QOS evaluation model for a Campus-Wide Network: an alternative approach

Martí GRIERA (Martí.Griera@uab.es)
Maribel JIMÉNEZ (Maribel.Jimenez@uab.es)
Juan Antonio MARTÍNEZ (Juanan.Martinez@uab.es)

Abstract

This paper discusses a model for evaluating the Quality of Service of a Campus-Wide network. The main goal is to achieve a mathematical model that can be applied to any network, providing its availability. The network where it will be tested has more than 5000 nodes with SNMP managed network devices.

The model discards the classical approach to QOS that considers parameters such as delay or jitter. Instead, we evaluate quality as a function of the availability indexes for either the services or the network itself.

Once the model is described, we will explain how it has been implemented at the Universitat Autònoma de Barcelona Campus network and the requirements – software and hardware – to install such a system in any network.

Keywords

Network monitoring, service monitoring, quality of service (QOS), availability evaluation, management.

1. Introduction

‘The network is always down’ is perhaps one of the sentences network administrators hear most frequently. With this idea in mind, we tried to develop a system that showed this perception is not always true.

This paper describes the process for obtaining a mathematical model to evaluate the quality of service of an arbitrary given network. The main goal is to determine whether a network accomplishes the ‘5 9s’ ideal availability. Once this has been done, we will introduce a method for determining the quality of high level services.

2. Parameters to evaluate the QOS of a generic network

Almost all the literature concerning quality of service focuses on parameters such as delay, packet loss or jitter ([2],[3],[5]). In our opinion, these parameters are crucial to determine the quality of a WAN link, but we think they are not so important for local networks (LAN). Experimental approaches show that under normal circumstances, delays are negligible and packet loss minimal on local environments.
Working with the above parameters involves additional problems: they require a
distributed measurement environment ([13]). This does not happen on WAN links,
where acquiring these values is quite simple. It normally requires no more than
consulting the router status at the end of the link. For LAN environments, a solution of
this kind is complex and not always possible. A switched network, for example, has
different values for jitter, loss and delay for each segment.

The easiest approach to solving this problem is to use SNMP ([10],[11]). Although
SNMP is especially useful in determining reachability, it is not suitable to determine
delays or jitter when accessing a device. Our experience also shows that when pinging
an SNMP device, the response time is more conditioned by the device load than by the
network traffic. This is easy demonstrable simply by pinging an idle station (a PC for
instance) and its SNMP device ([11]). We can achieve different results, and these results
are almost always higher for the SNMP device.

With this idea in mind, we found it unacceptable to construct a model based on an
inaccurate measurement parameter. The cost of a simple system based on network
probes is not affordable. The basic structure of the network is shown on figure 1.
Essentially it is a switched network with more than 70 segments. So we looked for a
model that would be easier and, in addition, more accurate to implement.

![Fig.1. Schematic view of the network](image)

Thinking about the network as a whole, we see that quality will be based on different
indicators for the local environment and the WAN links. With our previous reasoning,
packet loss and jitter are correct and accurate for WAN links. For the LAN, we think it
is better to have a numeric value that shows the availability for the whole network. In addition, the services classically associated with this (such as mail, ftp, www,...) should also be analyzed, in order to get a complete view of the network and its behaviour. If we can get numerical indicators for each mentioned item, we will not only be able to determine the quality, but also to evaluate how migrations will affect the network. We will begin by analysing the LAN environment, and after this will look at high level applications ([4]).

3. Practical model to evaluate the availability of an arbitrary network

3.1. Requirements for the model

The model we are looking for must:

- Be suitable for any network, provided that their network devices are SNMP-capable
- Give a numerical value that corresponds to the availability at any moment. Furthermore, it must generate periodical reports of the mean value of the availability.

The basic idea behind the model is to evaluate the influence of a network failure over the global network. To achieve this, we determine how many users are affected. Once this number and the total users of the network are known we can determine the relative importance of the failure.

Although this was the original approach, we think that this model is too linear and does not adjust to the real environment. Given a network failure, the relative importance is not only provided by the number of users, but also by the criticality of the hosts connected to it. For example, although two routers can affect the same number of users, their impact is different depending on their function. So, we must introduce an additional value that considers this functional approach.

Practical considerations show that it is absolutely impossible to monitor all network devices. Monitoring all the devices gives a more accurate value, but requires a dedicated machine and the measurement itself may affect the results. Monitoring a small number of devices gives an incorrect value for availability. Therefore, a compromise must be made.

3.2. Practical implementation

In our scenario, the network is structured in ethernet segments with a set of central switches and other second-level switches. As a whole, the network can be seen as a switched structure that ends in shared 10/100 segments.

Although the traffic load can be different in each segment, in terms of availability the failure of a switch affects all the segments it contains. So, in our analysis we consider only the first and second level switches. Arriving at the end devices would give better granularity, but the monitoring load then increases from tenths of devices to more than 150.
For the mathematical model, let $K_{c1}$ be the total number of users affected by the failure of a given switch. If we divide the number by the total, we get the relative importance $k_{c1}$. Furthermore, from our experience, we give each of the devices a subjective importance $K_{c2}$ in the range $[0,10]$ – where 0 indicates minimum criticality and 10 maximum –. We also obtain the relative coefficient for the subjective constant. Table 1 shows our statement applied to the network:

<table>
<thead>
<tr>
<th>Segment description</th>
<th>Network device</th>
<th>IP Address</th>
<th>User coefficient ($K_{c1}$)</th>
<th>Relative user coeff. ($k_{c1}$)</th>
<th>Critical value (up to 10)</th>
<th>Relative critical value</th>
<th>Global coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corporate Router</td>
<td>Gw</td>
<td>158.109.0.3</td>
<td>2920</td>
<td>0.3333</td>
<td>7</td>
<td>0.0729</td>
<td>0.20312</td>
</tr>
<tr>
<td>Central Backbone</td>
<td>CB</td>
<td>158.109.0.26</td>
<td>2920</td>
<td>0.3333</td>
<td>9</td>
<td>0.0937</td>
<td>0.21354</td>
</tr>
<tr>
<td>Comp.Sciene Switch</td>
<td>si0swfo1+si0swfo2</td>
<td>158.109.2.233</td>
<td>187</td>
<td>0.0213</td>
<td>8</td>
<td>0.0833</td>
<td>0.05234</td>
</tr>
<tr>
<td>Science School</td>
<td>c7p1sw1</td>
<td>158.109.8.236</td>
<td>357</td>
<td>0.0407</td>
<td>6</td>
<td>0.0625</td>
<td>0.05162</td>
</tr>
</tbody>
</table>

Table 1. Detail of the table to implement the model

From coefficients $k_{c1}$ and $k_{c2}$ we obtain a global coefficient that we will introduce into the model. This coefficient is obtained by computing the arithmetic mean for both coefficients. If we call this new coefficient $k_d$, the availability can be computed as:

$$\text{Instant availability} = \sum_{i=1}^{n} K_{d(i)}$$

where $K_{d(i)}$ is each of the availability coefficients and $d(i)$ a binary value that shows if a network device is working properly at a given moment. In order to get the mean value we studied some methods such as mean filters, but finally we concluded that the arithmetic mean was suitable. So, for the instant availability,

$$\text{Total availability} = \frac{1}{n} \sum \text{Inst. availability}$$

In order to get the real mean value over time, we can say that:

$$\text{Mean availability} = \int_{0}^{t} d(t) dt$$

Computing $d(t)$ would require interpolation methods to evaluate availability as a function of time. This would be useful if we wanted to make predictors, but this is not the goal. If we assume that the intervals of measurement are constant, integral becomes a sum and no interpolation is needed, as we work in a discrete space:

$$\text{Avail}[n+1] = \frac{n \ast \text{avail}[n] + \text{avail}[n+1]}{n+1}$$
which gives us the availability over time. We only need to have the previous values of
the availability and the number of samples. The reader may infer that this formula is
none other than the rectangles method to compute the integral above. Instead of making
a whole set of rectangles, we assume we have computed the mean value up to the
previous sample and then we compute the area of this ‘large’ rectangle. Figure 2 shows
these simplifications:

![Figure 2. Simplification of the evaluation of the mean value](image)

Therefore, network availability can be computed only with one sample, the previous
mean and the number of samples carried out up to the moment the measurement is
taken.

### 3.3. Considerations regarding practical implementation on the availability model

In order to implement the previous model, we have written a C program that gets the
data from the network devices we want to monitor. A configuration file tells the
application which devices to monitor and which coefficients to enter into the model.

Regarding these coefficients, a previous analysis has been done in order to choose them
correctly. We first studied how the instantaneous availability changed with the chosen
coefficients. Our first approach was to consider only the 'objective' part of the
coefficient (the number of users a device connects). But our analysis showed that
approach is not realistic. So, subjective coefficient is needed. This part of the coefficient
gives a higher sensitivity (though the model remains stable) but also makes a more
accurate model.

Although we studied various methods in order to compute the global coefficient, the
arithmetic mean value was suitable for the model. Other methods can be chosen, and
they are correct both for the 'subjective' coefficient and for the evaluation of the mean
value. The most important point is to tune the coefficients before implementing the
model. And this work must be done by each network administrator. The program
provides instant availability for a given network with the coefficients introduced. Once
they are chosen, they should not be changed, as the historical data would make no
sense. In our opinion, the most important factor is not the number, but the historical analysis of the gathered data.

To evaluate instant availability, we consider a device to be active when it responds to an ICMP request. As the system is modular enough, other criteria can be applied, but this is simple and universally available. We use the classical ‘ping’ to get a binary value for a given device. To avoid possible problems with management, we make a set of ‘pings’ and consider it available when we get more than 75% of the responses. With this data the availability value can be easily computed.

This instant value is stored in a text file, where the last line indicates the number of samples taken up to the moment and the mean availability. With these two values and the new sample, the new mean value is simple to compute with the formula above. In order to clarify this, we show the last lines of one of our availability files:

```
0.984809713600  7641    Thu Dec 28 11:25:00 2000
0.984811701337  7642    Thu Dec 28 11:30:00 2000
0.984813685544  7643    Thu Dec 28 11:35:00 2000
0.984815675251  7644    Thu Dec 28 11:40:00 2000
0.984817614283  7645    Thu Dec 28 11:45:00 2000
0.984819647086  7646    Thu Dec 28 11:50:00 2000
0.984821632224  7647    Thu Dec 28 11:55:00 2000
0.984823616843  7648    Thu Dec 28 12:00:01 2000
0.984825600943  7649    Thu Dec 28 12:05:01 2000
```

We can see that computing the new value is straightforward. We get the new instantaneous value as explained. The new mean value is computed with this value and the values we get from the last line of the availability file. The new computed value and the increased number of samples is saved in the file, together with the time the sample is taken.

Network administrators should be careful when implementing the model in networks where VLANs exist. The station that evaluates the availability should be able to get data from all the network devices. This is usually accomplished just by installing both the management station and the managed devices in a common VLAN. In fact, this is not a restriction, as our experience shows that there is normally a VLAN which includes all the managed network devices and the management station(s).

Any other technologies and environments (level 4 switching, firewalls...) are supported as long as there is IP connectivity between the management station and the managed devices and no filtering of ICMP echo request and replies is carried out. Proprietary methods could be implemented to substitute the ICMP test, if this is filtered. Our software could accept this with no major changes, but this feature has not been developed, as we have no need to do so.

4. Evaluating quality of the services: the extension of the model

The first idea was to implement a system that simply reported the state of the services that were working on the network. So, the same configuration file used for the network
could be used to evaluate the availability of a given server or service. All we had to do was to add a new section with this elements.

As the model is greatly extensible, if we get rid of the criticality coefficient, we obtain a valid model for evaluating the availability of a given service. The only difference resides in the evaluation of the instant availability. Instead of ICMP queries, we should do it with specific programs to check this services.

Our implementation provides these tools to determine whether network services are working or not. This tools substitute the ping test for specific tests on each service. We have implemented check-service routines for each service. Among other information, this routines return whether the service is working or not.

But we wanted a more powerful and flexible tool. The idea was to know whereas –for example – a mail server is consuming too much CPU time, memory,… The possibilities we had for a centralized monitoring system were:

- Perform remote monitoring from a central station. This central station would call remote shell (rsh) utilities to determine whether processes are working or not.
- Perform local monitoring in each machine and export the results to a central station in order to be processed.

The first idea has potential security holes. Many UNIX administrators are reticent to let a remote machine execute rsh processes, even more if some of this processes should run under a privileged account. The second approach is fair, but we should think of a system to export data.

We consider NFS to be a first approach as we work in a UNIX environment, but when thinking in a network monitoring system the standard is SNMP. Using SNMP would be not only desirable, but also it would integrate with tools to make graphical reports of snmp-gathered data (like for example MRTG).

4.1. Details of implementation

Up to the moment, our requirements are:

- A set of routines to get the data we want to monitor.
- An SNMP daemon capable to communicate itself with the previous routines and to export the values when an snmp-get call is received.
- A MIB to access this information ([6],[9],[11]).

The first point was not a problem for us, as we had encoded a set of monitoring scripts which obtained the important data from the services we were interested in. The problem was to export this data via SNMP.

The required SNMP daemon (classically snmpd) doesn’t work in this scenario, as it would not recognize the values we would ask it. The daemon we work with is a modified version of the University of California-Davis ([12]). We generated a mib definition file and we integrated the font code in the daemon([6],[9]). The generated code is absolutely portable.
Regarding operative issues, the local processes write the monitoring results to a file (we call it snmp-data). When the daemon receives a query (snmp-get), it reads the value from this file. In our implementation all variables are read-only as we don’t expect to interact with a service based on snmp variables.

5. Overview of the system

As a whole, the system consists of:

- A set of scripts – written in C – which verify that network services are working properly. This network services are bootp, dhcp, dns, ftp, http, mail (pop, imap, smtp), radius and news. These scripts run locally on each machine and are completely portable (they are running on Solaris, Linux and IRIX). Portability to non-UNIX environments has not been done but it should be done without major changes. The only requirement is a C/C++ compiler and snmp tools (mainly snmp-get) for that platform.
- A configuration file that tells about the services that should be monitored and the machine where they run. This file also contains the values to compute the availability of the network. This file resides on the central monitoring station.
- Mathematical routines to evaluate the availability with the previous coefficients.
- A modified version of the snmp daemon. This daemon and a corporative mib (under the mib2 tree) allows local monitoring and remote consulting from any management station. It also provides integration with graphical tools to view this data.

Important benefits we get with the system are:

- A quite simple system to verify the proper function of all network services. The system is modular. So, it can grow with new services without any problems.
- Monthly reports of the availability of both the local connectivity and the external connectivity.
- Monthly reports of the services’ availability.
- Graphical view of the evolution of all the systems being monitored.

5.1. Comparison to other existing products

Before developing our own product we evaluated other existing software, both commercial and freeware, but we did not find any of them that solved our needs. They were either suitable either to monitor the status of the network or to graphic data.

In the first group of programs we find Big Brother ([1]) and NoCoL ([8]) for example. They give an idea of the status of the network on a given moment but they do not provide the historical reports we desired. In our opinion, it is really important to provide the monthly (or periodical) reports of the network and service availability. None of the software we saw gave this kind of reports. Some of them provided some data for the services but no model was found for the whole network. Big Brother could provide some kind of report for the services, but with data processing.

Regarding the graphical software, it generally has no model implemented but simply plots the data we order it to. This happens for example with MRTG ([7]). We think it is
not incompatible with our software and, in fact, we use it to graphic the data our model produces. In our opinion it is a really useful software to graphic any numerical data. But is should remain clear that MRTG generates no data, but simply plots it. MRTG is specially suited to plot SNMP gathered data and it can easily plot data from other programs.

As a whole, we can find software that either plots the data or shows the status of the network but we have found none of them that implements a model for the availability of the network or the services. We think ours does and, in addition, it is a model that can be highly configured to fit any network. Our experience shows that it is a really useful tool not only for the technical staff but also for generating reports to inform the managers.

6. Conclusions

The described model is a new approach to evaluating network quality through availability. It is simple, scalable and easy to develop. The numerical index for the network and the services, greatly helps network managers to decide which servers or services to improve and to determine potential problem sources.

In addition, the scripts are of great help to the network operation center (NOC) personnel, as a tool to determine if a service is not running properly. Although it has not been explained, a web interface to the system is available, which greatly simplifies the tasks of remote monitoring.

7. Acknowledgements

Thanks to Mr. José Manuel Castillo for his help on the codification of the network services monitoring scripts as a part of his thesis on Computer Engineering.

8. References


[5]: Matthews,W; Cottrell, Les: PingER: Internet Performance Monitoring. INET'99 Proceedings (Poster nr.26)


