Planetopus: A System for Facilitating Collaborative Network Monitoring

Weichao Li, Waiting W. T. Fok, Edmond W. W. Chan,
Xiapu Luo, and Rocky K. C. Chang

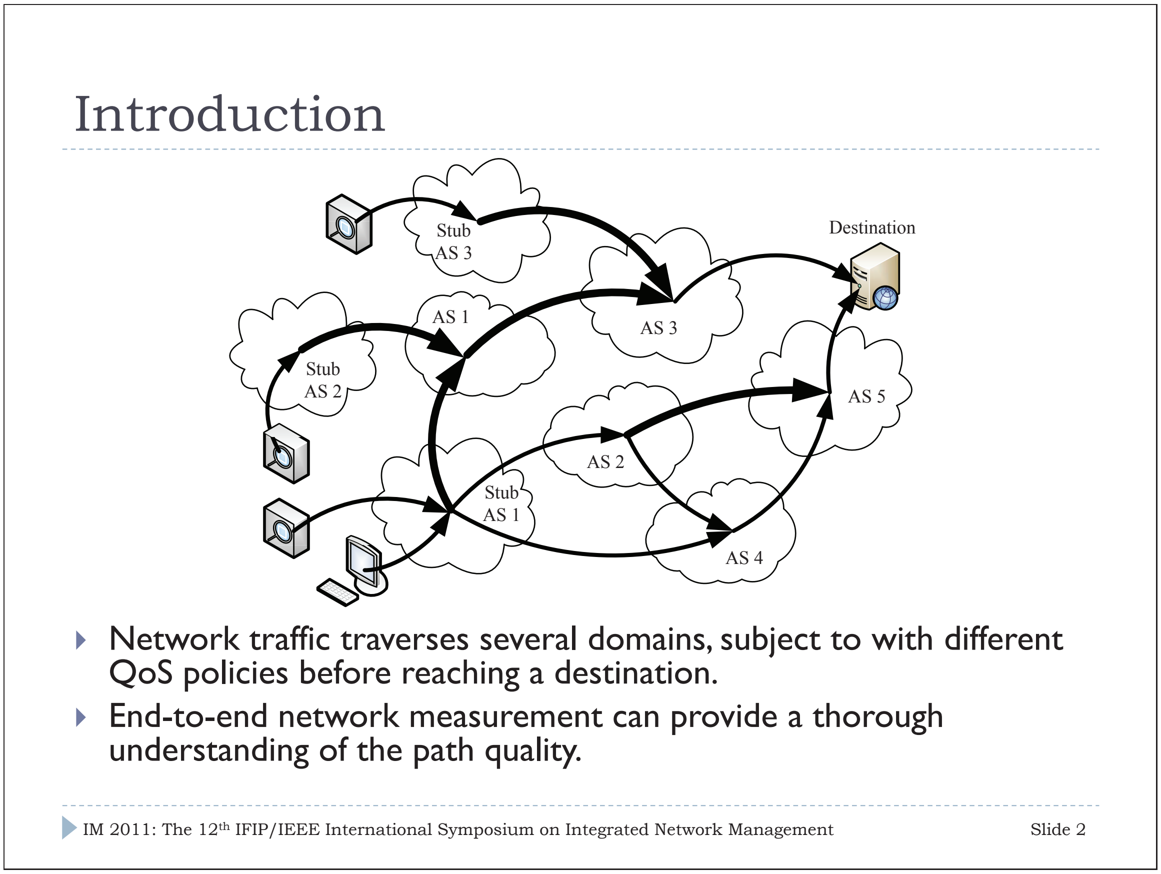
Department of Computing
The Hong Kong Polytechnic University
Hunghom, Kowloon, Hong Kong

Email: {csweicli|cswtfok|cswwchan|csxluo|csrchang}@comp.polyu.edu.hk

IM 2011: The 12th IFIP/IEEE International Symposium on Integrated Network Management

Abstract

Many new methods and tools have been developed to measure the quality of network paths for the last ten years. However, there are relatively few works that consider deploying these methods for collaborative network measurement: a number of measuring points belonging to different autonomous systems collaborate on monitoring and diagnosing their network performance. In this paper, we present Planetopus, a distributed system for facilitating collaborative network monitoring. Planetopus provides a single platform for configuring and scheduling measurement tasks performed on a set of distributed measuring points. Planetopus currently performs measurement mainly using OneProbe and tcptraceroute. Moreover, we introduce two useful facilities for analyzing the measurement data: a new metric for quantifying route changes and a heatmap-based visualization method for discovering patterns and anomalies from a set of path measurements. We demonstrate the utility of Planetopus through several case studies in which poor routes are identified and corrected, different ISPs' network services are compared, and network performance problems are diagnosed.



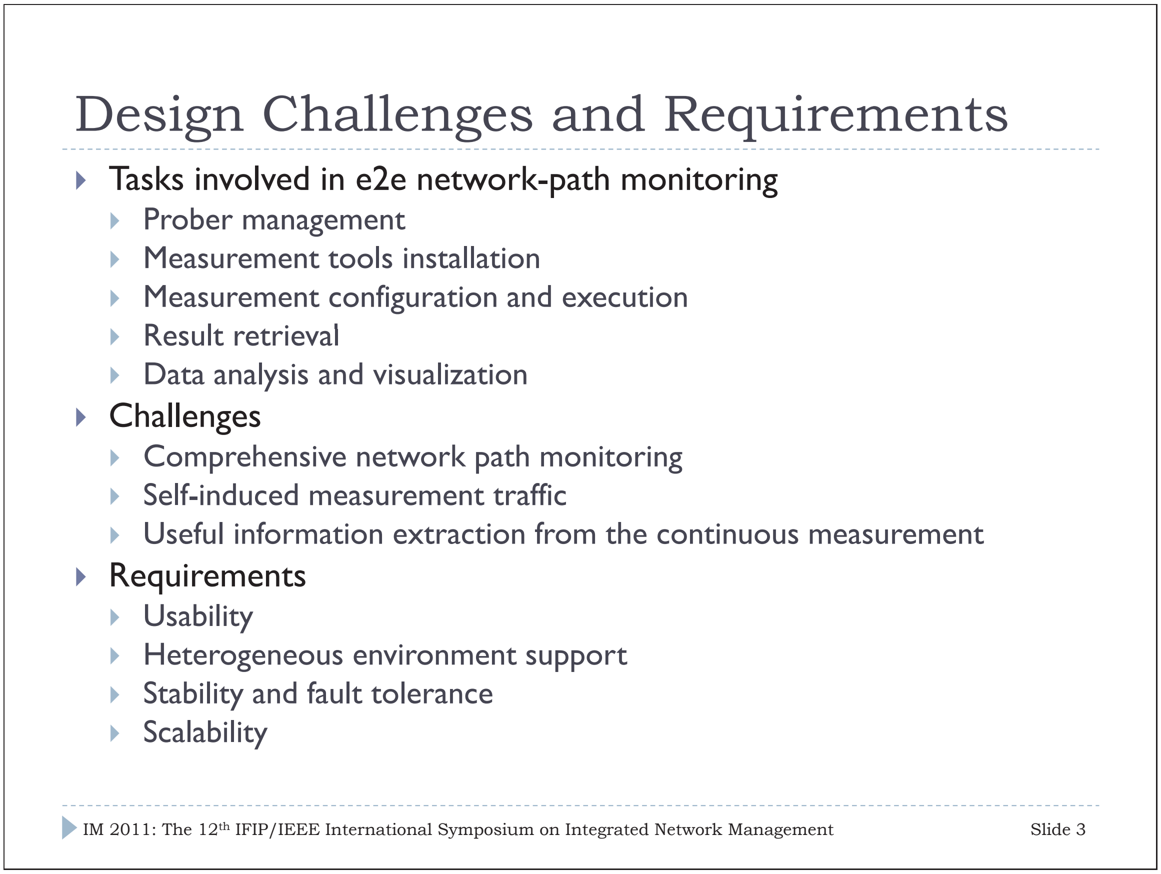
The continuous growth of the Internet increases the complexity in network topology and diversity in network path quality. Network traffic from a stub autonomous systems (AS) can traverse several domains under different QoS policies before reaching a destination. As a result, many monitoring platforms and measurement tools were developed for network administrators to manage and optimize their networks, as well as for researchers to study the network behavior and characteristics.

To gain a thorough understanding of the end-to-end (e2e) path quality, many new measurement tools have been introduced under the scope of active measurement. For example, OneProbe [23] can measure round trip time (RTT), asymmetric packet loss rate and reordering rate based on pairs of back-to-back TCP packets, while SProbe [25], DSLProbe [19] and Asymprobe [17] can measure the capacity of an e2e path. The IETF IPPM working group has also defined a number of performance metrics for active measurement.

Besides the methods for measuring path quality, there are other equally, if not more, important issues for collaborative network monitoring. Such issues concerns different steps in the entire cycle of network monitoring, including the deployment of measurement tasks, management of the measurement processes, analysis of measurement data, and effective communication of the results to users. The entire cycle must continue despite various kinds of hardware and software failures.

In this paper, we present *Planetopus*, a flexible and scalable distributed system for active network measurement. Planetopus can easily deploy third-party measurement tools to monitor network path performance with low configuration effort. Planetopus can effectively correlate and analyze measurement data obtained by a set of collaborating vantage points from different ASes. To augment the correlation and facilitate the collaboration, we introduce several analysis and visualization techniques for time series of path-quality data and route fluctuations.

We have deployed Planetopus to perform long-term continuous Internet path monitoring for more than two years. Our experience shows that Planetopus is able to help diagnose various network performance problems and to discover their root causes. Moreover, Planetopus can help academic researchers easily deploy and evaluate their tools on a real network environment.



Active monitoring of e2e network path quality involves a number of tasks, including source node and prober management, monitoring tool deployment, measurement configuration and execution, measurement monitoring, measurement data analysis, and measurement result visualization.

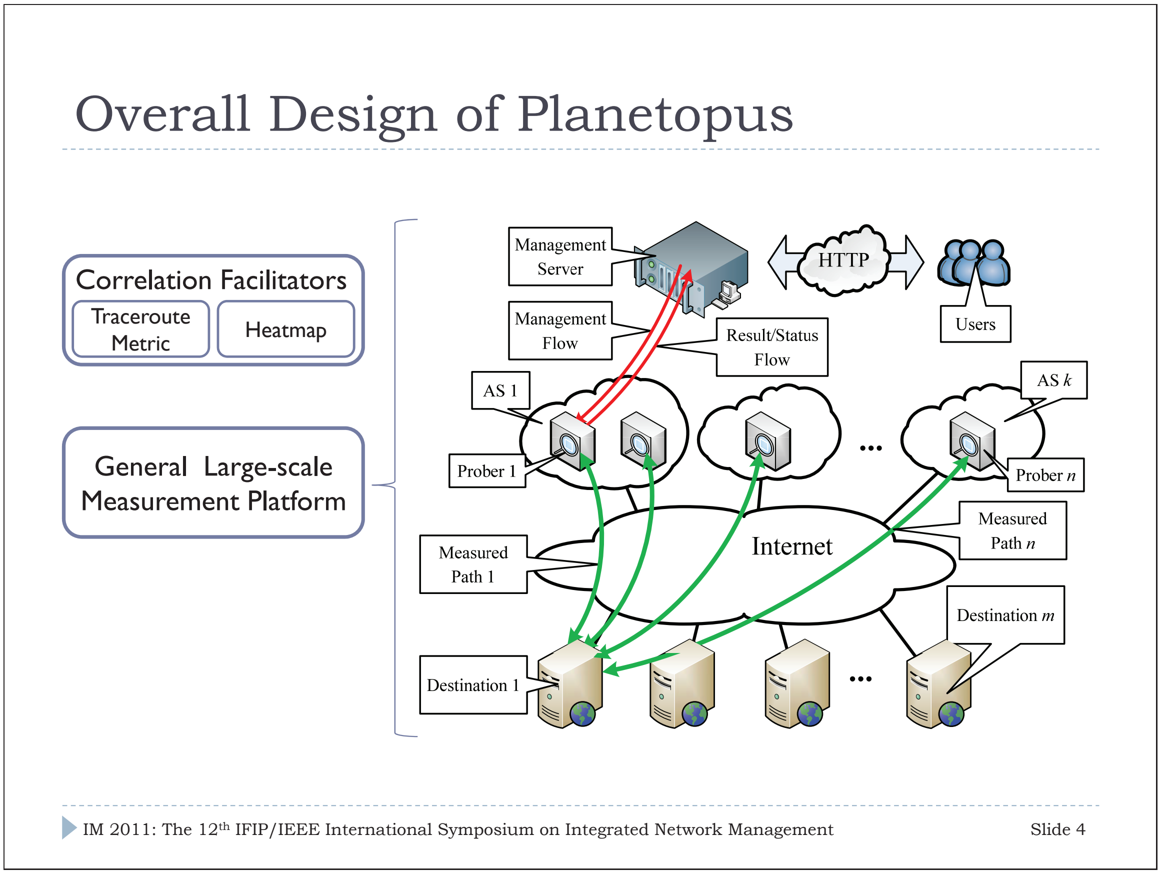
There are several challenges to active network-path monitoring. One of them is to perform comprehensive network path monitoring. Since a single prober can monitor only a small subset of Internet paths due to limited resources, we need several probers, which can be located at different places, to carry out the measurement simultaneously. However, this approach can increase both cost and complexity of the path management.

Another challenge is that measurement traffic can affect legitimate traffic in the network path under monitoring. Excessive measurement traffic can be treated as abnormal and therefore discarded. To avoid congesting a destination, we should control the number of probers simultaneously probing the same destination.

In addition, it is not an easy task to extract useful information from the continuous measurement. Therefore, providing an efficient mechanism to analyze and visualize the results is necessary. Although such mechanism can reduce the cost of both management and monitoring, few existing works consider this aspect carefully.

Taking all the challenges into consideration, we propose a number of requirements for the platform design:

- **Usability** The platform must be user-friendly. A well-designed GUI will enhance user experience and lower the configuration effort. Also, a well-designed data processing and visualization mechanism can increase the efficiency of result analysis and event detection.
- **Heterogeneous environment support** The platform must support complex network environment, different types of probers, and various monitoring tools.
- **Stability and fault tolerance** The probers must be able to run the measurement automatically after deployment. Failure of centralized server will not interrupt the measurements deployed to the probers. Likewise, failure of probers will not affect the server and other nodes. When the prober recovers from the failure, the ongoing measurement will resume automatically.
- **Scalability** The platform must scale to support hundreds or thousands of nodes and can measure as many paths as possible.



Instead of expanding the measurement scale in terms of number of probers and monitored network paths, we introduce a collaborative monitoring approach that coordinates a set of probers located in several neighboring stub ASes. Each participating AS conducts measurement with the same set of remote endpoints. The collation of measurement results can help identify and correct poor routes, compare different providers' network services, and diagnose network performance problems.

We design Planetopus to meet the requirements stated in the previous slide. As shown in the diagram above, Planetopus comprises two fundamental parts: a general platform for collaborative network measurement and correlation facilitators for result analysis and visualization. There are *n* probers and *m* destinations, therefore $n \times m$ network paths for the measurement.

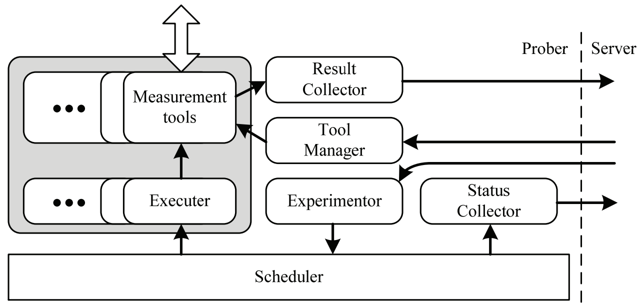
Probers are located at specific points in different ASes depending on the network topology and measurement goals. Planetopus facilitates large-scale measurement by supporting probers implemented in virtual machines. For example, PlanetLab [5] currently consists of 1094 nodes at over 500 different places[1]. Once a prober is registered, it will report its system utilization to the management server every several minutes. When the measurement begins, the prober will launch the measurement tool to inject probe packets to a destination and send the measurement data to the management server.

Destinations are the remote nodes of the e2e network paths. They could be pre-configured machines for cooperative measurement, or existing network points that fulfill the requirements of non-cooperative measurement tools. For example, the One-Way Active Measurement Protocol (OWAMP) [26] requires setting up a receiver before measurement execution, while OneProbe can treat a non-cooperative web server as a destination.

Management server manages probers, maintains and installs measurement tools, and allows users to configure and deploy measurement. After launching the measurement, the server will collect measurement data and store them into a database. It also provides a web-based GUI for users to configure measurement settings and visualize measurement results for different network performance metrics.

[1]Reported by the PlanetLab official site by Aug 24, 2010.

Prober Architecture



- Light-weight and widely supported by different operating systems
  - Realized by script languages (Perl, Python)
  - Network Time Protocol (NTP) for time synchronization
  - Utilize *cron* as the scheduler
- The measurement execution can be delayed by a random period of time within a pre-defined period to avoid flooding the same destination.
- Support testbed nodes, such as PlanetLab nodes

The prober architecture is depicted in the figure above. We use common script languages (such as Perl and Python) to implement each component, so that the prober can be widely supported by different operating systems, including PlanetLab nodes. The prober also regularly performs time synchronization via Network Time Protocol (NTP).

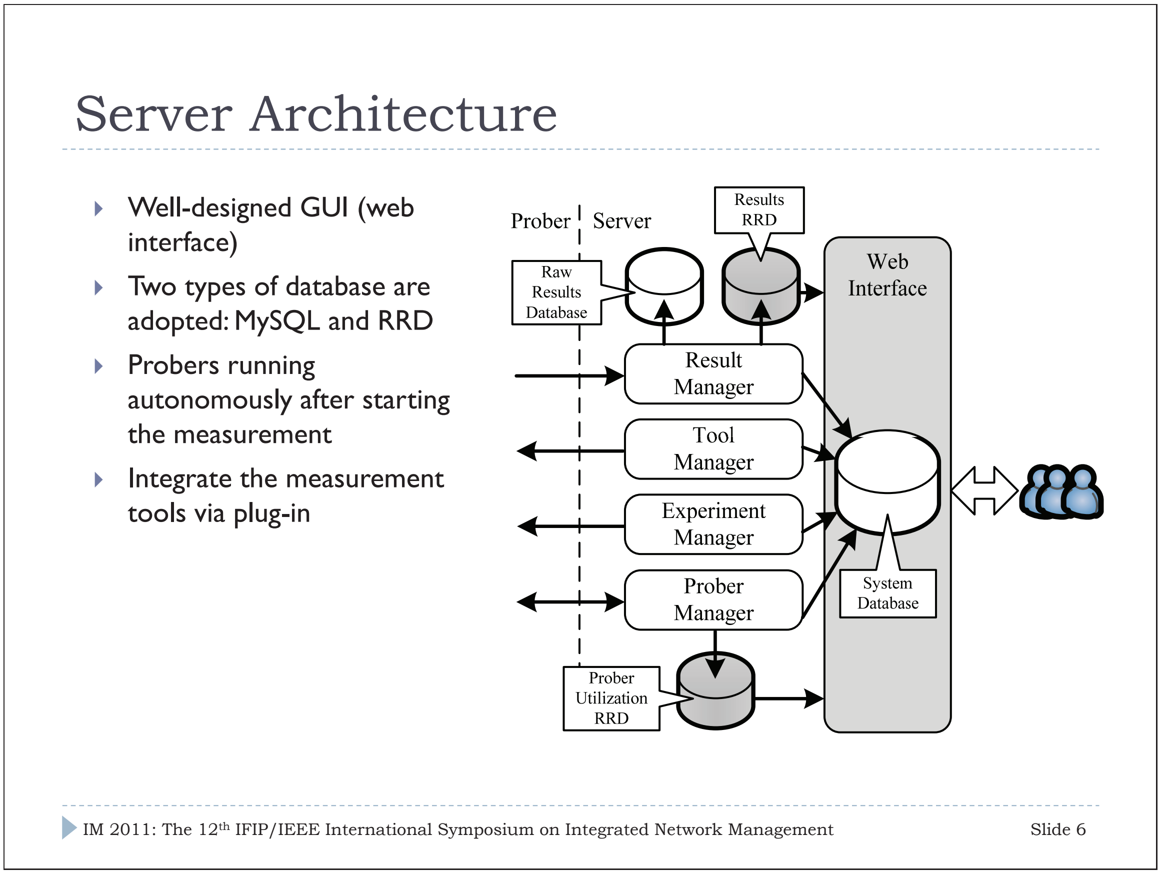
Status collector collects system utilization of the prober (such as CPU, memory, and hard disk) every five minutes and sends the information to the management server via HTTP. *Tool manager* maintains the measurement tools and accepts the instructions from the management server to perform various actions (e.g., downloading and installing measurement tools).

Experimentor accepts the instruction from the server to execute a network measurement project. When it receives a command that requires the prober to start a network measurement project, it will download a configuration file from the server with the following parameters: start time, end time, monitoring tools and their required parameters, storage location, transmission mode for the results, and measurement schedule. The experimentor then parses the file and sets up the required environment for the project, including the measurement tool validation and storage space preparation. After finishing all these tasks, *executer* is then registered to the *scheduler*.

We utilize *cron* as the scheduler and register the executer in the cron job list. Therefore, the executer will be launched by the cron process every minute. The executer controls the start time for a measurement and launches the measurement tools according to the configuration file. To avoid flooding the same destination by a set of probers, the executer delays the measurement execution by a random period of time within a pre-defined value. Moreover, the executer can resume the project after recovering from the failure, so that the prober can work alone after deployment. When the end time of the project has been reached, the executer removes itself from the cron job list.

Result collector maintains the measurement data collected by the tools and transfers them to the management server for further processing. The prober does not process the data to avoid generating extra system load that can affect the network measurement.

Security is another important issue that needs to be taken into consideration. In Planetopus, we use SSH to protect the data transfer between probers and the management server (except the system utilization information). The management server distributes the public RSA key to the probers and uses the private key to encrypt the traffics. Moreover, a compromised prober should not affect the other probers and the server.

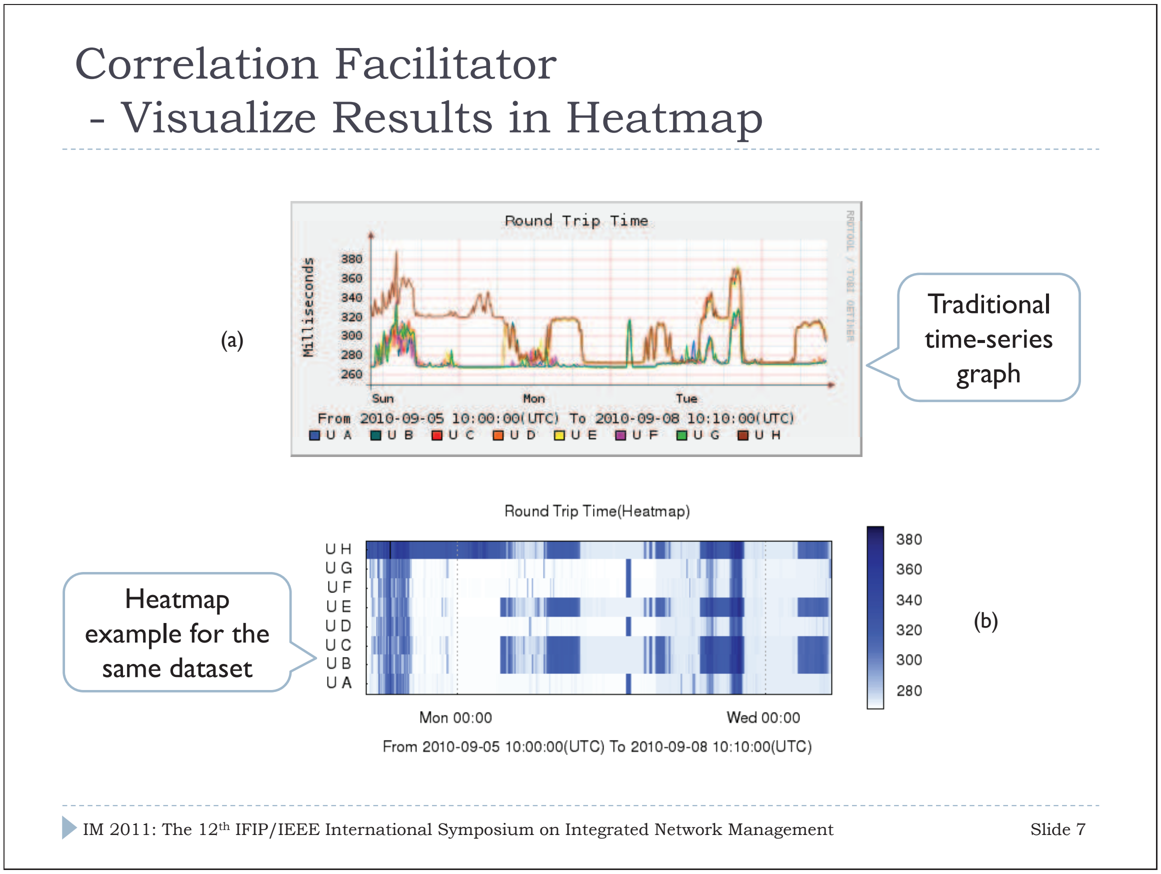


We have implemented a web interface to handle the interaction between users and Planetopus. The interface accepts various user requests for creating network monitoring project, choosing probers and measurement tools, and viewing measurement results and prober status. The user requests are then parsed and delivered to the exact components. We have also implemented access control to the web interface by categorizing users into several groups with different privileges.

Two types of database are adopted: MySQL and Round-robin database (RRD). The user information and the settings of each component are stored in the *system database* using MySQL. The long-term time-series data, like the prober status collected by the *prober manager* and the measurement results collected by the *result manager*, are saved into the RRD files, which are well suited for time-series data storage. Its storage fraction will be reused and old data will be aggregated with a lower time resolution. Consequently, the size of the dataset will remain constant and the data can be fetched quickly. We have integrated RRDTool [6] to handle the RRD and visualize measurement result. The raw data of the measurement results are also stored simultaneously for further analysis.

After the *experiment manager* informs the prober to download the configuration files, the prober will keep running the measurement without relying on the management server. This design can lower the server load as the number of probers increases. Moreover, the failure of the server or some probers will not affect the measurement at other probers.

All measurement tools are maintained by a *tool manager* as plug-ins. We prepare a set of interfaces to let users define new measurement tools. A configuration page for a measurement tool is generated based on user definitions. After a measurement tool is selected for the measurement project, the required parameters will be transferred to the page to form a complete execution command. The executable binaries of the measurement tools for different OSes can also be kept in the server. The tool manager will deploy the required binary and execution command based on the OS version of the prober.

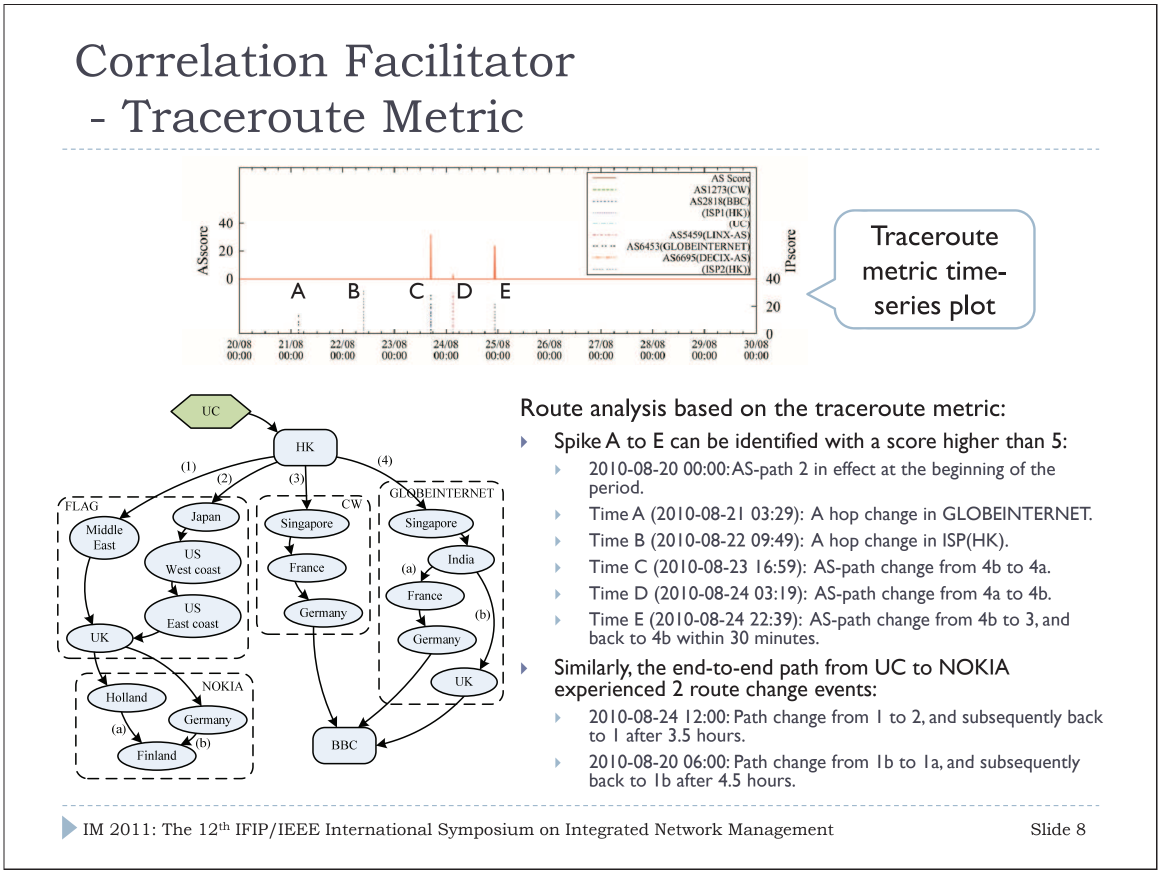


A good visualization methodology can provide sufficient information for users to understand the current network status. For example, time-series graphs, which are used in most existing monitoring platforms, depict the history and trend of the monitored targets. We can also observe diurnal patterns, and special events (such as sudden peak or decline) from the graphs.

Time-series graphs can also be used to compare different datasets by denoting them with different colors. Figure (a) in the above slide shows an example of time-series graph consisting of multiple curves for three days' RTT history obtained from eight correlated network paths. Though we have adopted various colors for each curve, it may not be easy to identify which curve represents the exact path when some of the curves are jumbled together.

As a result, we resort to heatmap, a graphical representation of data where the values are displayed with gradient color in a two-dimensional map. In particular, darker squares denote larger values while lighter ones denote smaller values. Different datasets are visualized in different rows. Figure (b) is the heatmap view for the same datasets in the previous example. We can easily distinguish each path and point out their correlation quickly. For example, the paths from probers UB, UC, and UE share the similar performance while UA, UD, UF, and UG can be considered as another group. In addition, the darker color of UH shows that this path suffered from network congestion during the early period. It is not easy to obtain such information based on the time-series graph.

We have implemented a heatmap plotting library based on gnuplot [1]. The library fetches data from RRD files and generates a heatmap graph dynamically. We integrate the library into Planetopus as a facilitator and allow users to switch between these two visualization methods. The facilitator can effectively discover the correlation of measurement results.



We augment the network monitoring by using the routing information obtained by a set of probers. Each prober performs tcptraceroute [29] to measure the forward route for a specific destination. We analyze the traceroute data to identify any route change event and correlate the data obtained by different probers to infer the root cause of network problems. For example, we visualize the traceroute results for two European destinations NOKIA and BBC from a prober UC during a 10-day period in Aug 2010. We resolve the geographical location of each intermediate router by analyzing the DNS hostname of each hop [27]. We also group all the hops with the same geographical location together. For instance, the HK→Middle East→UK paths in FLAG consists of 4 logical paths or 15 IP addresses, in which 8 IPs are mapped to Alexandria of Egypt and 4 IPs to London. Therefore, such visualization provides a high-level view on how the traffic is routed through each autonomous network. However, similar to existing tools like [18] and TraceViz [11], such visualization method may produce a large diagram for a long-term traceroute data, thus reducing its readability. Moreover, processing the traceroute data is a time-consuming job, because we have to resolve the per-hop IP address to the ISPs or ASes.

We therefore introduce a scoring scheme to distinguish route change event from load balancing or other traceroute anomalies, e.g. unresponsive hops. It is applied to periodical traceroute measurement for any e2e path. In the example, we see two types of route change:

- Traffic being forwarded to different ASes, and
- Traffic traverse different physical paths within the same AS.

Our scheme first covers the AS-path, with each AS as a node on the path. On the intra-AS level, each hop is regarded as a node on the path. Since each AS may have their own load balancing architecture and setting, it is considered separately.

We illustrate the scheme with an AS-path. Let $A_1, A_2, ..., A_n$ be the autonomous systems (AS) that a forward path passes through at t_0, and $t_l(A_x)$ be the timestamp that A_x last exists in traceroute up to t_{-1}. We define timing difference, $t_d(A_x) = t_0 - t_l(A_x)$. Assume traceroute for each e2e path are scheduled at fixed interval, say 600 seconds. For A_x that exists in the route at both time t_{-1} and t_0, $t_l(A_x)$ = t_{-1}. Therefore, $t_d(A_x) = t_0 - t_{-1} = 600$. In other cases, if A_x does not exist in recent traceroute results, we will observe a higher values $t_d(A_x)$. The longer time that A_x was not found from previous result, the higher the value.

For each run of an e2e path, the total time difference, $\sum_{x=1}^{n} A_x$, are normalized with historical values inside a fixed sliding windows to give a standard score. We argue that when route change occurs, nodes that do not exist on e2e path for a long period of time or are even never found appear on the e2e path. Path changes due to load balancing are short-lived, and the noises created are filtered during the normalization. Effects of hops that do not respond to all traceroute probes are also eliminated.

Case Study

- We use Planetopus to manage long-term collaborative monitoring since 1 January 2009.
- We deployed eight probers to measure 43 web servers around the world.
- Planetopus collects around 6-GB data daily.

Home » Round Trip Time   Choose Other Metrics: Round Trip Time

Name	URL	U B	U F	U C	U A	U H	U E	U D	U G
HKIX(HK)									
mingpao	www.mingpaonews.com	2.4	1.5	2.6	2.8	3.1	2	2.5	2
atnext	www.atnext.com	3.7	2.9	3.9	3.7	4.4	3.2	4.9	2.9
pccw	www.pccw.com	3.8	3	4	4.3	4.6	3.4	4.5	3.4
wifijucc	wifi.jucc.edu.hk	1.3	1.3	2.1	4.1	3.2	1.3	2.9	1.6
HKIX(ASGCNET)									
twgrid	www.twgrid.org	22.7	23.1	24	24.4	24.6	23.5	23.9	23.4
HKIX(KREONET)									
ktc	ktc.gist.ac.kr	38.6	59.7	40	40.3	40.6	39.5	39.9	39.4
kreonet	www.kreonet.net	35.3	35.8	36.8	37	37.4	36.2	36.6	36.2
Internet(China)									
taobao	www.taobao.com	34.8	33.5	35.4	34.8	186.3	34.5	35.2	35.1
lenovo	appserver.lenovo.com.cn	52.7	54.2	50.9	57.4	418.4	50.3	53.5	53.8
Internet(England)									
eng2	www.itraveluk.co.uk	241.7	243.6	242.5	233.5	258.8	241.7	242.4	244.2
eng4	www.oldmap.co.uk	216.5	217.5	218	272.3	272	217.6	219	217.8
eng3	www.maps-of-britain.co.uk	222.1	220.1	222.7	276	313.8	222.4	222.6	222
bbc	www.bbc.co.uk	217.5	221.2	217.5	271.4	308.7	221	221.7	221

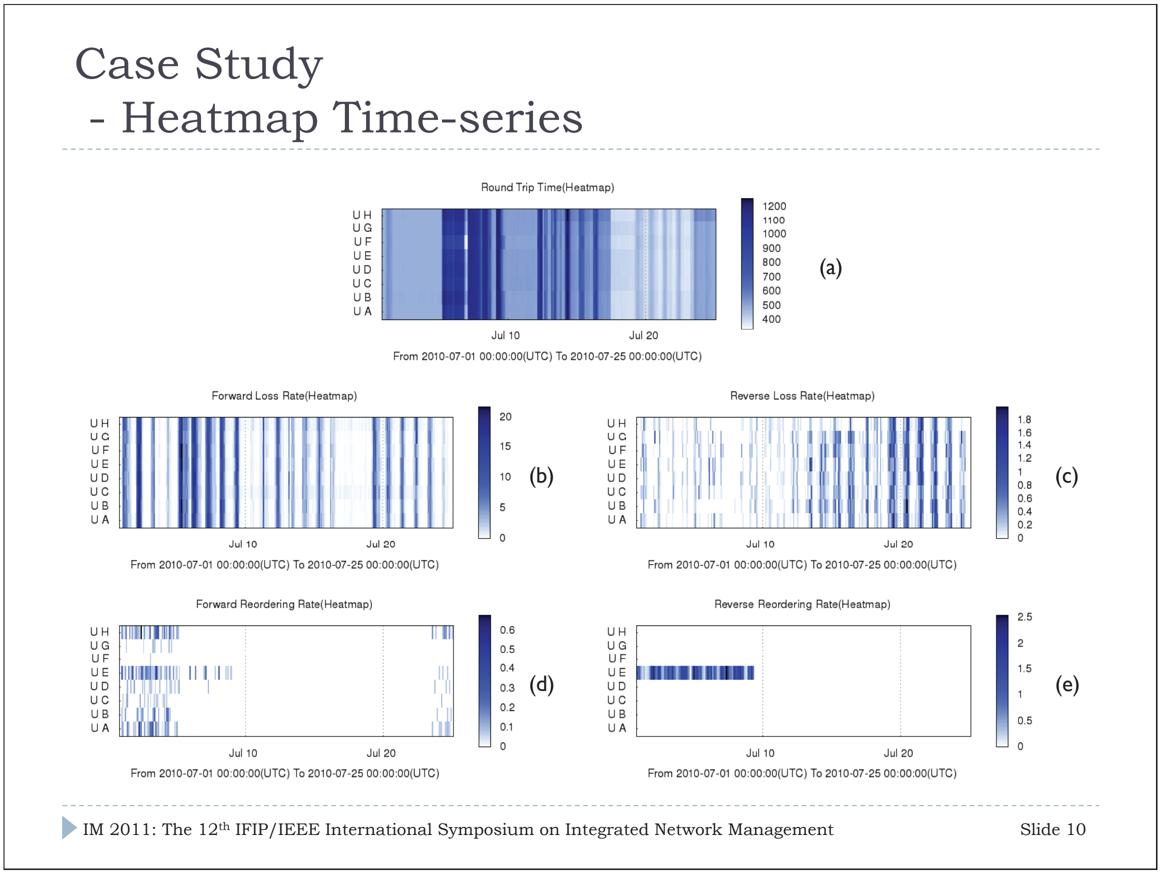
We use Planetopus to manage a long-term collaborative network path measurement from HARNET which connects the campus networks of eight universities in Hong Kong. In this measurement, we deploy a prober at each university and measure e2e paths from each prober to 43 web servers that are located in Hong Kong, Europe, US, Australia, China, Japan, Korea, and Taiwan. The measurement has been conducted smoothly since 1 January 2009. Planetopus collects and processes around 6-GB data daily and shows the result in a user-friendly interface.

The figure in the above slide depicts the comparison of a measured performance metric. The values are displayed in real-time, and grouped by probers and destinations. The green color in a cell means that the corresponding path has a better performance than other paths to the same target; otherwise, whereas the red cells are those that perform worse than other paths. Therefore, users can obtain an overview about the performance of these network paths.

We employ OneProbe to measure each network path because of its accurate, reliable, and metric-rich features [23]. OneProbe obtains several path metrics including RTT, asymmetric packet loss rate, asymmetric packet reordering rate, and capacity. We manage various OneProbe's parameters through Planetopus: sampling frequency of 2Hz, probe/response packet size of 536 bytes, and measurement duration of one minute. As a result, the total measurement traffic between the eight probers and a destination is less than 80 Kbits/s. As discussed in the previous slide, Planetopus also invokes tcptraceroute to get the routing information of each path.

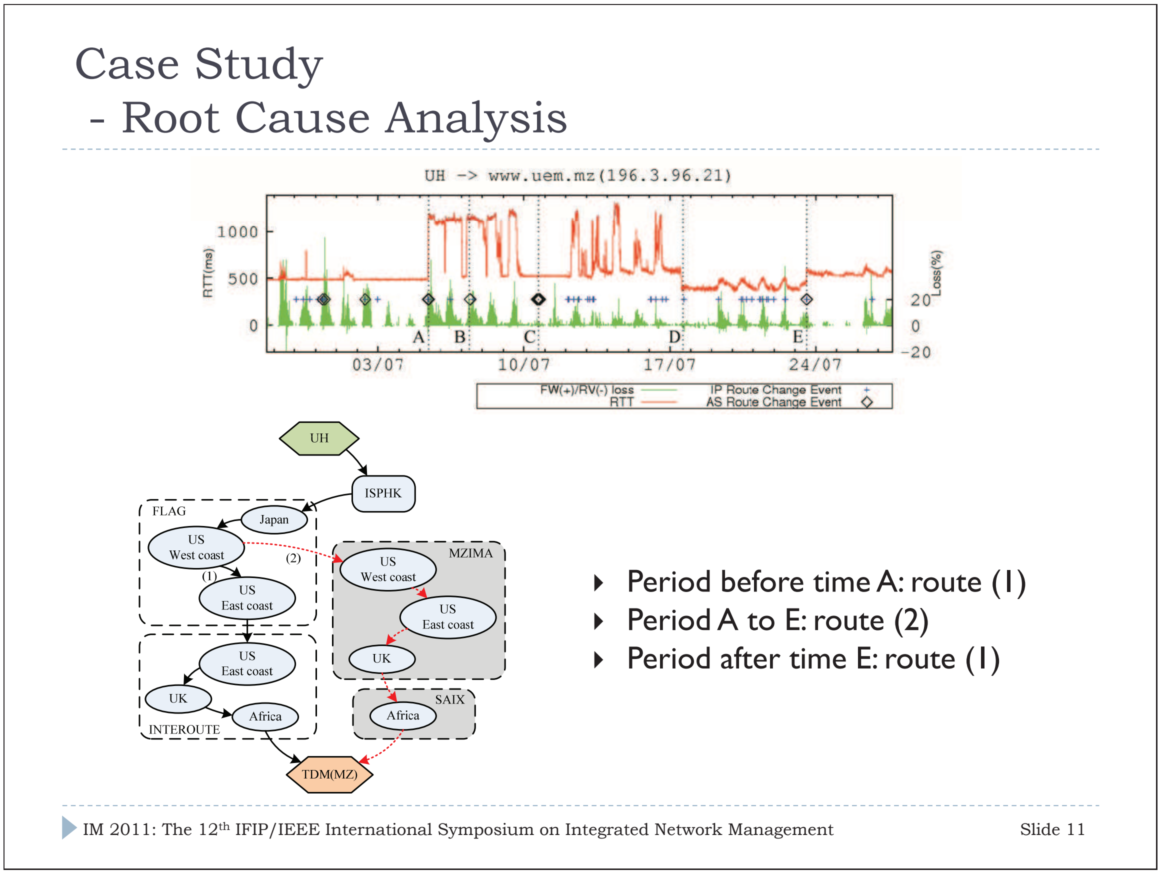
To avoid self-induced network congestion, we design our measurement by dividing the remote servers into five groups, each of which consists of eight or nine instances. All probers measure the paths to a group of servers for ten minutes. Then, Planetopus transfers measurement data obtained by each prober into the database and RRD and finally launches the measurement with another group of servers. Planetopus visualizes the quality of each path in the form of time-series diagram and heatmap through the web interface.

With the help of Planetopus, we can easily manage the collaborative network monitoring involving the probers located in neighboring ASes. We have shown in [16] that such collaboration allows users to identify poor routes, compare different ISPs' network services, and diagnose network performance problems. In the following, we use a case study to demonstrate how Planetopus captures abnormal network events during the period of 2010 FIFA World Cup South Africa and identifies the root cause.



We observe severe network congestions on the paths from the eight probers, UA, . . . , UH, to a web server (referred to as UEMMZ) in Mozambique during July 2010. Figures (a), (b), and (c) show the heatmaps to illustrate the RTT (in milliseconds), forward-path and reverse-path packet loss rate (in %), forward-path and reverse-path packet reordering rate (in %), respectively, during the period from 01 July 2010 to 25 July 2010. According to the heatmaps, the eight paths have similar diurnal patterns in RTT and packet loss rate.

While the eight paths have similar patterns in RTT and packet loss rate, they have different packet reordering patterns as shown in figures (d) and (e). In particular, the reverse-path packet reordering rate for the UE path is much higher than that for the other paths. This path also experienced longer packet reordering periods than the other paths. Since the UE's route to UEMMZ was the same as others, the packet reordering may occur within UE's AS. We also observed similar phenomenon from the other UE paths to different destinations. Another interesting observation is that the high reverse-path packet reordering for all the UE paths disappeared since July 9. We therefore believe that UE had changed its network configuration to resolve the issue.



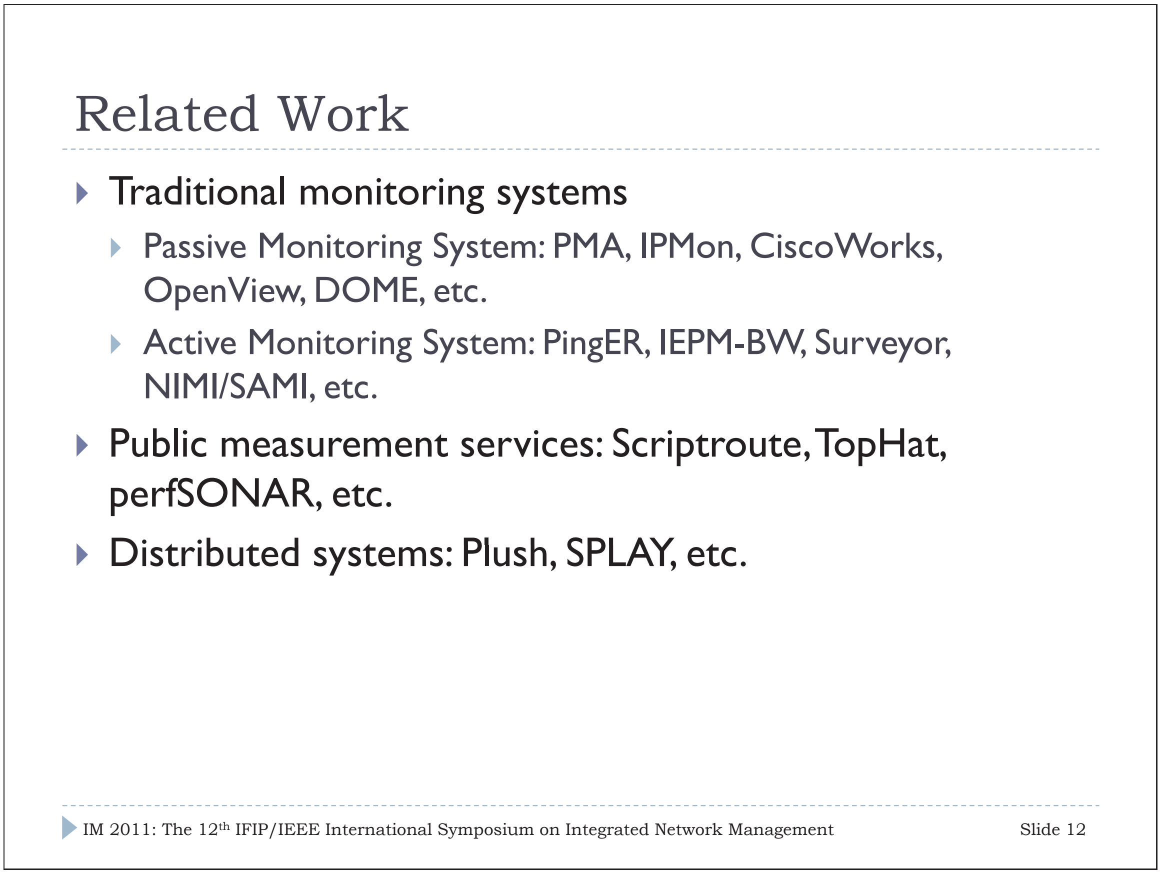
To infer the root cause of large RTT and high loss rate, we fetched the raw data from the database and performed an in-depth analysis. We calculate the traceroute metrics for this path both in IP level and AS level. Since the eight paths show similar performance in terms of RTT and loss rate, we only show the results for the path from UH to UEMMZ. Moreover, we only keep scores higher than five to filter out insignificant route change events. The top figure in the above slide shows the time-series of RTTs, loss rates, and the traceroute events between 28 June 2010 and 28 July 2010, where significant events are labeled as A-E.

Notice that the RTTs increased from 500 ms to more than 1000 ms at around July 5, 9:30 (event A). The surge lasted for around two weeks with an obvious diurnal pattern. Between July 17 (event D) and July 23 (event E), the RTTs dropped to 400 ms, but the diurnal patterns still existed. After event E, the RTTs resumed to 500 ms and the diurnal pattern disappeared.

During the period between events A and E, there were four remarkable AS-level route change events. At the first glance, the timings of the events A and E match quite well with the sharp changes of the network performance. To reveal the correlation between the performance and route, we further analyze the traceroute data, which is plotted in the bottom figure, before and after the events.

The network path traversed several ASes before reaching the destination in Africa. As shown in path (2) in the left hand side of the bottom figure, the path routed through Japan, US, and UK before reaching Africa involving two ASes—FLAG and INTEROUTE—before event A. However, the path did not go through INTEROUTE after event A, but through MZIMA and SAIX as shown in path (2) in the right hand side of the figure. After event E, the path restored to path (1). We also look into the routes at events B and C when two route change events appeared, and our analysis confirms that such events were caused by the non-responsive intermediate hops.

The traceroute analysis shows that the route change resulted in poor performance in that period. Our subsequent investigation discovers that a SEACOM cable experienced a failure in the north of Mombasa offshore at July 5, 09:19 GMT [10]. As a SEACOM partner, INTEROUTE interconnects East Africa with Europe by using the service of SEACOM [9]. Therefore, the traffic was redirected to SAIX which relied on two other submarine cables for intercontinental Internet connections [7], and the excessive traffic congested the network path. The route was finally restored to the original one; therefore, the observed performance returned to the level before the cable outage.



To better understand the evolving Internet, many network measurement systems have been built to facilitate users to carry out large-scale measurement. Some systems, such as NLANR's PMA (Passive Measurement and Analysis)[2] and Sprint IPMon [20], focus on passive measurement that collects data from many sources without injecting probing packets into the network. Many commercial tools, including those provided by network equipment suppliers (e.g., CiscoWorks by Cisco System, Inc.) and those provided by the third party (e.g., HP OpenView[3]), usually just provide support for passive measurement. Besides obtaining the information through protocols like SNMP, RMON, and IPFIX, such systems can deploy packet capture devices to some points and perform distributed passive monitoring. In [14], a distributed passive measurement infrastructure is presented. DOME [30] is another platform that can perform distributed real-time passive network measurement, while FLAME [13] provides an architecture for efficient programmable packet-level network monitoring.

Some measurement systems are designed to facilitate active measurement that sends probing packets to infer network performance. For example, IEPM group led PingER project [4] to monitor the e2e performance of the Internet links with ICMP ping, and IEPM-BW project [2] to measure the bandwidth. Surveyor [21] provides a measurement infrastructure that measures e2e unidirectional delay, packet loss, and route information along Internet paths. These projects are not flexible, because they were designed for some particular tools or techniques. As a result, it is difficult to extend and empower them with new capabilities.

NIMI [24] establishes a scalable infrastructure for large-scale e2e network behavior monitoring. Its successor, SAMI [8], further introduces the user authorization mechanism and provides a more secure access. These projects have defined an open environment for integrating new measurement tools, but the strict criteria limit the usage scenario. Moreover, the deployment cost and the performance issue also make them hard to deploy widely.

Several infrastructures provide public measurement services. For example, Scriptroute [28] deploys a set of tools on PlanetLab [5] nodes. TopHat [15] provides a topology monitoring service for the PlanetLab testbed and can aggregate measurements from other federated infrastructures. PerfSONAR [3] is another service-oriented infrastructure, which aims for troubleshooting e2e performance crossing multi-domain. Though these infrastructures extend the scope by drawing upon the existing large-scale network services, it is difficult to deploy measurement nodes at any point on demand.

Some distributed systems, such as Plush [12] and SPLAY [22], can deploy and manage large-scale distributed applications. However, such systems are not appropriate for network monitoring, because they lack both mechanism for scheduling periodic measurement and integration of result analysis and visualization. Motivated by the limitations of existing works, we design and build Planetopus. Our goal is making it a general infrastructure with the features of high flexibility, extensibility, scalability, and lightweight.

[2]The NLANR project had been officially ended and taken over by CAUDA since July 2006.

[3]In 2007, HP OpenView was rebranded and the division was renamed HP Software & Solutions in 2008.

Conclusion and Future Work

- Planetopus is a flexible and scalable network monitoring infrastructure
  - Aim at large-scale e2e network monitoring
  - A set of collaborative measurement points located in different ASes
  - Support the entire cycle of measurement: from design to result analysis and visualization
- Further research
  - Deploy more probers to the Internet and expand the measurement scale.
  - Introduce overlay network architecture to improve scalability and fault tolerance.
  - Study result accuracy when execution delay scheme is applied.
  - Investigate path selection mechanism to improve accuracy.

Planetopus is a flexible and scalable infrastructure for large-scale e2e network monitoring. Compared with the existing works, Planetopus solves the limitations of active measurement by collaborating a set of measurement points located in different ASes and covers the whole chain from measurement design and management to result analysis and visualization. We design and implement Planetopus as a general infrastructure with high flexibility, extensibility, scalability, and light weight. Users can easily utilize measurement tools to perform a long-term e2e network path monitoring with low configuration effort. To facilitate the study and analysis of the monitoring results, we integrate several novel visualization methods: traceroute metric for observing route fluctuation and heatmap for displaying the time-series data. We also discuss our experience on using Planetopus to infer the root cause from network events.

In future work, we will focus on large-scale e2e network path measurement. We are planning to deploy more probers in the Internet to expand the measurement scale. The architecture of overlay network may be introduced to our system to further improve the scalability and fault tolerance. To overcome some shortcomings of active measurement, we will study the result accuracy when applying some delay schemes for the measurement execution. We will also study the selection mechanism for the measurement paths to improve the measurement accuracy.

References

[1] gnuplot. http://www.gnuplot.info/.
[2] IEPM-BW. http://www-iepm.slac.stanford.edu/bw/.
[3] perfSONAR. http://www.perfsonar.net/.
[4] PingER. http://www-iepm.slac.stanford.edu/pinger/.
[5] PlanetLab. http://www.planet-lab.org/.
[6] RRDTool. http://oss.oetiker.ch/rrdtool/.
[7] SAIX network topology. http://saix.net/cgi-bin/saix_topology.pl.
[8] SAMI. http://www.psc.edu/networking/projects/sami/.
[9] SEACOM partners with Interoute to open a digital super highway between East Africa and Europe. http://www.interoute.com/news-events/news/seacom-partners-interoute-open-digital-super-highway-between-east-africa-and-europe.
[10] SEACOM service down; SEACOM actively seeking options to restore service - 5 July 2010. http://www.seacom.mu/news/news_details.asp?iID=143.
[11] TraceViz. http://www.hokstad.com/traceviz-visualizing-traceroute-output-with-graphivz.html.
[12] J. Albrecht, C. Tuttle, A. C. Snoeren, and A. Vahdat. PlanetLab application management using Plush. *SIGOPS Oper. Syst. Rev.*, 40(1):33–40, 2006.
[13] K. Anagnostakis, M. Greenwald, S. Ioannidis, D. Li, and J. Smith. Flexible network monitoring with FLAME. *Computer Networks*, 50(14):2548 – 2563, 2006.
[14] P. Arlos, M. Fiedler, and A. A. Nilsson. A distributed passive measurement infrastructure. In *Proc. PAM*, 2005.
[15] T. Bourgeau, J. Aug, and T. Friedman. TopHat: supporting experiments through measurement infrastructure federation. In *Proc. TridentCom*, 2010.
[16] R. Chang, W. Fok, W. Li, E. Chan, and X. Luo. Neighbor-cooperative measurement of network path quality. In *Proc. IEEE GLOBECOM*, 2010.
[17] L. Chen, T. Sun, G. Yang, M. Sanadidi, and M. Gerla. End-to-end asymmetric link capacity estimation. In *Proc. IFIP Networking*, 2005.
[18] B. Cheswick, H. Burch, and S. Branigan. Mapping and visualizing the Internet. In *Proc. USENIX Annual Tech. Conf.*, 2000.
[19] D. Croce, T. En-Najjary, G. Urvoy-Keller, and E. Biersack. Capacity estimation of ADSL links. In *Proc. ACM CoNEXT*, 2008.
[20] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot. Packet-level traffic measurement from the Sprint IP backbone. *IEEE Network*, 17(6):6–16, Nov. 2003.
[21] S. Kalidindi and M. J. Zekauskas. Surveyor: an infrastructure for Internet performance measurements. In *Proc. INET*, 1999.
[22] L. Leonini, E. Rivière, and P. Felber. SPLAY: distributed systems evaluation made simple (or how to turn ideas into live systems in a breeze). In *Proc. NSDI*, 2009.
[23] X. Luo, E. Chan, and R. Chang. Design and implementation of TCP data probes for reliable network path monitoring. In *Proc. USENIX Annual Tech. Conf.*, 2009.
[24] V. Paxson, J. Mahdavi, A. Adams, and M. Mathis. An architecture for large scale Internet measurement. *IEEE Communications Magazine*, 36(8):48 –54, Aug. 1998.
[25] S. Saroiu, P. Gummadi, and S. Gribble. SProbe: a fast technique for measuring bottleneck bandwidth in uncooperative environments. In *Proc. IEEE INFOCOM*, 2002.
[26] S. Shalunov, B. Teitelbaum, A. Karp, J. Boote, and M. Zekauskas. A one-way active measurement protocol (OWAMP). RFC 4656, IETF, September 2006.
[27] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies with Rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2–16, 2004.
[28] N. Spring, D. Wetherall, and T. Anderson. Scriptroute: a public Internet measurement facility. In *Proc. USITS*, pages 17–17, Berkeley, CA, USA, 2003. USENIX Association.
[29] M. Toren. tcptraceroute. http://michael.toren.net/code/tcptraceroute/.
[30] T. Wolf, R. Ramaswamy, S. Bunga, and N. Yang. An architecture for distributed real-time passive network measurement. In *Proc. IEEE MASCOTS*, 2006.