Towards Measurement Consolidation for Overlay Optimization and Service Placement

Dirk Haage, Ralph Holz
Fakultät für Informatik
Technische Universität München, Germany
{haage,holz}@net.in.tum.de

Abstract

Overlay networks and distributed services are currently favorite approaches to add new functionality to the network. Often, these approaches strongly rely on accurate information about the current state of nodes and network to increase performance, efficiency and user experience. Based on previous work, we show how UNISONO provides this information while hiding the complexity of the measurement task. UNISONO is a generic framework for host-based distributed network measurements. In this paper, we present UNISONO as a way towards consolidated measurement for the creation of overlays as well as for service placement. We give an overview of the UNISONO concept as well as security considerations of such a service and show how distributed services benefit from its usage.

1 Introduction

Overlays seem to be one way of bringing more functionality to future networks. They can be used for services like media streaming, but are also discussed as a way to enhance grids or Service-Oriented Architectures [1]. It is a generally accepted fact that cross-layer information – i.e. measurement and host information from several layers in the network stack – can be very useful for the optimization and re-structuring of such overlays. Such information can be of much interest in SOA environments or virtualized environments. The nature of these services – i.e. a complex service is cut into small parts, each provided as a service of its own and distributed over the network – offers an interesting possibility: fast relocation or reinstantiation of the service itself, depending on the needs of the user or the situation within the network. An optimization strategy – independent of the approach – is likely to rely on accurate information about hosts and network. This opens new challenges.

First, the task of information gathering is complex and needs to be adapted to the substrate as well as to available resources, used technology etc. E.g. wireless networks have different characteristics compared to wired networks so that measurement algorithms provide inaccurate or wrong results. Addressing this separately for each application, service or overlay is a waste of resources. It is likely that application developers will either use the same algorithm in all environments or none at all.

Second, with more applications relying on information that has to be obtained by active measurements, the measurement noise within the network increases substantially. This wastes the limited resources of the network. Furthermore, each concurrent measurement within a distributed system adds bias to the measurement result, rendering them hardly usable [3].

In this article, we present UNISONO, the cross-layer measurement component used by SpoVNet (Spontaneous Virtual Networks) [8] for optimization purposes. Together with CLIO – the connector to overlays – this is an implementation of the split architecture we have proposed in [4]. The service encapsulates measurement methods and logic.

In the following, we will provide an overview of UNISONO concepts and describe its functionality. Afterwards we take a glimpse on security issues that have to be considered for a distributed measurement service and present

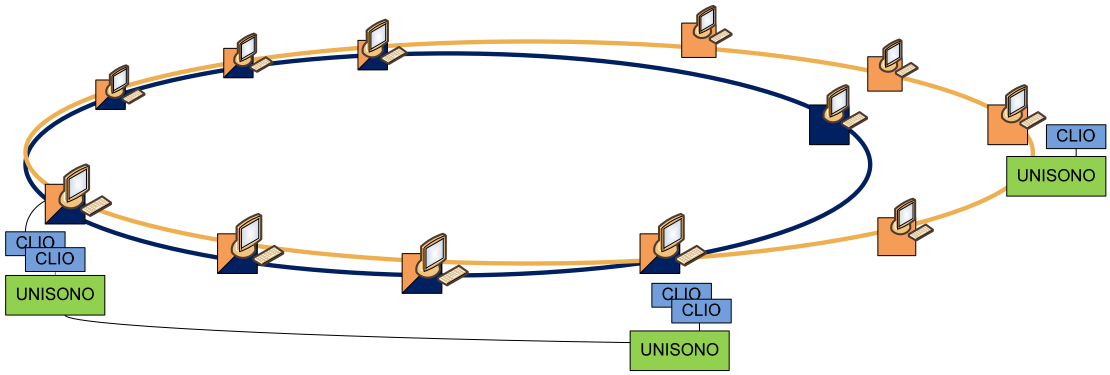

Figure 1. Using UNISONO with the CLIO connector for overlays. Nodes painted in two colors participate in two overlays, the others only in one. Note that UNISONO runs only once on each node, allowing for aggregation of measurement information.

some countermeasures. We show on the basis of two examples how UNISONO is already used, and additionally how a SOA environment can benefit from it. We conclude with its current status and future work.

2 UNISONO Framework

UNISONO is realized as a local service that only accepts queries from local applications. Queries are issued as *orders* in an asynchronous way via a simple interface. They can be one-shot, but also periodic with result delivery in fixed intervals or on exceeding a given threshold. The information is gathered with several measurement methods. These are encapsulated in modules, adhering to a plugin-concept. This enables easy extension of UNISONO with more methods and the quick exchange of algorithms. It is also possible to have more than one method for each purpose.

The main objective of UNISONO is to reduce the number of measurements within the network, especially for active measurements such as bandwidth estimations or topology detection. A first step towards this goal is already taken by cutting out the measurement functionality of applications. This allows aggregation of overlapping requests and efficient caching of results. If two queriers need the same measurement within a certain time frame, the measurement is only executed once.

The measurement logic is encapsulated in UNISONO, keeping applications light-weight and unaware of the underlying mechanisms. Furthermore, instances of the same service on different hosts are able to coordinate measurements and exchange results. It also allows distributed caching of measurement results.

In addition to network and link properties, UNISONO provides node properties via the same interface. Among them are CPU properties, available RAM and HDD, but also host uptime, host interfaces (like WLAN) and battery state. Compared to SNMP [2], this permits additional processing like the computation of trends and change probabilities directly within UNISONO. Aside from that, the application uses a single source to get information and only needs to implement one interface.

If UNISONO is used together with an overlay, the interface to UNISONO is usually implemented in the component we call CLIO (Cross-Layer Information for Overlays). Figure 1 shows this setup. CLIO contains the logic to manage orders and return results, but in contrast to UNISONO this is done for the local overlay node, using the address space and identifiers of the overlay. CLIO is also responsible for resolving overlay identifiers to underlay identifiers, which are then passed to UNISONO. CLIO has one further important purpose: it allows to provide advanced functionality in the context of overlay measurements.

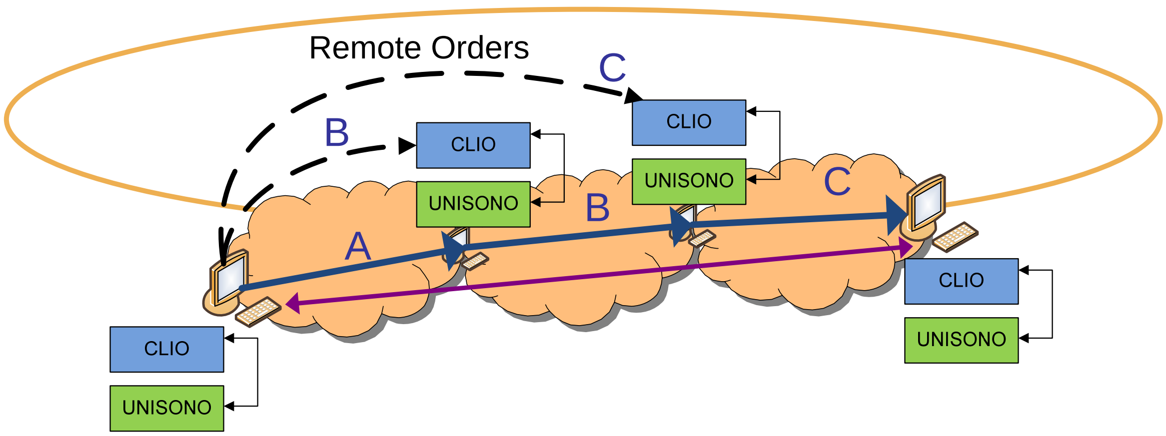

Figure 2. Conducting measurements across connectivity domains. Remote orders are used to request measurements from nodes in other connectivity domains.

3 Advanced Functionality

CLIO/UNISONO offers some measurement options that advance beyond what current measurement frameworks could offer for the optimization of overlays, especially if used combined.

Measuring Across Connectivity Domains Measurements across middleboxes like NATs are usually difficult. Some overlays allow to detect the boundaries of connectivity domains and can return the relay path (forwarding nodes at the boundaries) to a requester. Ariba [6], also a SpoVNet component, is an example. With a call to Ariba, CLIO can determine the relay path and split a measurement into the corresponding domain-internal measurements. These are aggregated upon completion and the result computed. Without Ariba-like overlays, this method is not an option; but we can still use overlay measurements, see below.

Overlay Measurements CLIO/UNISONO can be used for message-based overlay measurements. UNISONO can ask CLIO to send messages of a certain size to another CLIO instance (in the same overlay) and wait for a reply. Messages are time-stamped upon sending and receiving. This can be used for, e.g., latency and bandwidth measurements on overlay level.

Remote Orders With the help of CLIO, UNISONO can conduct remote measurements. A node A can request a measurement from a node B to a node C. CLIO is used to send and receive remote orders (requests for measurements) over the overlay to which it is connected. Remote Orders can be very useful for some optimization purposes. They also allow to measure a node from different points in the network and aggregate the results (cooperative measurements).

Aggregation and Low Impact Distributed measuring and aggregation is a central concept in CLIO/UNISONO. Recent results are held in a cache. If two queriers need the same measurement within a certain time frame, the measurement is only executed once. The cache is currently an in-memory SQLite database, with confidence intervals for measurement types pre-defined.

Complex Cross-Layer Measurements Each network technology might require specialized measurement procedures for the same information that is to be obtained. The plugin-based architecture of UNISONO allows the existence of more than one plugin for the same task. It is thus possible to combine multiple plugins into a new one to carry out cross-layer measurements. One example is the usage of a layer 2 technology detection and selection of the correct procedure for a bandwidth measurement based on its results.

4 Security Considerations for Distributed Measurements

CLIO/UNISONO was designed with security in mind. Communication between CLIO instances of a SpoVNet is authenticated, encrypted and integrity-protected. UNISONO instances only communicate via CLIO, and UNISONO only allows access from pre-registered CLIOs on the local host. A Policy Engine is a central component. Policies define which measurements a UNISONO instance allows. Access is defined with role-based ACLs. Access rights are granular: it is possible to assign a maximum resource usage or a rate limit. These mechanisms can be used together to limit the impact of malicious nodes in the network trying to conduct Denial-of-Service attacks on the local host or on a remote host. Local protection can be achieved with appropriate policy settings (for outgoing measurements and incoming remote orders). However, an attacker could attempt to request remote measurements from a high number of nodes to one single node (a type of DoS). This is more difficult to defend against. One can either use very prohibitive local policies (e.g. disallowing most remote orders). Or one can use *destination limits*. These are rate limits that define the maximum number of remote measurements that may be conducted to any given destination. However, this will only work if all or most nodes use similar policies. This is the case in SpoVNets, but more difficult to achieve in other settings. The effectivity depends largely on the number of malconfigured nodes in the network and the number of nodes known to the attacker (to whom he can send remote orders). We have done some first rough estimates and believe that our protection can be sufficient for small networks (in the order of 10,000 nodes), but becomes much less effective in larger settings. For such networks, we recommend to disable remote orders entirely.

5 Optimization of Distributed Services

While the main target of UNISONO was the organization and optimization of overlay networks, the collected information can be used in a more generic way. We give three examples how UNISONO can be instrumented for optimization purposes. The application-layer multicast (ALM) service[5] and the event notification[7] within SpoVNet [8] already do this while the placement of SOA services in the network is the generalization of this concept.

Application-layer Multicast Service The Multicast Service within SpoVNet utilizes the information provided by UNISONO in multiple manners. First, it is possible to structure a distribution tree for different metrics. The current implemenation allows a formation based on delay demands or on bandwidth demands. In WLANs, nodes can additionally be grouped according to their access points. This allows the usage of a more efficient data distribution within these wireless networks by using characteristic of a shared medium. Furthermore, UNISONO provides a way to detect IP multicast connectivity between nodes. With this information, the ALM service is able to use the IP multicast infrastructure where possible (e.g. within an AS or a company network) and connect these "islands" via its application-layer functionality.

Distributed Event Service The event service provides SpoVNet-based applications a fully distributed publish/subscribe system. With this, it is possible to provide, aggregate and distribute any type of information. Furthermore, it features event correlation. To support the distributed nature of a SpoVNet-Overlay the service is splitted into three parts: a publisher, a correlator and an aggregator. Each of these parts can be placed on any node within the overlay. The placement of each component is crucial for the service to efficiently and within certain QoS restraints. Furthermore, dynamically changing groups of nodes interested in different parts of the provided information need individual (re-)placement. UNISONO provides the information about involved nodes and network links with little overhead, allowing the event service to achieve its goal without a deep knowledge about the underlying network. The service utilizes network delay and bandwidth estimations as well as available computational resources (mainly CPU and available memory) for its multi-dimensional optimization. Without UNISONO, this information must be collected by the service itself, resulting in more complexity and resource usage.

SOA Service Placement Taking the above described event service as a starting point, we can generalize the concept for service-oriented architectures. Similar to the given example, SOA services provide small tasks that can be plugged together to provide new, complex services. Placement of such services within the network is currently mostly static. Realizing a SOA service as a virtual network appliance (i.e. as small virtual machines) would not only allow fast re-placement of the service but also replication. The automation of these processes requires information about the underlying network as well as about the node a service is running on and the usage of the service itself. This cross-layer information is the same as required by overlays, hence the information already provided by UNISONO. Although we believe it is still a long way away, our vision would be a self-organizing SOA environment, where services are provided where needed, with more efficiency as can be provided by a centralized administration.

6 Ongoing Work

While we are continously adding more plugins for measurement procedures, correlation mechanisms and statistics to UNISONO, we are also considering to extend UNISONO with an overlay of its own to optimize communication between instances as well as the distributed caching strategies. We are currently focusing on correlation mechanisms for underlay and overlay measurements (e.g. stretch for RTT). UNISONO will be extended to raise statistics, which may help in investigating dynamically chosen caching times.

7 Conclusion

We have presented UNISONO, our framework for cross-layer measurements in distributed environments. UNISONO provides generic access to node and link information that is crucial for efficient ressource usage and service placement in both, overlay networks and SOA environments. The algorithms used for measurements are dynamically chosen for optimal results. The consolidation of all measurement functionality in UNISONO significantly reduces the ressources used for measurements.

References

[1] M. Amoretti, F. Zanichelli, and G. Conte. Sp2a: a service-oriented framework for p2p-based grids. In *Proc. 3rd international workshop on Middleware for grid computing (MGC '05)*, pages 1–6, New York, NY, USA, 2005. ACM.

[2] J. Case, R. Mundy, D. Partain, and B. Stewart. Introduction and applicability statements for Internet-Standard management framework. RFC 3410, Internet Engineering Task Force, Dec. 2002.

[3] D. Croce, M. Mellia, and E. Leonardi. The quest for bandwidth estimation techniques for large-scale distributed systems. In *ACM Hotmetrics 2009*, Seattle, jun 2009.

[4] D. Haage, R. Holz, H. Niedermayer, and P. Laskov. CLIO – a cross-layer information service for overlay network optimization. In *Kommunikation in Verteilten Systemen (KiVS) 2009*, Kassel, Germany, March 2009.

[5] C. Hübsch. Considering Network Heterogeneity in Global Application Layer Multicast Provision. In *EuroView*, July 2008.

[6] C. Hübsch, C. P. Mayer, S. Mies, R. Bless, O. P. Waldhorst, and M. Zitterbart. Reconnecting the internet with ariba: Self-organizing provisioning of end-to-end connectivity in heterogeneous networks. In *SIGCOMM 2009 Demos*, 2009.

[7] A. Tariq, B. Koldehofe, G. Koch, and K. Rothermel. Providing Probabilistic Latency Bounds for Dynamic Publish/Subscribe Systems. In *Proceedings of the 16th ITG/GI Conference on Kommunikation in Verteilten Systemen 2009 (KiVS 2009)*, Kassel, Germany, Januar 2009. Springer.

[8] O. Waldhorst, C. Blankenhorn, D. Haage, R. Holz, G. Koch, B. Koldehofe, F. Lampi, C. Mayer, and S. Mies. Spontaneous virtual networks: On the road towards the internet's next generation. *it – Information Technology Special Issue on Next Generation Internet*, 50(6):367–375, December 2008.