SONoMA: A Service Oriented Network Measurement Architecture

Béla Hullár[1], Sándor Laki[2], József Stéger[1],
István Csabai[1,3], and Gábor Vattay[1]

[1] Department of Physics of Complex Systems
[2] Department of Information Systems
Eötvös Loránd University, Budapest, Hungary
[3] Department of Physics and Astronomy
The Johns Hopkins University, Baltimore, MD, USA
{hullar,laki,steger,csabai,vattay}@complex.elte.hu

Abstract. Distributed network measurements are essential to characterize the structure, dynamics and operational state of the Internet. Although in the last decades several such systems have been created, the easy access of these infrastructures and the orchestration of complex measurements are not solved. We propose a system architecture that combines the flexibility of mature network measurement facilities such as PlanetLab or ETOMIC with the general accessibility and popularity of public services like Web based bandwidth measurement or traceroute servers. To realize these requirements we developed a network measurement platform, called SONoMA, based on Web Services and the basic principles of SOA, which is a well established paradigm in distributed business application development. Our approach opens the door to perform atomic and complex network measurements in real time, handles heterogeneous measurement devices, automatically stores the results in a public database and protects against malicious users as well. Furthermore, SONoMA is not only a tool for network researchers but it opens the door to developing novel applications and services requiring real-time and large scale network measurements.

1 Introduction

In the last 50 years Internet has grown from an academic experiment with several small attached networks to a highly interconnected heterogeneous system that spans several continents. Currently it is a network of networks that consists of millions of private and public, academic, business, and government networks of local to global scope. Besides the Internet's expansion the growing number of users and applications generate huge and more complex network traffic to be handled which poses many challenges for network operators and the network itself. As a consequence traffic control, forecasting, performance analysis and monitoring are becoming fundamental issues for network operators and interesting targets for researchers as well.

T. Korakis et al. (Eds.): TridentCom 2011, LNICST 90, pp. 27–42, 2012.

To determine the key performance metrics needed to analyze network behavior and network traffic, numerous independent network measurement infrastructures and testbeds have been developed and deployed all over the world. These infrastructures aim at helping researchers to examine many interesting aspects of the Internet like network topology, traffic behavior, one-way and queuing delay fluctuations or routing policies. Nevertheless, the way to use them is varying from case to case and sometimes very complex. In general, for performing a network measurement users have to solve many individual tasks: writing measurement scripts, deploying them to the probing nodes, executing them in a distributed manner and finally collecting the results appeared on the probing nodes. In many cases, this mechanism is much more complicated than the measurement itself.

Some ISPs provide simple public measurement services (e.g. Looking Glass), which are very popular among Internet users thanks to their convenient Web based accessibility. Their network measurement capabilities, however, are very limited and insufficient for the research community. One of the open questions in network research is how the *versatility and flexibility* of the large network measurement infrastructures can be combined with the general *accessibility* and popularity of the lightweight services.

This paper introduces a Web Service based network measurement platform, called SONoMA (Service Oriented Network Measurement Architecture) that is scalable, adaptable and open for scientists and other network developers, while its functionalities are easily accessed through a standardized interface. The Service Oriented Architecture (SOA) is a very popular principle in system design and integration concerning business applications. Naturally the key components of this principle can be used in the design of network measurement architectures as well.

SONoMA is a common and extensible network measurement framework which proposes an alternative to define and perform distributed network experiments. This SOA based approach aims to decrease the required time and efforts of network experiment implementation significantly. It enables complex network measurements that require the cooperation of different measuring nodes, while its services can be accessed via a standardized web services interface, not constraining the used programming language. We believe that the best way for taking network experiments is when researchers can use their favorite environment to describe measurement scenarios, not wasting time to discover exotic scripting languages.

To perform a measurement using this system there is only one thing to do: prescribe what you want to measure and then the framework will ensure to deliver the required measurement data. Nevertheless, the above request will be disassembled into individual executable tasks in the background. Each task will be performed on a proper set of measurement nodes in a completely distributed manner. Whilst the results are forwarded back to the user, they are automatically stored in a public data repository, called Network Measurement Virtual Observatory (VO) [9], too. In contrast to prior approaches, researchers do not have to write separate scripts to check the status of the measurement nodes,

spread the probing tasks among the nodes, then collect the results and finally post process the data. SONoMA can also interoperate with other systems, it is integrated part of the TopHat topology measurement system [3] serving as its medium precision measurement platform.

The rest of the paper is organized as follows: in Section 2 we overview the state-of-the-art approaches including prior network measurement facilities and testbeds. The key concepts of our service oriented network measurement architecture are presented in Section 3, while its implementation and the performance issues focusing on the timing overheads are detailed in Section 4. Section 5 outlines some scientific applications showing how simple SONoMA can be used to perform distributed network experiments. The final section summarizes our results.

2 State of the Art

The idea of building global network measurement infrastructures is not new. In the last decades several facilities have been developed and deployed all over the world. The mature ones like PlanetLab [11] or ETOMIC [10] provide almost full control over their geographically dispersed measuring nodes. Besides network measurements they open the door to test new network protocols and applications as well. This kind of freedom makes them general testbeds. Nevertheless, this freedom is not necessary for most of the use cases required by the network measurement community, making the development and deployment of network experiments needlessly complicated.

There are other projects like DIMES [13], which takes a different approach than building and maintaining a costly permanent infrastructure. The members of this community are volunteers installed the DIMES agent on their PC which uses the idle time of their computers to download and perform measurement tasks. The capabilities of these software agents, however, are very limited and it requires registration and the knowledge of a XML-based language for specifying network experiments. This language offers primitives only for round trip delay and traceroute measurements, which constraints the flexibility of the DIMES system.

Scriptroute [15], in contrast with DIMES, provides a general and flexible software platform for defining network experiments easily. Scriptroute is currently deployed and accessible on PlanetLab and proposes a Ruby based scripting environment which makes an ordinary PC be able to instrument network measurements remotely and safely. The weakness of this approach is the lack of a uniform mechanism for complex measurements which require the cooperation of more than one measurement nodes (e.g. bandwidth and chirp measurements, network tomography, etc.). Thus in this scenario users themselves have to build up their distributed measurements by synchronizing the active probing nodes. In addition, Scriptroute does not have data repository for storing the measurement details and results.

A very similar approach, called FLAME, was presented by A. Ziviani *et al.* [17]. In contrast to Scriptroute, this system enables measurements requiring the cooperation of different destination nodes and provides a central data repository for storing the results. To specify network experiments it uses another, less widespread scripting language, named Lua which has a simple procedural syntax and a high abstraction level aiming to reduce the software development time.

Several national and international R&E networks including GÉANT2 and Internet2 joined their forces to specify and implement a service oriented monitor infrastructure for collecting and publishing network performance data, called perfSONAR [5]. perfSONAR is based on Web Services as well as SONoMA and they have much in common in their architecture. However, there are several conceptual differences between them: perfSONAR was designed for monitoring purposes, while our approach focuses on the problems of complex network measurements, especially active probing.

There are several ways to manage resources in a network measurement infrastructure. For example, PlanetLab accomplishes a time sharing approach by running several virtual machines in parallel, while ETOMIC operates a time allocation system to ensure precise timing during the experiments. It is obvious that most of the network monitoring probes do not need dedicated resources, whereas in certain special cases precision will still require high availability of the peripherals and resources. Our architecture supports both resource managing mechanisms and introduces a sophisticated resource allocation strategy, which differs from the ones used in existing infrastructures.

Besides the above described large integrated measurement facilities, there are numerous ISPs providing lightweight measurement services like traceroute or ping, which are very popular among Internet users thanks to their public accessibility and simplicity. In contrast to the large network measurement infrastructures these basic services do not require user authorization, resource management and resource allocation. These services are advertised via their own Web based user interface, thus the heterogeneous interfaces do not facilitate to perform distributed measurements in a unified access manner. Furthermore, the measurement types they offer are mostly limited to ping and traceroute.

The approach discussed in this paper makes an attempt to integrate the complexity of large network experimentation facilities and the popularity of the lightweight services. It provides public access with reasonable limitations for performing basic measurements and gives full access to registered users for assembling complex Internet experiments. In addition, it supports synchronous services for fast and immediate measurements and asynchronous ones for long-running probes.

3 System Architecture

The idea of exploiting the advantages and flexibility of Web Service technologies in the context of network measurements is not a new one [5,8,6]. Building and maintaining network measurement repositories and sharing experimental data [16] is also an apparent trend in both engineering and scientific community.

Our system concept tries to bind the two together. Namely, we propose a three-tier system architecture that provides simple means to carry out a broad range of network probes on the one hand, and on the other the system is responsible for saving all raw network measurement data in a public repository, called Network Measurement Virtual Observatory [9]. This database provides a unified methodology to represent and publish measurement data and makes reproducibility possible. For example if different new methods become available to estimate some characteristics of the network they are easily fed by historical data for the sake of comparison.

In this section we briefly outline the web service technologies and then present our SOA based system design.

3.1 Web Services

Web Services Technology provides interoperability between different computers, platforms and applications to build up a distributed system. It has many advantages: First, Web Services are independent from any operating systems and programming languages. Second, they offer a simplified access to remote procedures. This platform independent approach is very useful in a distributed environment where a lot of systems interact. In the last decades it has became a mature technology which contains numerous standards and extensions like WS Security for using encryption and signatures to secure message exchanges. In addition, most of the programming languages support the creation and the use of Web Services and offer tools for the automated code generation.

The Service-Oriented Architecture (SOA) is an effective distributed system development paradigm based on Web Services. In this approach the system is made up of loosely coupled services which communicate each other through the network. The deployed services can be used by other organizations and companies according to the well defined interfaces. This system design with the standardization enables effective and strong cooperation between different participants of a field.

The SOA based system design is currently under adaptation in the field of network management[1]. Although in the literature there are only few proposals to coordinate network measurements based on the Web Services technology, a mature, well standardized solution is still missing.

3.2 Architecture Overview

Fig. 1. shows the architecture of the system. **Measurement Actors** (A) are the topmost entities in this model. In the most natural case Actors are user applications implemented in any programming or scripting languages capable of integrating Web Service functionalities. The main advantage of this approach is that user can simply embed network measurements into their applications seamlessly. They can process the results within that framework and/or trigger signals

[1] e.g. ETSI TS 188 001 NGN, ITU-T M.3060.

based on the evaluation of raw measurement data received in turn of calling the services. This model framework also offers a more convenient solution for simple measurement orchestration.Namely, Actors could also be an elaborated graphical user interfaces served via the world wide web. In both cases a well defined programming interface is provided to contact with the middleware, the Management Layer.

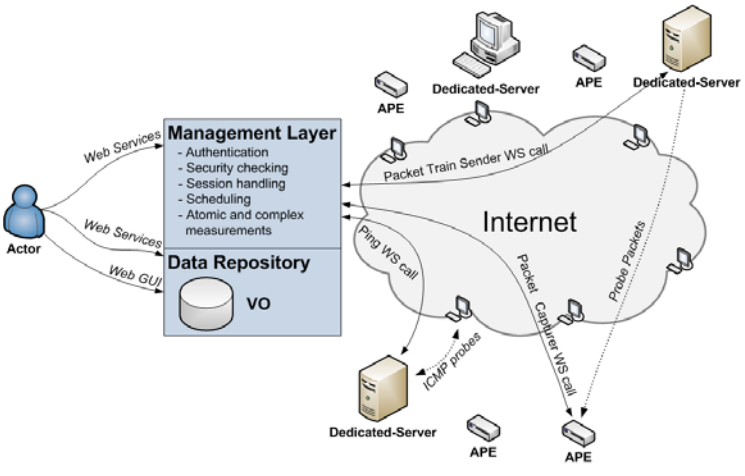


Fig. 1. Architecture overview. Actors, the Management Layer and Measurement Agents form a 3-tier framework. Data are archived in the public Network Measurement Virtual Observatory (VO).

The **Management Layer** (ML) is the second layer in this model and is responsible for implementing and serving the following fundamental operations: i) accounting both Actors and Measurement Agents, ii) authenticating Actors, authorizing requests and checking them against misuse, iii) handling measurement sessions, scheduling and composing experiments. In addition, this layer provides an access to Actors for submitting their complex or atomic measurements as well. These tasks are detailed in Fig. 2.

The **Measurement Agents** (MAs) reside at the lowest level of the proposed model. Conceptually, any network entities implementing the required web service interface can be a MA. However, it does not need to offer all the available network measurements that are defined in the SONoMA framework, since the ML keeps track of the capabilities of all the MAs. All the MAs are directly connected to the ML through which Actors can control them.

In the proposed system we apply two kind of classifications of the network measurements based on their timing and complexity. From user's point of view we differentiate between synchronous and asynchronous measurement calls. In the former case network measurements are parametrized in a way, that results are available within a few seconds. Thus the measurement results will be returned to the Actors directly and the code calling the service functions will be blocked until

the measurement ends. In the other case experiments could be parametrized for long runs and only a measurement reference is returned, which makes it possible to try and retrieve results later, when they are available. In both cases the raw data produced by MAs are simply stored in the VO.

The other classification scheme is based on the measurement complexity. We define atomic measurements and complex measurements. Atomic measurements are the simplest building blocks of any network measurement running on a single MA (e.g. traceroute, ping, etc.). These measurements (or a subset of the possible ones) are offered by MAs directly. However, complex ones are realized at ML level since they require cooperation and synchronization of multiple MAs and ,in most cases, combine several basic measurements (e.g, bandwith measurement, network tomography, geographical localization, etc.). In this case, ML has to collect and match pieces of information properly together, just like aligning a probe packets time stamps upon emission and reception. In addition, a complex measurement may contain post processing, data formatting and evaluation phases as well, and the aggregated or evaluated values like queuing delay histograms or location estimates will also be stored in the VO along with the raw measurement data like one-way delays or round-trip times required for the calculation of these network characteristics.

In contrast to prior works, our model enables both time-sharing and time-reserving measurement approaches. It is dependent on the requirements of the given network measurement, which are formulated in measurement rules by the ML. For example, while topology discovery and echo measurements do not require dedicated resources, thus `traceroute` and `ping` tools run in time-sharing mode, the network tomography measurement [14], which requires the generation and sending of highly correlated IP packets and precise time stamping is definitely executed in time-reservation mode.

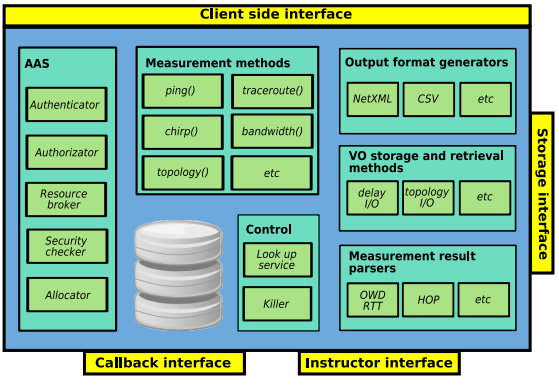


Fig. 2. The internal structure of the Management Layer middleware

4 Implementation Details

We have implemented the key elements of the proposed architecture including the Management Layer (interfacing to VO), two types of Measurement Agents, and a few Actor examples, as well. We separated the different functionalities and worked out a clear design, which makes the system adaptable to the new demands simply. This section overviews the building blocks of the system and the thoughts behind their implementations.

4.1 Measurement Actors

In contrast to FLAME and Scriptroute, SONoMA's users are free to realize their Actors in the programming environment they use everyday, since web service technology is supported by almost all programming languages. ML offers a service descriptor file (WSDL in document literal format) from which Actors can generate the interface code automatically. Then this interface code can be used to access all the measurement services through simple function calls (see Sec. 5 for examples). For demonstration purposes we developed a simple web accessible application where a few synchronous measurements are available [1].

4.2 Management Layer

An elaborate view of our Management Layer implementation is depicted in Fig. 2. The ML provides four operational interfaces, all dedicated for different purposes. i) The *Client side interface* is an input/output interface offered for the Actors. All the functionalities related to requesting measurement sessions, carrying out the measurements themselves and methods of data retrieval are described here in a WSDL description. The rest of the interfaces are hidden from the Actors. ii) The *Instructor interface* as for its functionality is similar to the Client side interface residing at one level lower, between ML and MAs. Note that the WSDL description of this interface is constrained to the atomic measurement methods only. iii) The function of the *Callback interface* provided to MAs are twofold. Most importantly it makes possible to release MA resources after finishing its measurement tasks in advance, which may occur from a bad experiment duration guess by the ML, so MA running long processes can notify the middleware of process termination and trigger the necessary steps of data retrieval. In addition, this interface is also responsible to handle MA notification of their waking up or shutting down. iv) The raw data of each and every atomic measurement is loaded to the VO via the *Storage interface*.

Considering the time line of a measurement, the modules of Management Layer split into three sets: i) services and tasks to be invoked before measurement, ii) services and modules, which are responsible for managing measurements, and iii) modules that pre-process and store raw measurement data.

The first group, the *Authentication, Authorization & Session handling* (AAS) collects the modules to call prior to measurements. The tasks of *Authenticator* and *Authorizator* is self-explanatory. Naturally, prior to granting Actors a

session and the proper privilege schema, Authentication module checks their credentials. Besides federating Measurement Agents of different network measurement architectures, SONoMA tries to make its services appealing to the possibly different set of users. Thus currently, PlanetLab Europe users can also access to the system without the need to apply for a SONoMA account. In this case the *Authenticator* automatically calls the PLE-RPCs to decide about the access.

The session binds a bunch of network measurements of an Actor together, which is also represented in the databases of the VO. The session also encapsulates the overall requests of an Actor such as the format it expects to receive the measurement data. The privilege schema, which is checked during measurement authorization, distinguishes between different Actors, e.g. an Actor authenticated as guest has no privileges to run asynchronous long measurements and quotas are introduced on the frequency they use the system, whereas respected Actors have larger freedom to exploit the capabilities of the system. The *Allocator* and the *Resource broker* modules are responsible for checking if the chosen MA is capable of a given experiment and whether the required resources are available. If the experiment requires a time-reserving operation then a certain estimated time interval will be allocated and parallel measurement requests will be omitted. The *Security checker* uses heuristics to filter out the unlucky combination of the experiment parameters, which may lead to malicious or blocking traffic. It also maintains a gray list of MAs that have some constraints on their atomic measurement methods, e.g. operators of MAs connected to low speed links typically would not like probe packet generation at a high rate.

The second set of modules, *Control* and *Measurement methods* manage the network measurements. Lookup service provides up-to-date status information of the MAs according to different filtering rules, like the nodes that are available and/or are capable of a certain measurement type or may inform Actor when a busy MA becomes free again.It is also the Lookup service's duty to handle signals from the Callback interface to initiate data retrieval from the given nodes or marking properly their operational information. The Killer service can be used to terminate a measurement and free allocated resources in case when the Actor is not interested in the result any more. The module named Measurement methods contains both low level measurements (like `shortPing`) offered by MAs and complex ones implemented at Management level (like `getAvailableBandwidth`).

The third group of modules operates on the raw measurement data. *Measurement result parsers* provide classes for each basic measurement type to read, represent and combine raw measurement logs. *VO storage and retrieval methods* build up database connection on demand and embody a queue to store raw measurement data. In addition, after raw data are emptied from SONoMA caches retrieval methods pull back information from VO to answer Actor requests. *Output format generators* serialize raw measurement data in the requested format (e.g. NetXML, CSV, etc.).

4.3 Measurement Agent

In the current version of the SONoMA system Measurement Agents are running on PlanetLab slices and on the Active Probing Equipments (APE) installed at OneLab [4] sites. While the agents running on PlanetLab are based on ordinary PC architecture, APE is an embedded, real-time system which provides an active probing platform with GPS synchronized precise packet time stamping hardware add-on. Implementing the two kind of agents required different programing libraries and environments, but ML and the system users do not experience any differences since the web services technology hides the details.

4.4 Measurement API and Data Interface

All the measurements and their results can be accessed through the Client Side Interface, which is the most important part of the system from the users point of view. However, the SONoMA system automatically archives the results of every productive measurement invocations, VO also offers tools to access or to sophisticatedly manipulate SONoMA's measurement data.

In Table 1 we enlist the currently deployed measurement services and also indicate those we plan to extend SONoMA with in the near future. Keep in mind

Table 1. The list of the supported measurement services *(under implementation)*

Function name	Short description
`getVersion`	queries the current version of the ML
`requestSession, closeSession`	opens a new session and close the current one
`getNodeList`	returns the list of measurement nodes according to the given type parameter
`shortPing, longPing`	synchronous and asynchronous ping measurement
`shortTraceroute, longTraceroute`	synchronous and asynchronous treaceroute measurement
`paralellPing, paralellTraceroute`	asynchronous ping and traceroute measurement towards different destinations
`ensamblePing, ensambleTraceroute`	asynchronous ping and traceroute measurement towards different destinations from different sources
`shortChirp, longChirp`	synchronous and asynchronous chirp measurement from a source MA to a destination MA
`getAvailableBandwidth`	performs bandwidth measurement between two measurement nodes
`shortTrain, longTrain`	synchronous and asynchronous back-to-back packet train sender from a source MA to several destination MAs
`topology`	performs traceroute measurements between a set of MAs and gives back the topology graph
`getProcessInfo`	queries the status of a submitted async. measurement
`terminateProcess`	kill all busy measurement compenents of a submitted async. measurement
`getResults`	returns the results of an asynchronous measurement
`queuingDelayTomography`	computes the distribution of queuing delay fluctuations on the topology spanned by the measurement nodes
`queuingDelayVariance`	computes the variance of queuing delay fluctuations on the topology spanned by the measurement nodes
`geographicalLocalization`	performs delay and topology measurements to localize the given IP address
`pcapSender`	emits a general packet pattern described by a standard pcap file

the fact that in case of issuing short measurements, results will be returned by the web service immediately after their ending, whereas for long measurement calls SONoMA provides the `getData` method to fetch partial information during or all information after the execution of a long measurement task.

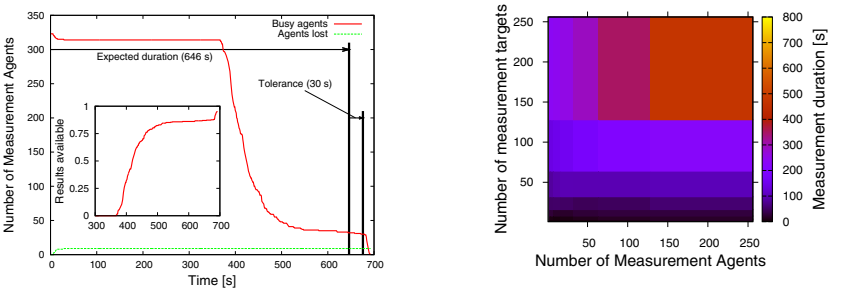
The Network Measurement Virtual Observatory presented in [9] is an efficient approach to store and share research data. Beyond archiving functions VO aims at providing easy-to-use analysis tools via both human and machine readable interfaces. One of the advantages of the VO is that it provides SQL access for the repositories, thus the users can edit and run their customized queries even including joint data. The main power of this solution is that the researchers can filter out the relevant information from the huge archives on server side efficiently.

There is straightforward binding between the SONoMA's and VO's data chunks, namely SONoMA sessions and processes are mapped one-by-one to collections and measurements in the repository, respectively.

4.5 System Performance

Inserting middleware components into the communication chains introduces extra overheads, which may appear in message lengths and also in the duration of the communication. In order to keep the lengths of messages between SONoMA components as short as possible,data compression is used: raw data delivered from a MA to the ML is always compressed, while between the ML and the Actors it depends on the session setup. To minimize the communication cost, when cooperating multiple MAs is necessary SONoMA calls them in a parallel fashion and different threads fork off to take care of data retrieval transactions.

In Fig. 3(a) an `ensambePing` measurement is carried out between all available agents (323 MAs in the example). Based on the `getProcessInfo` SONoMA method user can follow the state of their asynchronous running measurements, which returns the number of busy agents.It can also be seen on the figure that a few MAs, marked with green, (9 in our case) do not respond in a certain time out threshold, so the system marks them dead. Obviously, the time it takes for each MA to probe the targets varies a lot, since MAs have network access of different speeds, the scheduling performance of MAs themselves span, and also the firewall rules may differ according to the location from where a probe packet comes. This phenomenon can be read from the red curve in the main figure showing the number of active agents, to complement a MA after termination triggers data retrieval, so in the inset of Fig. 3(a) the percentage of available measurement log records increases monotonously. The figure also shows the expected duration of this particular measurement scenario and the overtiming tolerance the system bears (at 646 and 673 seconds respectively). In case, a measurement does not finish within the tolerated time interval, SONoMA sends termination signals to the busy MAs, and fetches their partial results if any. Note that more than 85% of the MAs finished within the tolerated time frame, and 75% finished $1/3$ of the duration earlier, so an Actor implementation may consider this fact and when having statistically enough measurement logs it can terminate the process even earlier.



(a) Evolution of a full-mesh round-trip-delay measurement between all available MAs.

(b) The overall duration of `ensamblePing` experiments in function of the number of MAs and targets.

Fig. 3. Performance test results

In Fig. 3(b) the scaling of the `ensamblePing` measurement is investigated. In the test case the number of MAs and the target nodes were varied. Measurements were terminated either by the system or if more than 75% of the agents delivered any results the Actor issued the `terminateProcess` method. The time between startup and termination is presented in the heat-map and the following observation can be made: i) for few targets the duration is independent of the number of MAs, ie. the launching overhead is small ($\approx$ 5 s when using $>$ 300 MAs) ii) the number of targets is linearly proportional to the duration; iii) when using more and more MAs the probability of choosing very *slow* nodes shifts from user termination to system termination.

5 Case Studies

To demonstrate the benefits of the SONoMA system, we show three use cases. First, we highlight the topology discovery experiment, which is available as a complex measurement, then we mention two composite examples: the tomography measurement, which is planned to be integrated into SONoMA, and an active IP geolocation application, called Spotter.

Note that the main steps of any program flow are i) instantiate the web service and request a session, ii) run the desired measurement and collect data, iii) close session and post-process the results.

5.1 Topology

Prog. 1 implements an asynchronous measurement to discover and draw the connectivity graph spanning among a set of MAs. When the user requests a session, a description of the desired result format (comma separated values, zipped output in this case) is passed. Next, the web method is called with the proper parameters (indicating the node set) which returns an unique processID identifier. As it can be seen on Fig. 4 this complex measurement is disassembled into the execution of individual `parallelTraceroute` probes, whose results are

cached by SONoMA, and when all MA delivered the raw data, they are pushed into the VO individually. When all measurement components are finished, the Actor is able to retrieve the set of links using the `getData` method. If raw data are not in cache any more, ML retrieve them from the VO repository before the construction of the link set to be returned.

Prog. 1. The pseudo code of a topology discovery measurement.

```
# instantiate web service
ws = ServiceServerSOAP( "http://sonoma.etomic.org:8888" )

# request a session
sessionID = ws.requestSession( user="User", zipResults=True, formatResults="CSV" )

# do the measurement
procID = ws.topology( sessionID, nodeList = [ "157.181.175.247", "132.65.240.38", ... ] )
(duration, working, ready) = ws.getProcessInfo( sessionID, procID )
# wait for some time & retrieve data
result = ws.getData( sessionID, procID )

# terminate session and post-process data
ws.closeSession( sessionID )
drawGraph( decompress(result) )
```

5.2 Spotter - Geolocation Service

Spotter [2,7] is a novel geolocation approach that aims to determine the geographic position of Internet hosts. In contrast to the existing services this approach is based on active delay measurements between the target (the computer to be localized) and the geographically dispersed landmark nodes (with known geographic position). The required measurements are performed real-time from more than 200 measurement agents of the SONoMA system. To determine the most probable position of the target Spotter uses a detailed statistical analysis of the delay-distance space. By exploiting the benefits of this statistical description this approach is less prone to measurement errors and other network anomalies than the state-of-the-art active geolocation techniques. In addition, for a given target Spotter returns not only the estimated coordinates, but also a spatial probability surface describing how likely the target is at given regions of the globe. Figure 5(a) illustrates how Spotter uses SONoMA for collecting the distributed delay measurements required by the geolocation process.

This service shows that SONoMA is not only a tool for network researchers but it opens the door to developing novel applications and services requiring real-time large scale network measurements. Thanks to its standardized Web Service interface, it can be easily integrated into any software components.

5.3 Queuing Delay Tomography

In [14,12] it has been shown that the ETOMIC system provides a high precision measurement platform which is suitable for inferring queuing delay information.

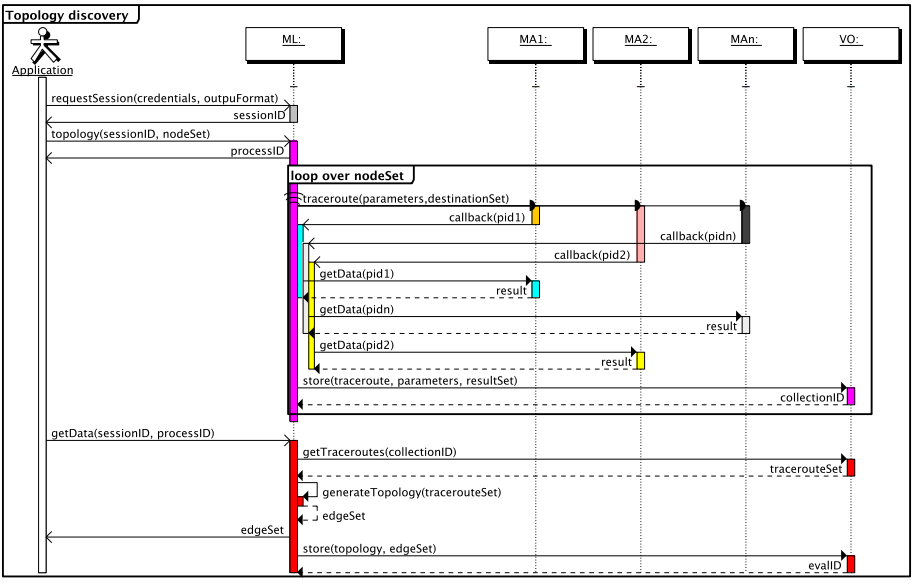
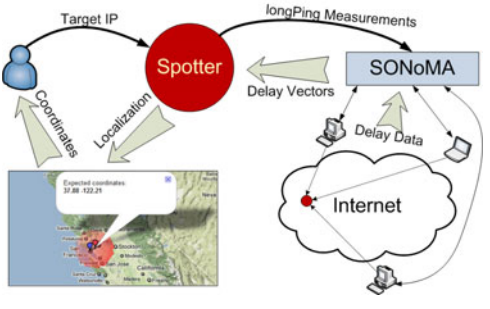
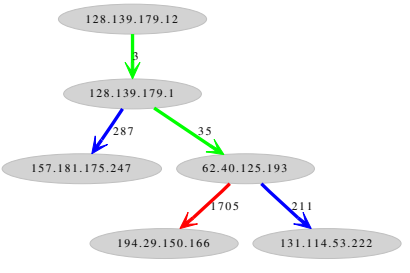


Fig. 4. This sequence diagram illustrates how a topology discovery is disassembled into atomic traceroute measurements running on MAs



(a) Spotter - Geolocation Service. This system uses SONoMA platform for performing active measurements from the landmarks to the target node to be localized.



(b) Tomography: Queuing delay probability distributions are inferred spanning between 4 MAs. The average delay in μs is indicated for each segments.

Fig. 5. Example applications using SONoMA active measurements

The MAs running on APE devices have also fair precision time stamping capability, in the order of μs, thus it is straightforward to use SONoMA as well to implement an Actor doing tomography measurements. Although hidden by the SONoMA framework, it also seems reasonable and easy to define cross platform tomography measurement based on the cooperation of APE and PlanetLab MAs, where APE's clocking serve for timing reference, too. In Fig. 5(b) we show the end results of a queuing delay inference, for which 4 APEs provided input through the SONoMA system. Internal segments are colored and labeled

according to the average queuing delay inferred in μs units. To perform this experiment we just changed the measurement data loading part of our extant tomography code to a `parallelTraceroute` and a `longTrain` call.

6 Conclusion

In this paper we introduced a novel network measurement architecture, SONoMA based on Web Services. In the proposed three-tier system users reach the Measurement Agents through an intermediate Management Layer. This middleware controls the access, offers atomic and complex measurements, hides the heterogeneity of the agents and stores the results in a public repository automatically. This concept enables the uniform handling of active and passive measurements and allows both time sharing and time reservation resource allocation schemata as well. Furthermore we have presented our publicly available system, whose agents are currently implemented and deployed on PlanetLab nodes and on the APE platform. Finally, we demonstrated the system's benefits with some application use cases including an existing IP geolocation service, called Spotter, which uses SONoMA as real time measurement platform. In addition, the presented infrastructure provides access to simple and complex network measurements in an easy and publicly accessible way.

Further research is needed to increase the number of measurement types both at complex and atomic levels, to integrate WS-Security into the authentication and user identification process as well as to examine the performance and the efficiency of the system implementation. The adaptation of volunteer, desktop grid like agents, that work well in other projects like DIMES is also among our future plans. In addition, the number of Measurement Agents will be extended with the precise active measurement infrastructures of the OneLab2 project and, of course, we count on other institutions to join the SONoMA platform [1] as users or as Measurement Agent operators as well.

Acknowledgment. The authors thank the partial support of the National Office for Research and Technology (NAP 2005/ KCKHA005), OTKA-80177, the EU FIRE NOVI project (Grant No.257867) and the EU ICT OneLab2 Integrated Project (G.A.No. 224263) and the project TAMOP4.2.1/B-09/1/KMR-2010-0003 of the National Development Plan.

References

1. The sonoma web portal (2010), `http://sonoma.etomic.org`
2. Spotter geolocation service (2010), `http://spotter.etomic.org`
3. Bourgeau, T., Augé, J., Friedman, T.: Tophat: Supporting Experiments through Measurement Infrastructure Federation. In: Magedanz, T., Gavras, A., Thanh, N.H., Chase, J.S. (eds.) TridentCom 2010. LNICST, vol. 46, pp. 542–557. Springer, Heidelberg (2011)

4. Csabai, I., Fekete, A., Hága, P., Hullár, B., Kurucz, G., Laki, S., Mátray, P., Stéger, J., Vattay, G., Espina, F., Garcia-Jimenez, S., Izal, M., Magaña, E., Morató, D., Aracil, J., Gómez, F., Gonzalez, I., López-Buedo, S., Moreno, V., Ramos, J.: ETOMIC Advanced Network Monitoring System for Future Internet Experimentation. In: Magedanz, T., Gavras, A., Thanh, N.H., Chase, J.S. (eds.) TridentCom 2010. LNICST, vol. 46, pp. 243–254. Springer, Heidelberg (2011)
5. Hanemann, A., Boote, J.W., Boyd, E.L., Durand, J., Kudarimoti, L., Łapacz, R., Swany, D.M., Trocha, S., Zurawski, J.: PerfSONAR: A Service Oriented Architecture for Multi-Domain Network Monitoring. In: Benatallah, B., Casati, F., Traverso, P. (eds.) ICSOC 2005. LNCS, vol. 3826, pp. 241–254. Springer, Heidelberg (2005)
6. She, W., Zhang, J., Yang, J., Zhang, M.: A new conceptual distributed network measurement architecture based on web service. In: 14th IEEE International Conference on Networks, vol. 2, pp. 1–7 (2006)
7. Laki, S., Mátray, P., Hága, P., Sebők, T., Csabai, I., Vattay, G.: Spotter: A model based active geolocation service. In: IEEE INFOCOM 2011, Shanghai, China, April 10-15 (2011)
8. Allman, M., Martin, L., Rabinovich, M.I., Atchinson, K.: On Community-Oriented Internet Measurement. In: Claypool, M., Uhlig, S. (eds.) PAM 2008. LNCS, vol. 4979, pp. 112–121. Springer, Heidelberg (2008)
9. Mátray, P., Csabai, I., Hága, P., Stéger, J., Dobos, L., Vattay, G.: Building a prototype for network measurement virtual observatory. In: ACM SIGMETRICS - MineNet (2007)
10. Morató, D., Magana, E., Izal, M., Aracil, J., Naranjo, F., Astiz, F., Alonso, U., Csabai, I., Hága, P., Simon, G., Stéger, J., Vattay, G.: The european traffic observatory measurement infrastructure (etomic): A testbed for universal active and passive measurements. In: Tridentcom 2005, Trento, Italy, February 23–25, pp. 283–289 (2005)
11. PlanetLab. An open platform for developing, deploying and accessing planetary scale services (2003), `http://www.planet-lab.org/`
12. Rizzo, T., Stéger, J., Csabai, I., Vattay, G., Pollner, P.: High quality queueing information from accelerated active network tomography. In: Tridentcom (2008)
13. Shavitt, Y., Shir, E.: Dimes: Let the internet measure itself. ACM SIGCOMM Computer Communication Review, 71–74 (October 2005)
14. Simon, G., Stéger, J., Hága, P., Csabai, I., Vattay, G.: Measuring the dynamical state of the internet: Large scale network tomography via the etomic infrastructure. Complexus (2005)
15. Spring, N., Wetherall, D., Anderson, T.: Scriptroute: A public internet measurement facility. In: USENIX Symposium on Internet Technologies and Systems (2003)
16. Szalay, A.S., Budavári, T., Mali, T., Gray, J., Thakar, A.: Web services for the virtual observatory. In: SPIE Conference on Advanced Telescope Technologies, vol. 4846 (2002)
17. Ziviani, A., Gomes, A.T.A., Kirszenblatt, M.L., Cardozo, T.B.: FLAME: Flexible Lightweight Active Measurement Environment. In: Magedanz, T., Gavras, A., Thanh, N.H., Chase, J.S. (eds.) TridentCom 2010. LNICST, vol. 46, pp. 526–541. Springer, Heidelberg (2011)