+MTTR) where MTTF is the mean time to failure once the device starts working. If MTTR <<< MTTF (where MTBF=MTTF+MTTR), then MTTF/(MTTF+MTTR) is approximately equal to MTBF/(MTBF+MTTR); for these systems, MTTF and MTBF could be used interchangeably in the formula.