# 18th ICCRTS

**Title of Paper**

Monitoring in Disadvantaged Grids

**Topics**

Topic 8: Networks and Networking

Topic 5: Experimentation, Metrics, and Analysis

Topic 7: Architectures, Technologies, and Tools

**Name of Authors**

Trude H. Bloebaum, Frank T. Johnsen,
Norwegian Defence Research Establishment (FFI)

Gunnar Salberg
Narvik University College, Norway

**Point of Contact**

Trude H. Bloebaum
Norwegian Defence Research Establishment (FFI)
P.O. Box 25
NO-2027 Kjeller
Norway

E-mail: Trude-Hafsoe.Bloebaum@ffi.no

# Monitoring in Disadvantaged Grids

**Abstract**

In disadvantaged grids communication resources are scarce and variable. Thus, it is important that middleware and applications are able to adapt to the available capacity. This means that knowledge of the networking environment the C2 software is operating in is of great importance.

Monitoring of networks carrying data is important to improve the quality of service. There are two main types of monitoring: Monitoring in the planning phase, i.e., for example testing that the maximum achievable throughput is in accordance with the agreement. Monitoring during deployment, i.e., when the network is actually being used has different goals. Here, it is important to know the current load (e.g., link utilization) to be able to shape and control data traffic in a coherent manner. There exist many solutions capable of doing this, but they are mostly geared towards use on the Internet and in corporate networks. Sending data to and from mobile units, like military vehicles moving in a combat zone gives challenges that may make some current tools unsuitable. In this paper we focus on freely available tools, and attempt to identify which tools are suitable for the planning phase and the deployment phase in disadvantaged grids.

## 1. Introduction

The Service Oriented Architecture (SOA) paradigm provides a set of design guidelines to create loosely coupled interoperable systems. Currently Web services technology is the most common means for implementing such service oriented systems. In fact, NATO has identified Web services as *the key enabling technology for realizing NNEC* [1]. Furthermore, NATO has stated that there is also a need for Quality of Service (QoS) support in networks and middleware to ensure proper use of resources and timely delivery of important data. However, current Web services standards do not provide complete QoS support. The standards are geared towards Internet usage, where bandwidth is abundant and disruptions seldom happen. In underdeveloped and degraded environments like military tactical networks, on the other hand, best effort handling of network resources is inadequate. Here, applications and middleware need to adapt to changes in resource availability to ensure prioritization of important data. In order to make such adaptation, the system needs to know about the environment it operates in. In other words, there is a need to monitor resource use/availability and adapt accordingly.

In real deployments, multiple systems and potentially a large number of users rely on the same limited network resources for communication. As we have seen in [4, 5], integrating existing systems using Web services can lead to large amounts of information having to be exchanged. In order to transport as much of this information as possible, with as little resource use as possible, it is vital that one is able to adapt communication behavior to the available resources. Such adaptation relies on accurate knowledge about the current network resource situation.

Building a correct and up-to-date view of the network resource availability is challenging at best, particularly in a limited capacity network where resource availability is also highly variable. The simplest approach to network capacity measurement relies on flooding the network with probe information, and measuring the delivery performance of these probes. This approach gives a fairly

accurate measurement of the maximum network capacity, but the act of performing the measurement is likely to disrupt any ongoing communication.

In this paper, we investigate some of the different approaches to network monitoring that exist in order to find an approach which is suitable for use in limited capacity networks. Furthermore, we test the suitability of a couple of the more promising network monitoring tools in an emulated tactical communications network.

## 2. Problem statement

Previously, we have created a system for admission control [2], to ensure that the network is not overly congested by not allowing more clients into the network than it can handle for a certain traffic class. Further, we have developed a system for prioritizing important Web services traffic through the network and in scheduling for an application server [3]. However, both admission control and prioritization requires knowledge of the network to make informed and suitable decisions on behalf of the user.

An interim solution [6] on the way to SOA deployment performs monitoring of available network resources, providing the user with a graphical representation of network load. This enables user awareness of congestion, allowing for manual compensation in the use of the C2 systems. This approach requires continuous and active user involvement in reaction to changes in networking resources. Our ultimate goal is to let the middleware or the systems themselves perform the adaptation, so that the user can focus solely on the more important, tactical tasks at hand. Before systems can make autonomous adaptations, they need definite measurements as input to the adaptation decision algorithm. In this paper we aim to address the first step towards implementing such functionality, i.e., we attempt to identify tools that can be used to provide the measurement data. We focus on the tactical environment where communication resources are scarce, in particular we consider the so-called *disadvantaged grids*.

Disadvantaged grids are characterized by low bandwidth, variable throughput, unreliable connectivity, and energy constraints imposed by the wireless communications grid that links the nodes [7]. Monitoring under such conditions is not trivial, and existing tools may not be entirely suitable for use in that kind of environment. For a tool to be usable, it needs to fulfill certain requirements, as discussed Section 4.1.

## 3. Network monitoring

Network monitoring is a term which can encompass a number of different techniques designed to provide information about a number of different network aspects. Common tasks that rely on some sort of network monitoring are fault detection, configuration management, checking adherence to security policies and restrictions, and monitoring of the performance of both network components such as routers and load on network links.

In this paper we focus on the performance aspects of network monitoring, and how to measure network performance under less than ideal conditions.

### 3.1 Performance measurements

Network measurement techniques are often categorized by how they perform their measurements, either actively or passively.

*Active* measurement techniques insert additional traffic into the network, most commonly in the form of network probes. The measurements are then based on the performance these probes experience. Active probing can be used to measure a number of different network parameters, and requires few local resources, such as CPU and storage. The main downside however, is that these techniques insert traffic into the network, thus increasing the traffic load. This means that an active measurement is *intrusive*. Some techniques rely on saturating the network with probes, and will thus affect non-measurement traffic attempting to use the same network. Thus, the degree of intrusiveness varies by the technique applied.

*Passive* measurements, on the other hand, perform measurements of the network traffic already in the network. This means that passive measurement techniques in general have less network overhead, but are limited to information that can be derived from ongoing communication. Due to the fact that these techniques rely on analyzing information owned by others, privacy and security issues might also arise. In addition, the passive measurement techniques can be CPU intensive, as they rely on analyzing a potentially huge amount of ongoing traffic.

In addition to these two main categories there exist a number of techniques than can be considered to be hybrids, as they combine elements from both techniques. Hybrid solutions include those techniques which

- send probes and monitor these probes in a passive manner,
- use passive measurements when enough information can be derived from ongoing traffic, and active probing to compensate for lacking data, and
- insert information, and thus additional overhead, into the network by adding data to already existing data packets instead of sending dedicated probes

Another central difference between network measurement techniques is what they measure, both with respect to which performance parameters they cover, and which part of the network the measurement is done over.

A network path consists of all the routers and links a packet must traverse to get from one host to another, and measurements that cover a full path are called *end-to-end* measurements. A number of per-hop measurement techniques also exist, and these measure the performance of each link separately. End-to-end measurements are useful for end systems that need to adapt their network usage to current resource availability. This is because the measurements provided by an end-to-end mechanism give results that are directly applicable to the exact communication that will be taking place. One downside of relying on end-to-end measurements is that two measurements of paths that appear to be independent of each other might turn out to rely on the same limited resource. As an example, a measurement between nodes A and B and an unrelated measurement between A and C might both traverse the same bottleneck link without this being detectable from looking at the measurement results.

*Per-hop* measurements avoid this problem by measuring each link independently, and can thus give a more detailed view of current network conditions. The downside to this type of measurement is that the measurement must be performed by routers or other devices within the network itself. In addition, measurement results must be transmitted to nodes requiring this information. This exchange of measurement results can generate enough network load for it to be significant in low capacity networks.

As mentioned above, measurement techniques also vary with respect to what they measure. Common parameters include various types of delay, loss rates, link capacity and available bandwidth. There are multiple ways of measuring these parameters, but in order to accurately compare tools it is important to distinguish between the various aspects of performance measurements that exist. Throughout this paper we will utilize the definitions from [9], which present the following table defining terms related to capacity and bandwidth measurements:

| | |
|---|---|
| Capacity | The maximum rate at which packets can be transmitted by a link |
| Narrow link | The link with the smallest capacity along a path |
| Available bandwidth | A link's unused capacity |
| Tight link | The link with minimum available bandwidth along a path |
| Cross traffic | Traffic other than the traffic created by the probing |

Table 1: Terms and notions relating to available bandwidth measurement (from [9])

# 4. Monitoring of disadvantaged grids

Existing monitoring tools are mostly implemented, designed, and intended for civil networks. Tactical networks differ from civil networks, most notably due to the much lower capacity. We first discuss requirements for tools that are to be employed in tactical networks, before we list some of the currently available tools.

### 4.1 Requirements analysis

There are a number of network performance parameters that can be measured using various tools, with each tool being designed to measure a specific subset of these parameters. Which tool to use thus relies on the intended usage of the measurement information.

Monitoring tools may leverage different measurement techniques (e.g., active monitoring, passive monitoring), which leads to differences in intrusiveness and resource use while obtaining measurement results. Ideally we want tools with *low intrusiveness* for use in tactical networks.

Measurements may be performed per-hop or end-to-end in the network. For actual use, that is, servers delivering data to clients, a measurement of the entire path is beneficial because such a measurement has the same granularity as the client/server communication. Thus, we want a tool that performs *end-to-end* measurements.

Different tools take different approaches to monitoring and results analysis. Some gather information over time and perform offline calculations later, while others gather information and perform calculations at run-time and may provide measurement results in (near) real-time. *Responsiveness* in the tool is important when it is being used in a deployed network, and the measurements are subject to decisions regarding adaptations of network use. Responsiveness is of

lesser importance when a tool is being used to plan a network, since that situation seldom requires answers in real-time.

Ideally we want a tool that can be made a part of the middleware or application. In deployments, military networks often constitute a system-of-systems, meaning that one can encounter heterogeneous networking technologies. In this case hardware specific solutions are too restrictive and cannot be used across network boundaries, which is a drawback when attempting to measure a path. This means that a *generic software solution* is preferable to tools that are proprietary and require special hardware support.

In this paper, we focus on measurement techniques suitable for disadvantaged grids, in which bandwidth is one of the main limiting factors for efficient communication. When planning a network, identifying the capacity is most important in that respect. In a deployed network, where the measurement results are to be used for adapting to the current resources, the achievable bandwidth is what we need to measure. This means that for planning, we need a tool that can *accurately measure the capacity in low resource environments*. Conversely, we need a tool that can *accurately measure the available bandwidth in a deployed network*.

### 4.2 Selected tools

In our study we focus on freely available tools, many of which are released as open source. We chose not to focus on vendor specific solutions, as these often dictate that routers of a certain make and model must be available in the network. Thus, our findings can be of interest to a larger community as we do not focus on one specific networking technology. There exist prior evaluations of such tools (see e.g., [13]), however they are geared towards Internet and corporate network use. This means that the results may not be directly applicable to tactical networks, where capacity is the limiting factor rather than cross traffic.

Table 2 shows the tools we identified, their key properties, and a theoretical evaluation of their suitability for use in disadvantaged grids. In this paper we show this summary for brevity, and focus on the most suitable tools in an actual evaluation. For the complete discussion about all the tools, see [8].

| TOOL | Category | Technique/ Method | Protocol | Metrics | | | | |
|---|---|---|---|---|---|---|---|---|
| | Active/ Hybrid/Passiv | | | Bandwidth Type | Path or per Hop | RTT | Delay | Loss |
| | (1) | (2) | (3) | (4a) | (4b) | (4c) | (4d) | (4e) |
| **Badabing** | Active | Packet Pair | TCP | Bandwidth Capacity | multi/path | | | X |
| **Bing** | Active | VPS | ICMP | Bandwidth Capacity | Path | X | | X |
| **BProbe** | Active | Packet pair | ICMP | Available Bandwidth | multi/path | X | | |
| **BWPing** | Active | Packet pair | ICMP | Bandwidth Capacity | multi | X | | |
| **Clink** | Active | VPS | UDP | Bandwidth Capacity | multi/path | X | | X |
| **CoralReef** | Passive | Packet Train (timeout) | TCP | No | multi | X | | X |
| **CProbe** | Active | Packet pair | ICMP | Available Bandwidth | multi/path | X | | |
| **IEPM** | Hybrid | Packet pair | TCP, UDP | Achievable bandwidth | multi / path | X | X | X |
| **Iperf** | Active | Path flooding | TCP, UDP | Bandwidth Capacity | multi/path | X | X | X |
| **Netperf** | Active | Path flooding | TCP, UDP | Bandwidth Capacity | multi/path | | X | X |
| **Nettimer** | Hybrid | VPS/ tailgating | TCP | Bandwidt capacity | single/ per-hop | X | | |
| **PathChar** | Hybrid | VPS | UDP, ICMP | Achievable bandwidth | per-hop | X | X | X |
| **Pathload** | Active | SLOPS | UDP | Available Bandwidth | multi/ path | X | X | X |
| **Pathrate** | Active | Packet pair, packet train | UDP | Bandwidth Capacity | multi/ path | | | |
| **PChar** | Hybrid | VPS | UDP, ICMP | Achievable bandwidth | single/ per-hop | X | X | X |
| **Pipechar** | Active | Packet pair | UDP | Available Bandwidth | single/ per-hop | | | |
| **SPAND** | Passive | Packet pair | ICMP | Bandwidth Capacity | path | X | X | X |
| **SProbe** | Active | Packet pair | TCP | Bandwidth Capacity | multi | X | | |
| **STAB** | Active | Packet tailgating, Packet trains | UDP | Bandwidth Capacity | one-way | | X | X |
| **Sting** | Passive | Packet pair | TCP, ICMP | No | path | | | X |
| **Surveyor** | Active | Packet pair | UDP | No | one-way | X | X | X |
| **TReno** | Hybrid | TCP simulation | UDP, ICMP | Available Bandwidth | multi | | X | X |

Table 2: Summary of monitoring tools

## 5. Experiments

Our choices for further testing were Pathload and Iperf. Based on the characteristics of these two tools, they seem to be good candidates for monitoring in tactical networks [8]. Pathload is able to measure the available bandwidth, whereas Iperf can measure both the available bandwidth and the capacity for the same type of links. This means that Pathload could potentially be used at run-time to measure the bandwidth that is available. Iperf, on the other hand, could potentially be used in the planning/deployment phase to identify the maximum achievable bandwidth. Thus, these two tools represent candidates for the two tasks we seek to fulfill.

### 5.1 Pathload

Pathload [10] is an active measurement tool for estimating the available bandwidth end-to-end of a path. To perform a test, access to both the sender and receiver ends (server and client, respectively) is required. The bandwidth estimation technique Pathload uses is based on sending periodic packet streams. The variety used in pathload is called self-loading periodic streams (SLOPS), and is a technique that can be used to estimate the available bandwidth. Of the freely available tools, previous tests have shown it to be one of the most accurate [13].

Pathload is built as a sender and receiver process. The sender transmits a periodic packet stream to the receiver. The sender timestamps each packet in the stream prior to its transmission, whereas the receiver records the arrival time of each packet and calculates the relative one-way delay (OWD). Pathload uses the relative magnitude of OWDs in its bandwidth estimation, meaning that the measurement methodology does not require clock synchronization. When a stream is received, the receiver inspects the sequence of relative OWDs to check whether the transmission rate is larger than the available bandwidth (this can be detected because the stream causes a short-term overload in the tight link in the path). Thus, the receiver can decide whether the stream rate is larger than the available bandwidth based on the self-loading effect of the periodic streams. The tool uses UDP for the streams, and a TCP connection between the two end points as a control channel for transferring details regarding the characteristics of each stream. Pathload does not base its results on a single stream; instead it sends a so-called “fleet” of streams. All streams in a fleet have the same size and number of packets, but a new stream is sent only when the previous stream has been acknowledged. This introduces an idle interval allowing the path to clear of any delayed packets. This approach means that pathload is less intrusive than many other measurement tools, as it can measure the available bandwidth without greatly affecting the throughput of other connections, and also without causing a persistent increase in queuing delays or losses in the path. This is in contrast to e.g., tools that flood the path.

Apart from bandwidth measurements, Pathload is also able to measure both data losses and delay.

### 5.2 Iperf

Iperf [11] is a network monitoring tool, implemented in C++, that uses both UDP and TCP streams in order to measure various network performance metrics between two hosts. It is an application level tool which requires running an Iperf server and an Iperf client on two different hosts. The fact that Iperf is on the application level means that it cannot accurately determine what goes on the network

level, so it uses the inherent properties of the two network protocols it supports to gain knowledge about the network path it is monitoring.

When using the UDP protocol, only the receiver can accurately know the throughput that is being offered by the network. The sender only knows its sending rate, but since the packets sent might be dropped along the way, the UDP measurements are done on the receiver side. The receiver periodically reports its findings back to the sender. Note that there is no way to know where along the end-to-end connection the bottleneck is; it is for instance possible that Iperf's UDP measurements reports a throughput that is lower than the actual capacity if the receiving host does not manage to process all incoming packets. The UDP measurements can be used to measure UDP throughput, loss, jitter and delay.

Iperf can also be used to measure the throughput that can be achieved when using TCP for transport. Unlike the UDP measurements, TCP can be used to perform measurements on both the sender and receiver side.

Sender side measurements are performed by looking at the amount of TCP data that the sender can successfully deliver to the TCP/IP stack on its host machine. Since TCP only accepts data from the application as long as the TCP buffers are not full, the amount of data delivered to the TCP/IP stack can be used to determine how much payload data is being successfully acknowledged by the receiver. Note that this means that the TCP measurement might, depending on the TCP buffer size, report a somewhat too high throughput initially, until buffers fill up.

In addition, it is possible to measure TCPs end-to-end throughput on the receiver side as well. This is done by the application measuring how much actual payload data it receives. Increases in congestion and re-transmissions will cause the observed rate (both on the sender and the receiver side) to drop.

**5.3 Test setup**

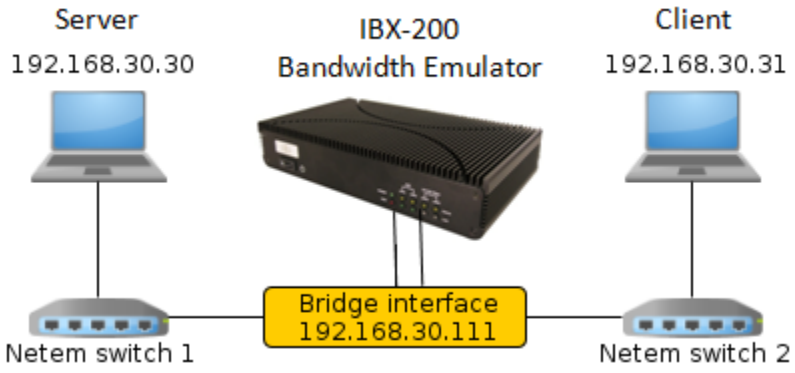



Figure 1: Test setup

The tests were conducted using the setup shown in Figure 1. We used netem [12] as our network emulator, installed on an IBX-200 embedded computer. Netem was set as a bridge in the network, so that it could emulate the network as needed to reflect conditions in tactical networks.

The tools were tested using bandwidths from 1 kbps up to 64.8 kbps. We used the default configurations of the performance monitoring tools.

### 5.4 Pathload results

Pathload uses a default data packet size of 1472 bytes, a payload of 1030 Kbytes, 100 packets in a stream, and 12 streams in a fleet.

Pathload uses the fleet of streams to identify the available bandwidth. If the result from a stream in the fleet is deemed inconclusive by the algorithm, then that stream is discarded. This means that if networking conditions are such that the algorithm cannot decide anything from any stream in a fleet, the tool may not be able to calculate a result at all.

The test results are presented in Figure 2. In some places Pathload estimated a higher available bandwidth than the actual links had in some of the tests, and in other tests the result was lower. This is to be expected from a technique that aims to have low intrusiveness, as more aggressive probing usually can yield more accurate results. Pathload was not able to measure tight links lower than 14.4 kbps.

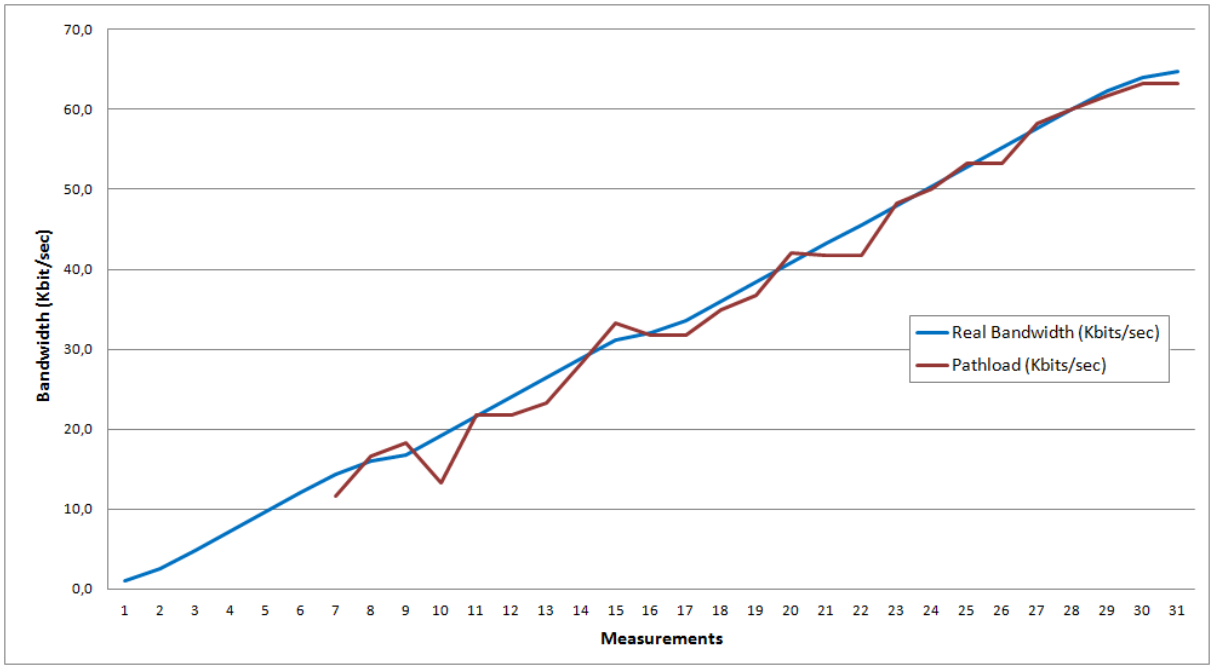



Figure 2: Pathload bandwidth measurements

### 5.5 Iperf results

In our experiments we used both versions of Iperf, testing both the UDP and TCP methods. The two tests were performed separately and independently. Note that when it comes to bandwidth, the UDP- and TCP-based methods measure different things and the results can thus not be compared directly to each other.

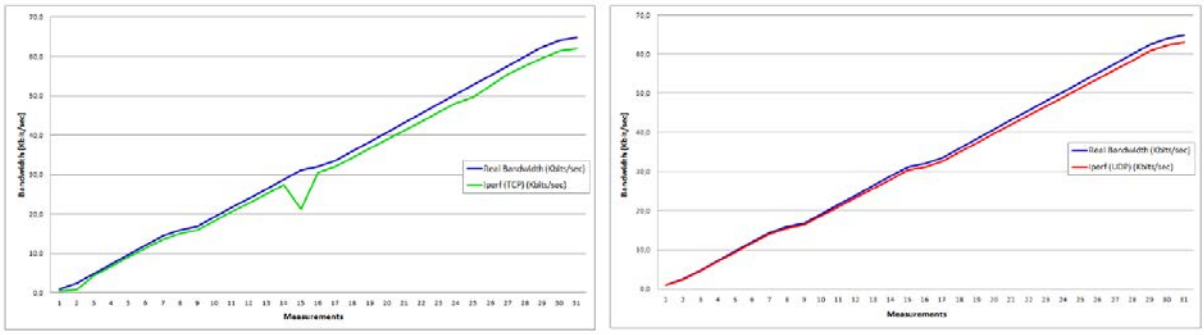
The UDP based tests relies on flooding the network with UDP traffic, and will thus negatively influence any other traffic attempting to use the network at the same time. This method allows Iperf using UDP to achieve good accuracy when it comes to measuring the capacity of the narrowest link, and thus the maximum possible throughput of the network path. Figure 3b shows that Iperf using UDP measures capacity with a high degree at all the tested capacities. This

measurement method is highly intrusive, and should only be used during dedicated planning and testing phases, as it will negatively impact other traffic.

Iperf using TCP does not measure the total capacity of the link, but rather the throughput that can be achieved using TCP. Measuring in this way means that the results represent neither the capacity nor the available bandwidth, but rather the TCP-friendly fair share of the total capacity that the sender can use. This differs from an available bandwidth measurement since using TCP for measurements will cause other TCP traffic to reduce their sending rate in order to accommodate the new stream. Figure 3a shows the results of using the TCP based measurement across a path with no cross traffic. In this scenario, the TCP measurements can be used to estimate the capacity of the link. The results show that the TCP measurements give a throughput that is slightly below the actual capacity of the link, which is as expected.

Due to the way TCP does its congestion control, using multiplicative back-offs, a loss of a packet can cause a temporary drop in the measured throughput, as shown in measurement 15 in Figure 3a.

The Iperf TCP measurements are less intrusive that the UDP-based method, and can also be used while other traffic is utilizing the network.




(a) Iperf measurements - TCP protocol.

(b) Iperf measurements - the UDP protocol.

Figure 3: Iperf bandwidth measurements

**5.6 Pathload and Iperf comparison**

Iperf (UDP) measures the capacity accurately, but is intrusive due to flooding. An average of all tests shows that Iperf (UDP) has less than 3% error rate (delta) between the real bandwidth and the measured bandwidth. The Iperf (TCP) measurements identify the possible TCP throughput, and have error rates between 4% and 5%, with only a few exceptions.

Iperf using UDP meets all the requirements for a planning tool to identify the capacity of a path in the network. The knowledge gained from using this tool can allow a network operator to verify that a given network performs as expected, and thus allow the operator to take measures if necessary. Such measures may include changing the network configuration and configuring applications to not use more network resources than what is expected to be available.

Pathload and Iperf (TCP) are both candidates for use in a deployed network to measure the available bandwidth. Pathload, however, cannot be used in all deployments, as it is unable to measure the tightest links (<14.4 kbps). Iperf (TCP), on the other hand, can deliver fairly accurate results for all the bandwidths we tested. But, while Pathload is designed to exhibit low intrusiveness, no particular

concern towards this has been voiced for Iperf. This means that Iperf (TCP) may adversely affect other traffic in the network while attempting to identify the possible TCP throughput.

The availability of a monitoring mechanism that provides up-to-date information about the network can be a valuable tool for both network operators and network users. Knowing the current state of the network will allow the users to adapt their behavior to changing network conditions, and thus enable a more agile resource use compared to a network where such information is not available.

In addition, a network operator may leverage information about network paths in order to detect performance issues and network outages as soon as they occur. This allows the network operator to change network configuration parameters and routing behavior in an attempt to mitigate the effects of a persistent, but unexpected network issue.

Here we have investigated two tools for network monitoring. Future work includes looking into an automated adaptation mechanism for middleware based on monitoring results. This is the next step, moving from the need of operator intervention to a fully flexible and agile system. However, many considerations need to be made when pursuing such as approach: First, there is the issue of computational overhead – the middleware itself adds overhead, and the integration with a monitoring tool coupled with the application logic necessary to perform automated adaptation further increases this overhead. Investigations must be made into whether the resource usage gains outweigh the drawbacks of added overhead. Next, one also needs to consider the computational and network overhead of the monitoring tool itself. The amount of overhead generated by a tool depends on a number of factors, e.g. the type of tool (active vs. passive as discussed previously) and the configuration of the tool (less frequent measurements mean less overhead, but at the same time less accuracy). Finally, even though there exist a number of freely available tools, the common denominator for these is that they are first and foremost designed for use in networks with higher capacity than disadvantaged grids. Thus, it could be beneficial to consider developing a tool specifically for the lowest capacity networks where the overhead of existing tools may adversely affect the operational utilization of the network.

## 6. Conclusion

In this paper we have introduced different approaches to monitoring, and identified two tools for further testing: Iperf and Pathload. Both are active monitoring tools that should be capable of identifying the capacity and/or available bandwidth of a path. Our goal is to test whether these tools can be used for monitoring in disadvantaged grids. We set up a testbed leveraging netem as our network emulator, using it to emulate the networking conditions of disadvantaged grids. We varied the capacity between 1 and 64.8 kbps in our experiments.

We found that the UDP monitoring mode of Iperf measures capacity accurately, and is able to provide results for the entire capacity range we tested. We recommend this tool for use when planning networks, when the goal is to verify the capacity available.

In a deployed network one can use either the TCP mode of Iperf or Pathload to estimate the achievable bandwidth. Iperf (TCP) was able to estimate the possible TCP throughput for all the capacities we tested, whereas Pathload was unable to measure at capacities below 14.4 kbps. Iperf is a rather intrusive tool, meaning that even if it is capable of delivering the necessary measurements it

will flood the network in its attempt to do so. Pathload is somewhat less intrusive with its SLOPS approach, though it too will affect the network to some extent. It seems that of the tools we considered, there are significant drawbacks when considering their use in a deployed tactical network. Due to intrusiveness, the requirements in Section 4.1 are not fully satisfied, and further research is recommended. Of the two tools tested, we recommend using Pathload because it is the least intrusive of the two. However, since it does not function below 14.4 kbps, it cannot be used in all deployments.

## References


[1] P. Bartolomasi, T. Buckman, A. Campbell, J. Grainger, J. Mahaffey, R. Marchand, O. Kruidhof, C. Shawcross, and K. Veum. “NATO network enabled capability feasibility study,” V2, October 2005.

[2] Frank T. Johnsen, Trude Hafsøe, Mariann Hauge, Øyvind Kolbu, “Cross-layer Quality of Service Based Admission Control for Web Services,” IEEE HeterWMN 2011, Houston, TX, USA, 9 December 2011

[3] Frank T. Johnsen, Trude H. Bloebaum, Jørgen Nordmoen, Jan A. S. Bremnes, Stig Tore Johanessen, Magnus L. Kirø, Ola Martin T. Støvneng, and Håvard Tørresen, “Role-based Quality of Service for Web Services”, IEEE HeterWMN 2012, Anaheim, CA, USA, 3 December 2012

[4] Frank T. Johnsen, Trude H. Bloebaum, Ketil Lund, and Espen Skjervold, “Towards operational agility using service oriented integration of prototype and legacy systems”, 17th ICCRTS, Fairfax, VA, USA, June 19-21 2012

[5] Andersonn Kohl , Élvio Carlos Dutra e Silva Junior, Jeronymo Mota Alves de Carvalho, Thiago Mael de Castro, “SYSTEMIC APPROACH AND TECHNICAL SOLUTION FOR LEGACY SYSTEMS INTEGRATION”, 17th ICCRTS, Fairfax, VA, USA, June 19-21 2012

[6] Christoph Barz and Norman Jansen. “Towards a Middleware for Tactical Military Networks – Interim Solutions for Improving Communication for Legacy Systems,” MCC 2011, 17-18 October 2011, Amsterdam, Netherlands.

[7] A. Gibb, H. Fassbender, M. Schmeing, J. Michalak, and J. E. Wieselthier. “Information management over disadvantaged grids,” Final report of the RTO Information Systems Technology Panel, Task Group IST-030 / RTG-012, RTO-TR-IST-030, 2007.

[8] Gunnar Salberg, “Monitoring in Disadvantaged Grids”, Master’s thesis, Department of Technology, Narvik University College, Norway, June, 2012.

[9] Venkat Mohan. , Y. R. Janardhan Reddy, K. Kalpana, “Active and Passive Network Measurements: A Survey”, (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (4) , 2011, 1372-1385

[10] Manish Jain and Constantinos Dovrolis, “Pathload: A Measurement Tool for End-to-End Available Bandwidth”, In Proceedings of Passive and Active Measurements (PAM) Workshop, 2002.

[11] Iperf monitoring tool. http://sourceforge.net/projects/iperf/

[12] The Linux Foundation. http://www.linuxfoundation.org/collaborate/workgroups/networking/netem

[13] Emanuele Goldoni and Marco Schivi, “End-to-End Available Bandwidth Estimation Tools, an Experimental Comparison”, TMA'10 Proceedings of the Second international conference on Traffic Monitoring and Analysis, ZURICH, SWITZERLAND, APRIL 7, 2010