Abstract. In the last couple of years the perfSONAR project has evolved into a robust scalable framework that offers a wide range of services for collecting network performance data which are being deployed all over the world. Data can easily be retrieved and examined using already available tools like perfsonarUI, CNM or VisualPerfSONAR. However, at this stage of development perfSONAR still has to deal with some unresolved issues. As middleware software perfSONAR is too visible to the end user and the relations between distributed services need to be officially defined. This paper proposes possible solutions to resolve these issues.

Keywords. MDM (Multi-domain monitoring), GEANT2 (the seventh generation of pan-European research and education network), NMWG schema (Global Grid Forum’s Network Measurement Working Group XML schema is used to represent measurement data), NREN (National Research and Education Network), perfSONAR (Performance focused Service Oriented network Monitoring Architecture), SOA (Service Oriented Architecture)

1. Introduction

Building a multi-domain monitoring (MDM) system [1] is probably one of the most important projects in the seventh generation of pan-European research and education network (GEANT2) community. The fact that an MDM system deals with network performance monitoring across multiple domains speaks volumes about its technical and political complexity. When we talk about MDM in GEANT2, we are in fact talking about perfSONAR (Performance focused Service Oriented network Monitoring Architecture) [3]. In its core sense, perfSONAR is a service-oriented framework that uses the perfSONAR XML protocol described by the NMWG schema [2] for client-service and service-service communication. Figure 1 represents a general overview of MDM and its relation to perfSONAR.

![Figure 1. Multi-domain monitoring service](image)

The perfSONAR framework was designed with various network monitoring tools in mind that are already available and deployed by the NRENs. This implies that the perfSONAR framework is not just another distributed network performance monitoring tool but also an interface that can be used as an abstraction layer for connecting various existing tools together. By wrapping these tools, it enables the user to retrieve data using a single, unified language.

The main end users of the perfSONAR services are network operation center (NOC) and the Performance Enhancement and Response Team (PERT) users. As their main objective is to concentrate on network performance and troubleshooting problems, they should not be concerned or even know about the perfSONAR framework.

Although hiding the perfSONAR middleware is mainly the responsibility of the visualisation tools, it is also a perfSONAR service issue. The
perfSONAR framework should be capable of correlating various data between different services. This can be achieved by using the NMWG schema to establish the data relations, which would assist visualisation tool developers in making perfSONAR transparent. This leaves the end user to deal only with the network devices and performance data without requiring knowledge of how data is gathered and shared.

2. Boosting perfSONAR functionality

Looking at the perfSONAR framework in general we can identify some of its main components. It comprises tools capable of gathering and storing performance data of network devices, it provides easy service discovery by enabling the registration of data with the Lookup Service (LS) [3], and it offers the Authentication and Authorisation Service [3] for dealing with security and NREN policy issues. Having identified these main components, we can create a simple scenario showing NOC or the PERT user actions, perfSONAR framework actions and visualisation tool actions.

2.1. Creating a scenario

In this scenario an end user wants to examine a traceroute path between some network points. The user enters the traceroute output from a traceroute tool or a visualisation tool into one of the available visualisation tools. The end user asks for bandwidth utilisation of the interfaces on the identified routers across the path, and also wants to execute some commands on these routers.

We shall concentrate on four perfSONAR services that are queried in the scenario. The first one is the Lookup Service where all services register information about the data they provide along with their locations or URLs. The second is the RRD MA [3] which handles bandwidth data. The last one is the Telnet/SSH Measurement Point (MP) [3] which allows users to execute commands on routers.

2.2. Identifying a problem

Most visualisation tools query the LS service for the locations of the perfSONAR services that provide bandwidth utilisation data for the network interfaces involved. They receive the locations because measurement services that store bandwidth information register interface IP addresses with the LS. Once the locations of these services are discovered, the visualisation tools query each service for the required data.

However, if the user wants to execute commands on the routers that have those interfaces, they are not able to do so, at least not without being aware of the Telnet/SSH MP, its location and the name of the router it can operate with. The user has no other option but to go through the list of available Telnet/SSH MPs and manually attempt to find the MP that has the device with the interface from the traceroute output. They are required to know the symbolic name specified in the device if they want to find this particular MP, as this symbolic name is what SSH/Telnet MP registers with the LS.

This is making the perfSONAR too visible to the end user who shouldn't even be aware of the MPs’ existence. It is unlikely that the user will know the symbolic name of the device, especially if the router is in some other domain that they have no control of. This problem scales to almost every perfSONAR service currently available, rendering perfSONAR practically useless in some cases.

Keeping in mind that the end user’s perspective always centers around the network, and that their objective is to find as much data as possible within the context of their survey, it is save to assume that this type of functionality is a must have.

2.3. Finding a solution

By changing our perspective of perfSONAR and looking at it as a large federated database system with perfSONAR instances as database tables, we can draw the conclusion that no defined relations exist between those tables. We need to identify the relations between the services, find a way for storing these relations and make them usable.

To do this we need to identify the means that are already available to us and use existing functionality more wisely. Taking a closer look at our example scenario we can see that the key element to a possible solution could be the Lookup Service as it contains data about deployed services that are registered with it.

If we go back to our database analogy, the registration data can be treated as a key holder or an index table of our fictional database. The RRD MA, for example, registers interface IP addresses along with the host name of the router they belong to. By querying the LS with a
router’s interface address and asking for the bandwidth utilisation, the user can get the appropriate RRD MA and the data they are interested in.

However, if the user queries the LS with the host name (the address of the loopback interface) to get bandwidth utilisation data for all the interfaces on the router, they may be unsuccessful as the host name field in the RRD MA configuration file is not mandatory and therefore might not be registered with the LS. Nevertheless, registering interfaces along with the loopback addresses of the routers is a step in the right direction.

The RRD MA has two possible view points, the interface and the host name, which represent entities that are used to query the service. Users query the RRD MA for data for a specific interface or for bandwidth utilisation data for all interfaces on a specified router. Identifying the view points of a perfSONAR service leads us to exact attribute names which have to be registered with LS and placed in service configuration files. This also implies that all identified field names in service configuration files have to be made mandatory. Making these kind of changes requires change in the NMWG schema as well. If we apply the same rules to the Telnet/SSH MP and try to identify its view points, we see that the host name is what should be registered to the LS. The user is now able to query the LS for the RRD MA that has bandwidth utilisation data for a particular interface and for the SSH/Telnet MP that has the same host name as a router that has this interface.

All other perfSONAR services should follow this simple set of rules to enable data and service correlation. Special care has to be taken in naming related identifier fields in configuration files. All related identifier fields should have the same name. For example if RRD MA uses “hostName” as field name for the address of the loopback interface, other services that use host name as a view point should use the same field name in their configuration files. A unified attribute name or view point identifier field name will bring consistency to the NMWG schema and eliminate clashing identifier field names.

We can conclude that these types of changes are easily manageable, will bring order to the NMWG schema and provide the perfSONAR framework with functionality it needs. Visualisation tool developers will be able to improve their tools and make them much more efficient, offering more functionality and improved perfSONAR transparency.

3. Automated visualisation

Automation and automated visualisation could be one of the main benefits of this new functionality, should the suggested changes be accepted. Using visualisation tools, the user would be able to get automatically all available services containing various metrics and performance data for a given network entity. For example, if the user identifies a network device they are interested in, the visualisation tool would enable them to execute commands on that device, provide them with the netflow data in raw or graphical form, show them bandwidth utilisation and packet drops for all the interfaces on that device, and so on. The user could have a set of interfaces like in traceroute output that they want to analyse. They can enter this traceroute output into a visualisation tool and automatically get all data for interfaces. With properly presented data the users would not only be able to see where the networking problem has occurred, they would also be able to find the cause of the problem more easily. Considering the fact that perfSONAR is an MDM system and that most of the interfaces or other network entities are coming from different network domains, we can conclude that this is useful and impressive functionality.

4. Summary

Looking at the service-oriented architecture, we can conclude that the service discovery is what gives the Service Oriented Architecture (SOA) its true power. PerfSONAR’s service discovery is realised through the LS using an object or an entity familiar to the end user as a key for the discovery. The discovery keys are formally described by the NMWG schema which combines the keys and the services in the same context. This is a new approach to service-oriented architecture and service discovery, and the guidelines written here are extending that approach to a point where it is possible to correlate keys and services in the same context. In other words when a user has an entity that they are interested in, they can discover all available services in the context of that entity.
By summarising the proposed changes and guidelines we can identify five action points:

1. Identify perfSONAR service view points by determining how the user queries the service data. These view points are actual keys or attribute names that have to be described by the NMWG schema, placed in the service configuration file and later on registered with an LS.
2. See if other services use the same view points and make sure to use the same attribute names as they do.
3. Make changes to the configuration file.
4. Make sure that the services register identified attributes with the LS
5. Ensure that the NMWG schema supports all these changes.

Building a multi-domain monitoring system is a highly ambitious project even for the large community of experts, software and network engineers in GEANT2. At the beginning of the perfSONAR project all obstacles and problems that this kind of MDM system would have to face were listed, including political, security and technical problems. PerfSONAR prevailed and managed to overcome all the obstacles, and now the vision of creating a multi-domain network performance system of this magnitude is just a step away from becoming a reality. I hope that this paper will contribute to turning this vision into reality.

5. Acknowledgments

PerfSONAR is a consortium of organisations who seek to build network performance middleware that is inter-operable across multiple networks. There are about 25 participants (12.5 FTE), from 17 organisations. It’s a joint effort between GÉANT2-JRA1, Internet2 and ESnet, RNP (Brazil). The University of Delaware has also joined the effort. The current version of the perfSONAR bundle, 2.2, has been deployed in multiple networks, countries and organisations all over the world.

6. References
