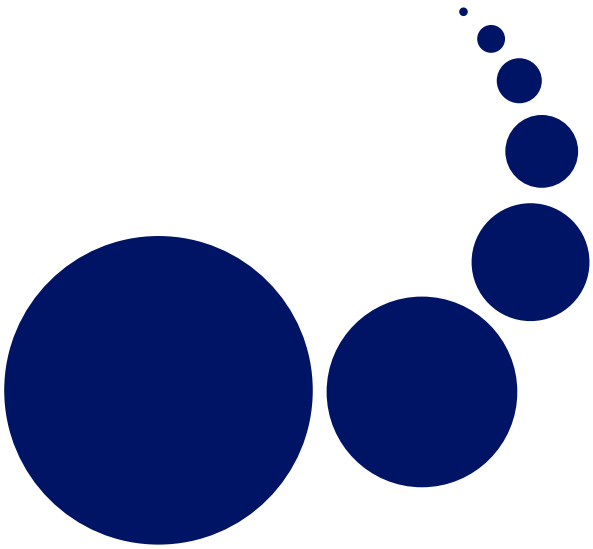
SCALABLE COMPUTING
Practice and Experience

Special Issue: Network Management in Distributed Systems

Editors: Jemal H. Abawajy, Mukaddim Pathan, Al-Sakib Khan Pathan, and Mustafizur Rahman



Volume 11, Number 4, December 2010

ISSN 1895-1767

SUBSCRIPTION INFORMATION: please visit http://www.scpe.org

Scalable Computing: Practice and Experience

Volume 11, Number 4, December 2010

TABLE OF CONTENTS

Scalable Computing: Practice and Experience
Volume 11, Number 4, pp. i–ii. http://www.scpe.org
ISSN 1895-1767

INTRODUCTION TO THE SPECIAL ISSUE: NETWORK MANAGEMENT IN DISTRIBUTED SYSTEMS

The emergence of Web as a ubiquitous platform for innovations has laid the foundation for the rapid growth of the Internet and emerging distributed computing platforms. Other than being a great source of fun and enjoyment, hundreds and thousands of people around the world rely on these systems for various tasks related to their livelihoods. The overwhelming growth of Internet and its users is a reality, which has put new thoughts among the research community to devise new ideas for giving coverage to a huge number of people around the globe. Side-by-side, the use of mobile and wireless devices such as PDA, laptop, and cell phones for accessing the distributed systems has paved the ways for related distributed computing technologies to flourish through recent developments. However, the increasing scale, complexity, heterogeneity, and dynamism of Internet and distributed computing systems with respect to communication networks, resources and applications have made such systems brittle, unmanageable and insecure. This scenario calls for innovative solutions to deal with the complexity, dynamism, heterogeneity, and uncertainty of distributed systems and provide a holistic approach for the development of systems that can meet the requirements of performance, fault tolerance, reliability, security, and Quality of Service (QoS). As of today, numerous applications depend on distributed computing and networking infrastructures to operate. The management of these functionally and geographically distributed infrastructures is crucial to ensure effective operation of applications. This special issue aims at compiling the recent advancements as well as some basic issues in the fields of network management in distributed computing and Internet-based systems.

In general terms, a distributed system is defined as a system that consists of several autonomous computers that communicate through a computer network. Usually there is a common goal that all these networked computers try to achieve. A computer program that runs in a distributed system is called a distributed program, and distributed programming is the process of writing such programs. The same definitions could be rewritten by considering different types of computing devices instead of only computers in traditional sense. With the advancements of micro-electronics, many low-resource computing devices have been devised that can form useful distributed systems. One of the most recent attractive technologies is wireless sensor networks. Sensors are small computing devices that form a distributed system or network over a target area to collect and process required data. The workloads and network traffic in such type of distributed system could be divided among the participants. Thus, sensor network becomes a major type of distributed system. Other commonly known distributed systems are various types of telecommunications networks like; telephone networks and cellular networks, computer networks such as the Internet, World Wide Web (WWW) and Peer-to-Peer (P2P) networks, multiplayer online games network and virtual reality network, distributed database network, distributed information processing network (e.g. airline ticket reservation system), aircraft control system, industrial control system, clustered and grid systems. As the topics of distributed systems are very diverse and vast, many types of network management issues fall under the intended talking point of this special issue.

As this special issue aimed at getting the latest advancements in the areas of network management in distributed systems, the papers have been submitted from the authors of diverse backgrounds working on various aspects of the issues related to the scope of the call. Among the submitted papers, after rigorous reviewing, some papers were suggested for some modifications. Accordingly authors have responded with the improved versions of the papers. The works from diverse fields related to the call for papers make this special issue a good common platform for getting the essence of a broad area of network management issues in distributed systems.

Out of the four accepted papers for this special issue, two are selected and extended versions of the papers presented at IDCS 2010, the *3rd International Workshop on Internet and Distributed Computing*, held in conjunction with the *12th IEEE International Conference on High Performance Computing and Communications* (HPCC 2010) at Melbourne, Australia in September 2010. Other two accepted papers are received via open call and have been selected through rigorous peer-review by renowned researchers in the area of distributed computing.

Network management as network layer resource signaling protocols provide a useful set of tools to dynamically install, maintain, and manipulate state in network nodes. To this end, Bless and Röhricht in their paper, *"Implementation and Evaluation of a NAT-Gateway for the General Internet Signaling Transport Protocol"*, present the design, implementation, and evaluation of an application level gateway for the General Internet Signalling Transport (GIST) protocol, which translates GIST messages in a way that allows to establish signaling sessions between any two GIST nodes across a Network Address Translation (NAT) gateway. In their work, the

authors have presented evaluation results in a real testbed environment. Results show only a slight overhead for processing initial Query messages on a GIST-aware NAT gateway in the range of about 2.15ms on average. All subsequent GIST messages show almost no processing overhead and do not exceed 0.026ms on average. These findings are used to demonstrate how this work can be directly adopted in real-world implementations.

Wireless Mesh Networks (WMNs) is an emerging technology for next generation wireless broadband networks. Routing in WMN is one of the most challenging issues to support stringent QoS requirements of resource management applications. Sen in this paper, *"An Efficient and User Privacy-Preserving Routing Protocol for Wireless Mesh Networks"*, attempt to meet the user QoS requirements while addressing security and privacy concerns in WMN, by proposing an efficient and reliable routing protocol that provides user anonymity in WMNs. The protocol is based on an accurate estimation of the available bandwidth in the wireless links and a robust estimation of the end-to-end delay in a routing path, and minimization of control message overhead. The user anonymity, authentication and data privacy is achieved via a protocol that is based on Rivest's ring signature scheme. Simulation results are presented to demonstrate that the proposed scheme is more efficient than some of the existing routing protocols. Specifically, it has been shown that the protocol has very low overhead and has high network throughput with a large number of source nodes in a WMN.

Recent years have witnessed the development of transport layer protocols to avoid congestions in Wireless Sensor Networks (WSN) and provide data or application level reliability support, thereby ensuring QoS requirements of heterogeneous WSN applications. In this context, Sharif et al. has proposed a light weight transport protocol in the paper, *"ERCTP: End-to-End Reliable and Congestion Aware Transport Layer Protocol for Heterogeneous WSN"*. The protocol achieves high data reliability using the distributed memory concept within network and results in minimum packet drop due to congestion by the effective implementation of congestion detection and rate adjustment scheme that uses stochastic control framework. Authors have evaluated the proposed scheme with existing transport protocols and demonstrated that it can control congestion and exhibits good throughput, limited end-to-end data packet latency, and high data packet reliability and low per packet communication cost.

Introduced in late 90's, Grid computing provides a loosely coupled, heterogeneous and geographically dispersed distributed computing platform. With the increase of application complexities in Grid computing environments, the presence of security threats are becoming prevalent. As human administrations are unable to cope with the amount of work required to properly securely the computing infrastructure, there is the need to find innovative solutions to overpower the limitations of manual management of the system, i.e. slow speed, increasing chances of errors and unmanageability by human administrators. In this context, Chopra and Singh has presented an agent-enabled Self-Protection Model (SPM) in the paper *"Implementing Self-Protection in Distributed Grid Environment"*. This model adopts intelligent agents to dynamically organize system management with centralized control. At the system level, each element contributes its capabilities on the functions of system management and cooperate with each other to implement autonomic computing for grid systems. With this approach, the authors seek to achieve system robustness and scalability, that has been demonstrated in the results from the conducted simulation experiments.

With the above four selected papers, we have tried to make this issue as rich as possible within our capacity. We have directed the authors to provide high quality manuscripts with minimal errors within the texts. We would like to thank the contributing authors of the papers for their efforts to deliver all the required items within the specified timelines. Specially, we would like to express our appreciation to the Editor-in-Chiefs, Dr. Marcin Paprzycki and Dr. Dan Petcu, and the reviewers for their immense support and cooperation for preparing this special issue. We hope this special issue would be a good addition to the current practices and experiences in the fields of network management in distributed systems.

Jemal H. Abawajy, *School of Information Technology, Deakin University, Geelong, VIC 3216, Australia,* jemal.abawajy@deakin.edu.au,
Mukaddim Pathan, *Information Engineering Laboratory, CSIRO ICT Centre, Acton, ACT 2601, Australia,* mukaddim.pathan@csiro.au,
Al-Sakib Khan Pathan, *Department of Computer Science, International Islamic University Malaysia, Kuala Lumpur, Malaysia* sakib@iium.edu.my
Mustafizur Rahman, *Department of Computer Science and Software Engineering, University of Melbourne, Parkville, VIC 3010, Australia* mmrahman@csse.unimelb.edu.au

Scalable Computing: Practice and Experience
Volume 11, Number 4, pp. 329–343. http://www.scpe.org
ISSN 1895-1767

IMPLEMENTATION AND EVALUATION OF A NAT-GATEWAY FOR THE GENERAL INTERNET SIGNALING TRANSPORT PROTOCOL*

ROLAND BLESS† AND MARTIN RÖHRICHT†

Abstract. The IETF's Next Steps in Signaling (NSIS) framework provides an up-to-date signaling protocol suite that can be used to dynamically install, maintain, and manipulate state in network nodes. In its two-layered architecture, the General Internet Signaling Transport (GIST) protocol is responsible for the transport and routing of signaling messages. The strong presence of Network Address Translation (NAT) gateways in today's Internet infrastructure causes some major challenges to network signaling protocols like NSIS. The address translation mechanisms performed by common NAT gateways are primarily concerned with address information contained in the IP and transport layer headers. Signaling sessions between two signaling peers do, however, rely on address information contained in GIST data units. If a non GIST-aware NAT gateway merely adapts addresses in the IP and transport headers only, inconsistent state will finally be installed at the signaling nodes. In this paper we present the design, implementation, and evaluation of an *application level gateway* for the GIST protocol, that translates GIST messages in a way that allows to establish signaling sessions between any two GIST nodes across a NAT gateway.

Key words: distributed systems, Internet, middleware, performance evaluation

1. Introduction. Network layer resource signaling protocols provide a useful set of tools to dynamically install, maintain, and manipulate state in network nodes. As a prominent example, the ReSource ReserVation Protocol (RSVP) was once designed to establish state in network routers for Quality-of-Service reservations on demand. In response to some limitations of RSVP, the *Next Steps in Signaling* (NSIS) working group of the Internet Engineering Task Force (IETF) designed an up-to-date signaling framework that is not limited to a particular signaling application only [1]. The NSIS framework follows a two-layered architecture where the lower layer, called *General Internet Signalling Transport* (GIST) [2] protocol, is solely responsible for the routing and transport of signaling messages, whereas the upper layer, called NSIS Signaling Layer Protocol, implements the actual signaling application's logic, e.g., for Quality-of-Service resource reservations [3].

Network Address Translation (NAT) [4] was once introduced to map non-publicly usable ("private") addresses to public IP addresses. NATs mostly deal not only with the translation of IP addresses of different address realms, but also with the mapping of TCP or UDP transport protocol ports within a session (so called Network Address and Port Translation – NAPT, in the following we also use the term NAT for NAPT). Due to this additional port multiplexing, one public IP address can serve many private IP addresses and thus mitigate the potential shortage of IPv4 addresses. But the price is lost end-to-end transparency [8] complicating the design and life of transport or application protocols. Main problems are the missing address binding information for communication initiated from the public side of the NAT gateway as well as the IP address information contained in IP payloads. For the latter so-called Application Level Gateways (ALGs) inside the NAT try to help translating the application specific address information in the payload. Their implementation is, however, sometimes error-prone, immutable and does not cope with newer protocol versions. Moreover, a variety of different NAT types exist [5] that cause even more complications due to their different behavior, aside from configurations with nested NATs. One solution to prevent the use of ALGs is to avoid putting IP address information in the payload, which is, however, not always possible.

In this sense, the strong presence of NAT gateways in today's Internet infrastructure causes some major challenges to signaling protocols like such of the NSIS suite. Since NSIS protocols are controlling resources in the network layer, they have to carry IP address information similar to some transport protocols. Not only that bindings must be established in these gateways to properly exchange messages with the actual signaling destination. Furthermore, NSIS signaling messages have to carry IP address information in their GIST payload that would not be translated by an ordinary NAT gateway. Hence, in order to allow NSIS signaling sessions to be established even across NAT gateways, the NAT gateway must be GIST-aware and rewrite some of the addressing information within the signaling message's payload.

*This work was supported by the Federal Ministry of Education and Research of the Federal Republic of Germany (support code 01 BK 0809, G-Lab, http://www.germanlab.de/. This is an extended and reorganized version of a paper of the same title that appears in the Proceedings of HPCC, Melbourne, Australia, September 2010, pp. 659-664

†Institute of Telematics, Karlsruhe Institute of Technology (KIT), Zirkel 2, P.O.Box 6980, 76049 Karlsruhe, Germany ({bless, roehricht}@kit.edu)

The GIST protocol specification [2] already describes a dedicated *NAT traversal object (NTO)* that carries necessary translation information which can then be used by a GIST-aware NAT gateway. This NTO is designed in a modular way which allows for the traversal of a number of subsequent NAT gateways. It must, however, be created, inserted into a signaling message's payload, and later interpreted by a GIST-aware NAT gateway. In this paper we present the design, implementation, and evaluation of an application level gateway for the GIST protocol, which translates GIST messages in order to allow for the establishment of signaling associations between GIST nodes across a NAT gateway.

The rest of this paper is organized as follows. Section 2 gives an overview of GIST's protocol operation with all necessary protocol specific details and discusses related work. Section 3 illustrates GIST handshake message exchange procedures across a legacy NAT gateway. In Section 4 and 5 we provide an analysis of the design and the implementation of a GIST-aware NAT gateway. Section 6 provides evaluations and performance measurements before we conclude in Section 7.

2. Background and Related Work. Different from other signaling protocols, such as RSVP, the Next Steps in Signaling framework follows a two-layered architecture as depicted in Figure 2.1. This two-layer split allows for a flexible and extensible Internet signaling protocol suite, where the routing and transport of signaling messages is separated from the actual signaling application's logic.

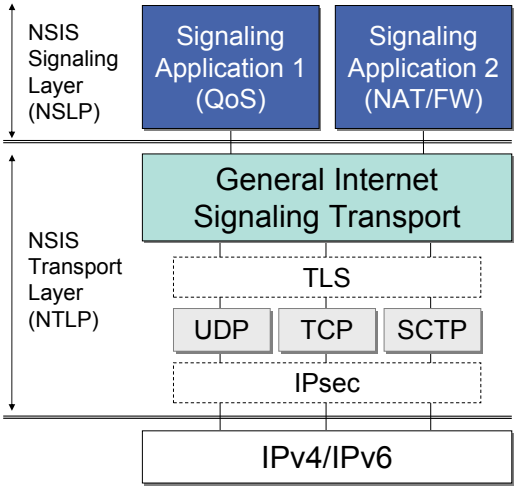


FIG. 2.1. *Layered Architecture of the NSIS Protocol Framework*

The NSIS framework therefore consists of a lower layer called *NSIS Transport Layer Protocol* (NTLP), whereas the signaling application is realized via specific *NSIS Signaling Layer Protocols* (NSLP).

The General Internet Signalling Transport (GIST) protocol is a concrete realization of an NTLP. It makes use of already present underlying transport protocols, like UDP, TCP, TCP with TLS, or SCTP and provides two modes of operation, namely a datagram mode (D-mode for UDP data) and a connection mode (C-mode for TCP and SCTP). GIST is responsible to discover NSIS-capable nodes along a data flow's path, to establish messaging associations between two adjacent GIST nodes and to transport signaling messages along this route.

In order to setup state between two nodes, GIST uses a three-way handshake, consisting of QUERY, RESPONSE, and CONFIRM messages as depicted in Figure 2.2. Subsequently exchanged messages from a particular signaling application are carried via GIST DATA messages (not shown in Figure 2.2).

QUERY messages are always sent in a so-called encapsulation mode (Q-mode) by the *Querying Node* and are intercepted by a *Responding Node*. In order to setup a messaging association between two adjacent GIST nodes, the initial QUERY contains a context-free flag (C-flag) which indicates that a new routing state must be established. The following RESPONSE indicates whether a CONFIRM must be sent by the Querying Node. In order to defend against denial of service attacks, the Responding Node can use a so-called *delayed state installation* mechanism where the installation of routing state is delayed until a final CONFIRM message arrives upon which a return routability check can be performed.

The way signaling messages are routed—e.g., strictly following the data path—is specified in a *Message Routing Method* (MRM). The MRM contains all necessary addressing information encapsulated in a *Message*

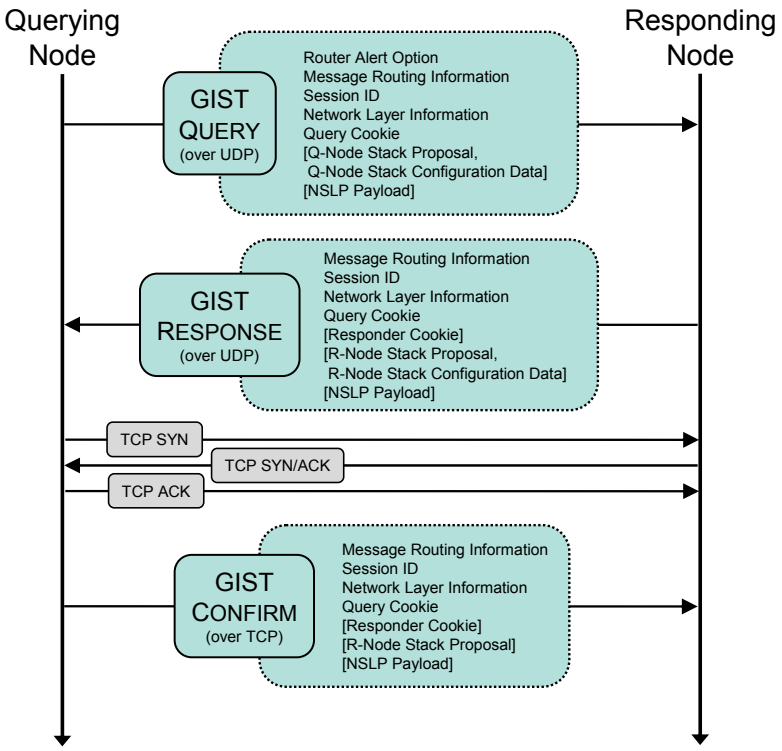


FIG. 2.2. *A detailed view of the message sequence of a GIST handshake for initial state setup, including a TCP-based messaging association*

Routing Information (MRI) object, e.g., the source and destination's IP addresses, as well as the transport protocol and port numbers. Multiplexing of messaging associations, i. e., the re-use of existing messaging associations for multiple flows and sessions, is controlled by a *Network Layer Information* (NLI) object. The NLI basically contains a unique peer identity and an interface address through which a signaling node can be reached. Due to the specific address information contained in the MRI and the NLI, these objects are of particular importance when it comes to address translations within a NAT gateway.

Common NAT traversal techniques, such as STUN [6] or TURN [7] require the presence of supporting servers and help applications to find out information about address translations so that they can put the correct address information into the payload. The dependency on other server and control protocols for network layer signaling protocols like NSIS is inappropriate at best. For instance, Raz et al. specified the construction of an SNMP-aware NAT gateway [9]. In recent work, Huang et al. even propose the use of a programmable NAT [11]. Even though different design aspects outlined in these papers are of particular interest in the context of this work, the specific solutions provided cannot directly be applied to an application level gateway for the GIST protocol.

In two Internet-Drafts Pashalidis and Tschofenig provide problem statements on a GIST NAT traversal [12] and a GIST legacy NAT traversal [13]. In order to traverse legacy NAT gateways, the authors propose the use of UDP tunnels for signaling and data traffic. However, this approach relies on static NAT bindings and does not differentiate between signaling and pure data traffic. Furthermore, the UDP tunnels add a significant level of complexity and overhead to the GIST peers. The proposal towards a GIST-aware NAT gateway on the other hand comes with a *transparent* and a *non-transparent* approach.

In the *transparent* approach the GIST header fields are simply translated by the NAT gateway as it is done with the layer 3 and layer 4 address information. This approach allows the NAT gateways to be used completely transparent for the GIST peers participating in a signaling session. However, if the signaling traffic is cryptographically protected between two GIST peers residing on either side of the NAT gateway and the NAT gateway is not actively participating as an NSLP entity (therefore terminating the cryptographic protection), the transparent approach cannot be applied. As cryptographic protection of NSIS signaling traffic is realized

either via IPsec or via TLS, signaling messages are encrypted above layers 3 or 4 and hence GIST data units cannot be modified following the transparent approach by a NAT gateway anymore.

The *non-transparent* approach uses the aforementioned NAT traversal object which is included by the GIST-aware NAT gateway into initial QUERY messages and which is then subsequently echoed back by the GIST responder. Different from the transparent approach discussed above, the non-transparent approach does not suffer from cryptographic protection of signaling traffic as it only affects the initial QUERY and the corresponding RESPONSE message which must always both be sent unencrypted. The work presented in this paper is therefore based on the non-transparent approach outlined in [12], but instead of storing the entire translation information in an NTO object, the translation information stored within our approach is split between the NTO object and the GIST Responder Cookie.

3. Analysis of Legacy NAT Handling. In order to setup state for a signaling flow between two adjacent NSIS signaling peers, GIST messages always carry addressing information in their header fields. However, once a GIST signaling message reaches a legacy NAT gateway, this gateway will only be concerned with address translations within the IP and UDP/TCP headers. In order to get a better understanding of the requirements of a GIST-aware NAT gateway (c.f. Section 4), this section provides an analysis of GIST's handshake message exchange across legacy NAT gateways.

The GIST specification introduced the notion of a *source addressing mode flag* (S-flag) in its common header which can optionally be used to indicate whether the flow source address corresponds to the signaling source address. By default, the S-flag is set to 0 but can also be used for legacy NAT detection during a GIST handshake in which case the S-flag must be set to 1. Depending on the locations of the NAT gateway and the Querying Node (QN) we must consider two different signaling message proceedings. The QN can either be located on the internal or the external side of the NAT gateway.

3.1. Querying Node on the Internal Side of the NAT Gateway. Figure 3.1 illustrates the case in which the QN is located on the internal side of the NAT gateway and the S-flag is set to 0. In this case, the QN is equipped with an IP address of a private subnet (10.1.2.1 in Figure 3.1) and sends an initial QUERY towards a signaling endpoint with IP address 141.3.71.56 in downstream direction. Therefore, the MRI of the QN holds the IP address of the QN as its source address and the IP address of the signaling endpoint as its destination address. The NLI of the QN is equipped with a unique interface peer-identifier of the QN and the corresponding interface address (in this case: 10.1.2.1).

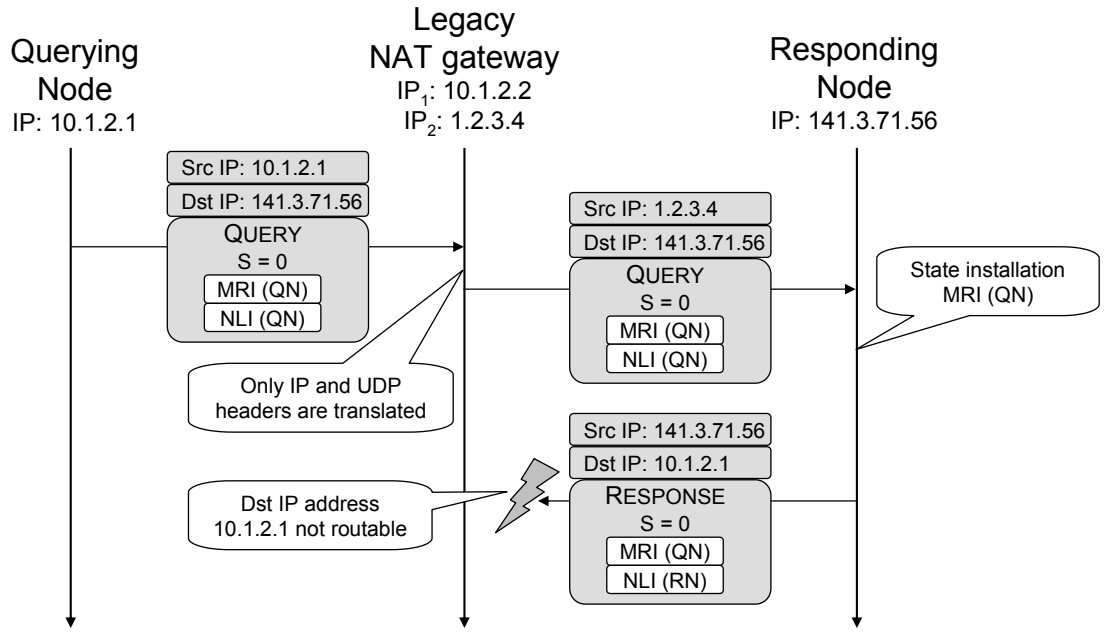


FIG. 3.1. *Legacy NAT traversal of a GIST handshake message exchange in case the Querying Node is on the* internal *side of the NAT gateway and the S-flag is not set (S = 0)*

The QUERY eventually reaches the legacy NAT gateway which only performs IP and UDP header address translations and is then forwarded towards the signaling destination endpoint. The Responding Node (RN) intercepts the message in order to establish a messaging association with the QN and therefore installs state according to the information contained in the MRI (QN). The RN generates a RESPONSE depending on the interface address given in the NLI (QN). This RESPONSE is, however, not routable and hence will not be delivered to the QN. The QN will then try to retransmit QUERY messages (not shown in the figure) and will eventually time out in case retransmitted QUERY messages keep S = 0.

In case a QUERY is (re-)transmitted with S = 1, the RN must check the source address provided in the IP header with the interface address in the NLI. As the address information contained in these two fields does not match, a legacy NAT can be assumed to be located on the path. Message routing methods, such as the default path-coupled MRM, need corresponding address translations which cannot be provided by legacy NAT gateways. As depicted in Figure 3.2, the RN returns an *Object Type Error* message including the MRI (QN) within its error subcode, indicating that no correct messaging association can be established based on this MRI, upon which the handshake message exchange is terminated.

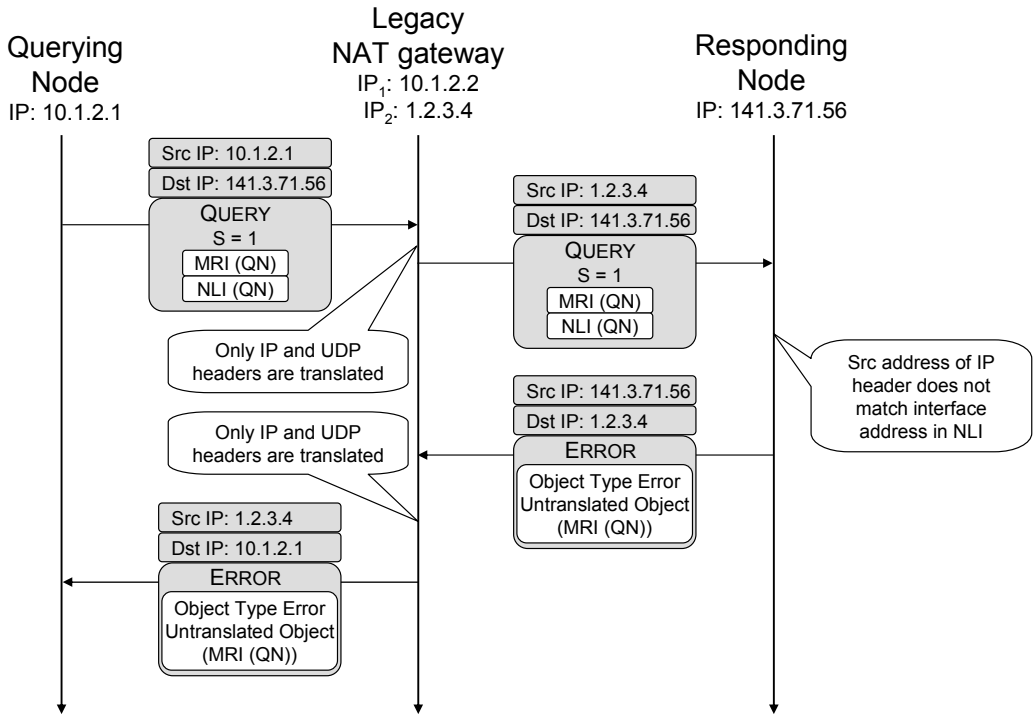


FIG. 3.2. *Legacy NAT traversal of a GIST handshake message exchange in case the Querying Node is on the* internal *side of the NAT gateway and the S-flag is set (S = 1)*

According to the GIST specification, the error message should use the destination address of the original IP datagram as its source address in the IP header. This makes the response more likely to be accepted by the NAT gateway and finally be forwarded towards the QN. In case this error message is blocked by the NAT, the QN will further retransmit QUERY messages until it eventually times out.

3.2. Querying Node on the External Side of the NAT Gateway. When the legacy NAT gateway is located on the side of the RN instead of the QN, GIST messages may only traverse the gateway in case static configurations have been carried out on the NAT gateway. As illustrated in Figure 3.3 we assume the QN to be equipped with a public IP address (1.2.3.4), whereas the RN has an IP address from the NAT gateway's private subnet (10.1.2.2).

The initial QUERY traverses the NAT gateway where IP and UDP header addresses are translated. Once this modified data packet reaches the RN, it sends a RESPONSE with S = 1 and an IP address derived from the NLI (QN), i. e., 1.2.3.4 in this case, back to the QN. The legacy NAT gateway forwards the RESPONSE as ordinary UDP traffic and only translates IP and UDP header information.

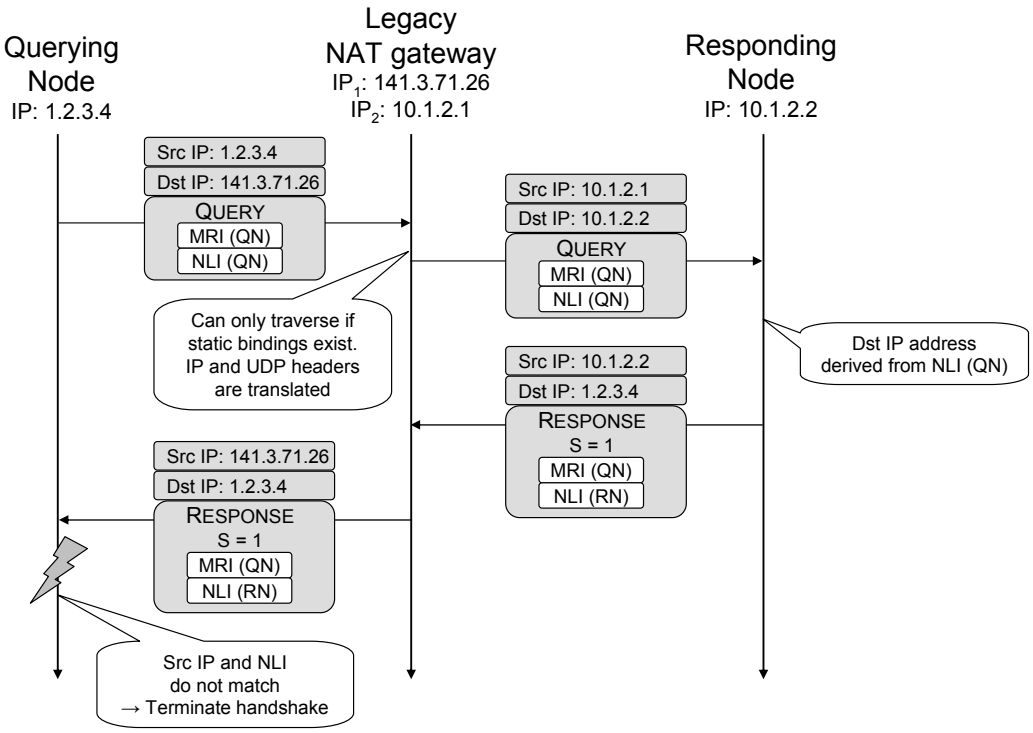


FIG. 3.3. *Legacy NAT traversal of a GIST handshake message exchange in case the Querying Node is on the* external *side of the NAT gateway and the S-flag is set (S = 1)*

Once the RESPONSE reaches the QN, it detects a mismatch between the IP header's source address (141.3.71.26 in Figure 3.3) and the NLI(RN) (10.1.2.2 in Figure 3.3) upon which the QN terminates the GIST handshake.

This section revealed that a legacy NAT gateway will always terminate a GIST message handshake and thus prevents GIST messaging associations to be set up, no matter if the legacy NAT gateway is located on the side of the Querying Node or on the side of the Responding Node. Therefore, NSIS signaling cannot be performed across a legacy NAT gateway.

4. Analysis and Design of a GIST-aware NAT Gateway. As outlined above, a legacy NAT gateway that performs address translations in the IP and transport layer headers only, but not in the GIST header, creates inconsistent states for signaling flows at the end-points which eventually leads to a termination of the signaling session at an early stage. In order for NSIS signaling to install routing state even across NAT gateways, such NAT gateways must be GIST-aware.

Note, however, that GIST-awareness does not mean that the NAT gateway must have the entire NSIS framework or any of the particular NSLPs installed. Instead, a GIST-aware NAT gateway must only modify GIST signaling messages that are exchanged without any routing state installed.

The GIST protocol specification [2] already introduced a so-called *NAT traversal object* (NTO) that stores address information about translated objects and needs to be included in initial QUERY messages by each intermediate NAT gateway. Figure 4.1 depicts the object definition of an NTO.

The modular design of an NTO allows to keep track of all necessary address information that has been replaced by NAT gateways along the path. The most important part of this object is the *Original Message Routing Information* element of the QN which is used among the signaling peers as referral to a common routing state. Furthermore, it contains a field where the number of NATs is stored that were traversed by the message and a list of translated objects which is of variable length and contains 16-bit wide fields for corresponding objects that have also been translated by intermediate NAT gateways. In particular, the Network Layer Information (NLI) object, which contains the outgoing interface address and peer-identity, as well as a modified Stack-Configuration-Data element, if present, should be included into the list of translated objects.

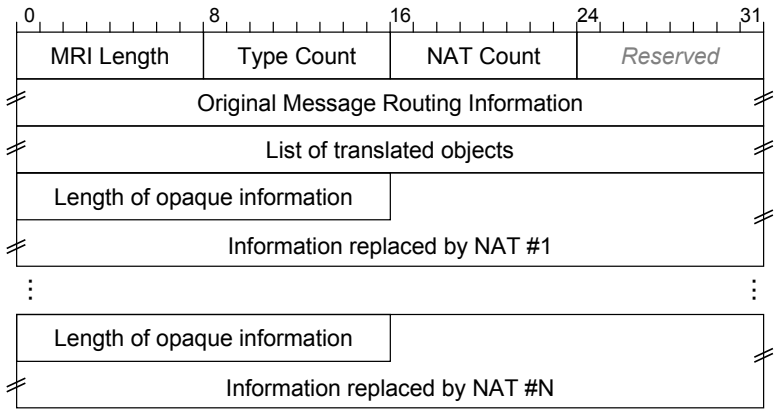


FIG. 4.1. *Type definition of a NAT traversal object according to [2]*

According to the GIST specification, a GIST-aware NAT gateway should only process QUERY messages that have the C-flag set as well as D-mode messages carrying the NAT traversal object. All remaining GIST messages, i. e., messages sent in C-mode or D-mode without NAT traversal object should be processed by the NAT gateway as ordinary data traffic. Since C-mode messages are carried inside a TCP transport connection they are not really visible to a NAT gateway anyway.

The reason for subsequent GIST messages, i. e., initial CONFIRM and subsequent DATA messages, not to be processed by the GIST-aware NAT gateway is, that messages after the initial QUERY and RESPONSE need to refer to a common MRI. Following this approach, this is the MRI of the Querying Node which must be exchanged via the NAT traversal object as outlined above.

An exemplified GIST three-way handshake between a Querying Node (QN) and a Responding Node (RN) across a GIST-aware NAT gateway is illustrated in Figure 4.2. First of all, a GIST-aware NAT gateway must establish bindings for the signaling data flows, e.g., for subsequent C-mode signaling. Once an initial QUERY passes the gateway, it must create new MRI and NLI objects that reflect the translated address information and adds a NAT traversal object that lists all translated objects. The NTO may also carry NAT specific information that is useful for the NAT gateway, e.g., carrying state or state referral information. In case a NAT traversal object already exists, this object must be extended by additionally modified objects. After that, the message is encapsulated in Q-mode and forwarded further along the path.

The Responding Node installs routing state according to the information contained in the original MRI (QN) and the translated MRI (NAT) and NLI (NAT). The triple (MRI, NSLP-ID, Session-ID) is used as referral to routing states. The QUERY's NAT traversal object as being received by the Responding Node is copied into the RESPONSE. Furthermore, this RESPONSE's MRI uses the original and unmodified values of the Querying Node.

In any further GIST messages that cross the GIST-aware NAT gateway and that belong to a flow for which bindings already exist, only IP addresses and TCP/UDP ports are translated. Subsequently sent CONFIRM and DATA messages always carry the untranslated MRI and NLI objects of the Querying Node and are not processed by the NAT gateway with respect to addresses contained in GIST PDUs.

Special rules apply to the delayed state installation mechanism where a Responding Node does not install state before it received a final CONFIRM. As outlined above, CONFIRM messages do only carry the untranslated MRI and NLI objects, preventing the Responding Node from a correct routing state installation in this case. The GIST protocol specification leaves this issue open to the implementation. However, the specification suggests to use the *Responder Cookie*, in which all of the translated objects that were received by the Responding Node can be carried securely. This Responder cookie is finally echoed back by the Querying Node's subsequent CONFIRM message, upon which the Responding Node receives the necessary information in order to properly install routing state.

5. Implementation. In this section, we describe the design and implementation of our GIST-aware NAT gateway and explain how we perform NSIS signaling even across NAT gateways. The implementation can be basically divided into a kernel and a user-space part as depicted in Figure 5.1. The kernel part of the GIST-aware NAT gateway intercepts and filters GIST packets. In case a GIST packet's payload must be further modified,

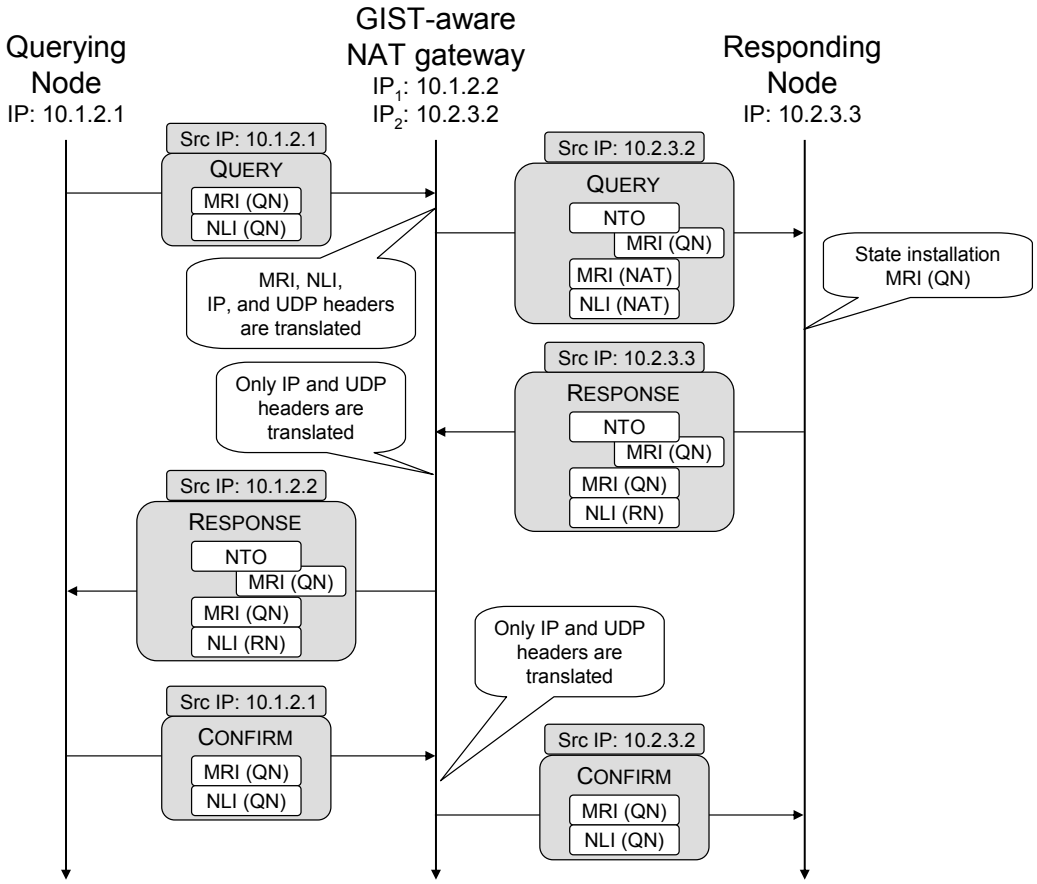


FIG. 4.2. *GIST handshake between a Querying Node (QN) and a Responding Node (RN) via a GIST-aware NAT gateway*

it is passed to a user space thread that performs the remaining packet translations before the modified packet is forwarded by the kernel.

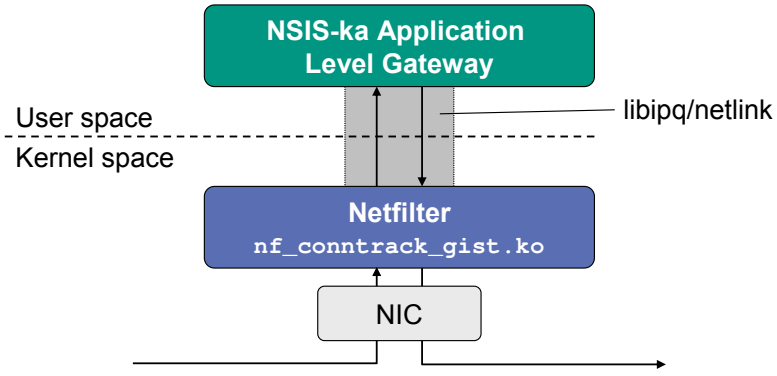


FIG. 5.1. *Architecture of an NSIS compatible Application Level Gateway*

Packet filtering is achieved in our implementation by means of the Linux netfilter framework [14]. Initial QUERY and subsequent RESPONSE messages are intercepted and passed into an `ip_queue` data structure. The communication between kernel and user space is realized on behalf of the Linux netlink messaging system [15].

5.1. Implementation of the Kernel Module. The Linux netfilter framework provides a set of hooks that correspond to different positions of a packet on its way through the protocol stack. Hooks can be used to perform rules or actions, e.g., on incoming packets, on packets being forwarded, or on outgoing packets. We

designed a GIST kernel module that registers itself at the netfilter's *PRE_ROUTING* and *POST_ROUTING* hooks and intercepts GIST QUERY and RESPONSE messages.

Once a GIST QUERY enters the netfilter, the connection is tracked and a `conntrack` structure is initialized for a subsequent RESPONSE. In case the netfilter instance receives a RESPONSE, NAT rules are created depending on the RESPONSE message's payload. These rules can then be used to establish corresponding NAT bindings for IP and protocol port translations for any subsequent messages that do only rely on the functionality of a legacy NAT.

Initial QUERY and subsequent RESPONSE messages must, however, be further processed by the user space part of the application level gateway and are therefore passed to user space by means of the `ip_queue` data structure.

5.2. Implementation of the User Space Part. The user space part of our application level gateway is based on the already existing NSIS-ka implementation [16]. Note however, that it is not necessary to run the entire NSIS-ka suite on a GIST-aware NAT-gateway, so only some NTLP object classes were re-used.

A netlink listener, where messages enqueued by the kernel are received, builds the first part of the application level gateway. Once packets are received by the NSIS-ka application level gateway, the entire PDU is parsed, beginning with the IP and UDP headers, and transferred into a GIST PDU. Address information in the MRI and NLI of GIST QUERY messages must then be translated, according to the NAT processing rules. Furthermore, a NAT traversal object must be inserted right after GIST's common header and the source addressing mode flag must be set to one in a GIST QUERY. After that, the GIST PDU objects can be serialized into byte code and IP and UDP checksums are re-calculated, before the packet is copied into the netlink's message buffer from where it is then sent back to kernel space.

5.3. Interaction between Kernel and Userspace Parts. The resulting work flow of the kernel and userspace parts of the implementation are illustrated in Figure 5.2. Once a packet enters the kernel and its netfilter hooks, the implementation inspects whether the data packet is a valid GIST message, i. e., if the UDP datagram is addressed towards the GIST well-known port 270 and if the first 32 bits of the UDP payload match the specified GIST magic number (`0x4e04bda5`). In case it is a GIST message and it is an initial QUERY (`ip_conntrack_info` set to `IP_CT_NEW`) or a corresponding RESPONSE (`ip_conntrack_info` set to `IP_CT_IS_REPLY`) the data packet will be further processed, otherwise it is forwarded as normal data traffic. An initial QUERY is passed to the user space part via the `NF_QUEUE` return code.

After that the kernel module tests for an NTO in the GIST message. In case it is a RESPONSE and an NTO is present, the connection between the sender and the receiver has already been established. The kernel module then needs to define how subsequent traffic should be handled for this connection and stores the necessary address information in the RESPONSE. Finally the kernel module needs to ensure that the `iptables` rules for the network address translations were updated accordingly and passes the packet on to the user space part.

The user space part uses the API of the `libipq` library to read messages queued in `ip_queue` from kernel space. First of all it parses the packet to see if the Router Alert Option is set. After that, the UDP header is parsed for corresponding port information. In case the GIST message is a QUERY, the source address information contained in the IP and UDP header must be later used for translation. Then the UDP payload is deserialized and parsed, based on existing classes and functions from the NSIS-ka framework, in order to retrieve the corresponding GIST objects. The results are stored in a dedicated peer address structure.

The ALG translation takes one of the most important parts in this process, as GIST objects like the MRI and the NLI must be translated with the addresses stored in the peer address structure. After that, an NTO must be created and inserted in case the packet is a QUERY that does not yet have an NTO. Furthermore, the original NLI is stored in the NTO's opaque information field. If the QUERY already has an NTO, the NAT count is increased by one and the translated types of the NLI and the stack proposal are added to the opaque information field. In case it is a RESPONSE, the NAT count is decreased by one.

Once the ALG processing is done, the GIST message must be serialized into byte-code by means of the NSIS-ka framework, the S-flag and R-flag must be explicitly set, and the GIST hop count must be increased by one in the GIST common header. For a QUERY the C-flag must also be set. Then the IP and UDP checksums must be re-calculated and the `libipq` must be used again in order to send the packet back to kernel space and from there further along the path on the network.

The implementation of the kernel module consists of 420 lines of C code, whereas the GIST-aware NAT gateway consists of additional 680 lines of C++ code, but makes heavily use of already existing libraries and

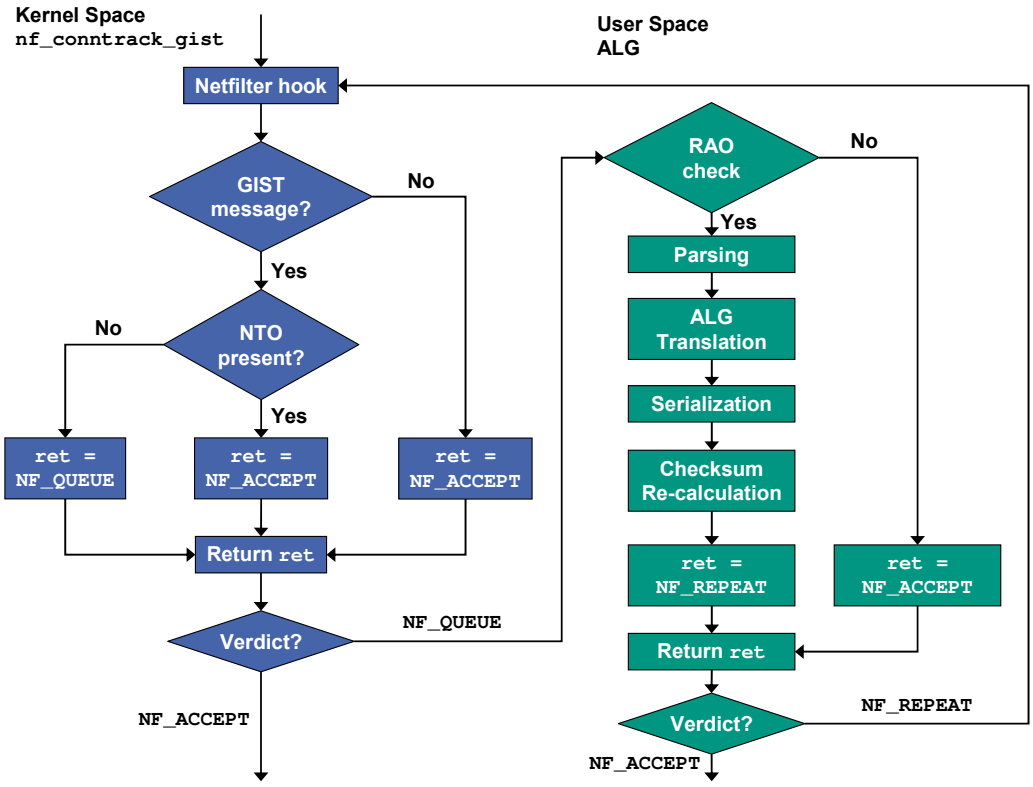


FIG. 5.2. *Illustration of the different processing steps in a GIST-aware NAT gateway between the kernel module and the userspace part*

data structures of the NSIS-ka suite. The GIST implementation and its underlying protocol library, which we had to use for evaluation tests currently contain 40,692 physical source lines of code, mostly based on C++ (93.78%). The code of our GIST-aware NAT gateway implementation is publicly available at `https://svn.tm.kit.edu/nsis/dist/nsis-ka/branches/20100602-gist-aware-nat-gw`.

6. Evaluation. We evaluated the implementation of our GIST-aware NAT gateway in a real testbed environment, consisting of four standard PCs being equipped with Intel Pentium 4 2.8 GHz CPUs, 4 GB DDR-400 RAM, and four 1000TX Ethernet cards. All four PCs ran Ubuntu 10.04 with Linux kernel 2.6.32. The topology is depicted in Figure 6.1 where two NSIS hosts that were equipped with the NSIS-ka framework exchanged signaling messages across two GIST-aware NAT gateways. For evaluation purposes, we assigned IP addresses from the private 10.0.0.0/24 subnet to the PC's interface cards and set up static routing tables accordingly.

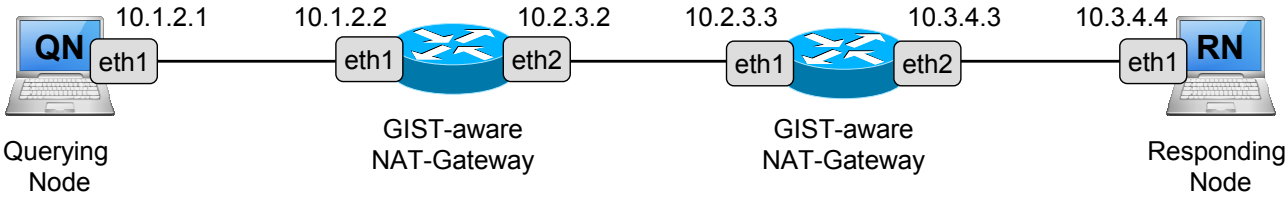


FIG. 6.1. *Evaluation Setup with two hosts communicating across two GIST-aware NAT gateways*

The latency between the two endpoints was intentionally kept small (approximately 0.165 ms, measured by 100 ping tests) in order to concentrate measurements on the pure protocol and processing overhead.

First of all, we measured the time spent by each of the GIST-aware NAT gateways that were used to process QUERY, RESPONSE, CONFIRM, and DATA messages. In order to focus on GIST message processing time, the DATA messages carried only a simple artificial Echo-NSLP payload consisting of 16 additional bytes for a 132

byte long Ethernet frame. The measurement points for the GIST message processing and translation time correspond to the timestamps of tcpdump packet captures at the ingress and the egress interface.

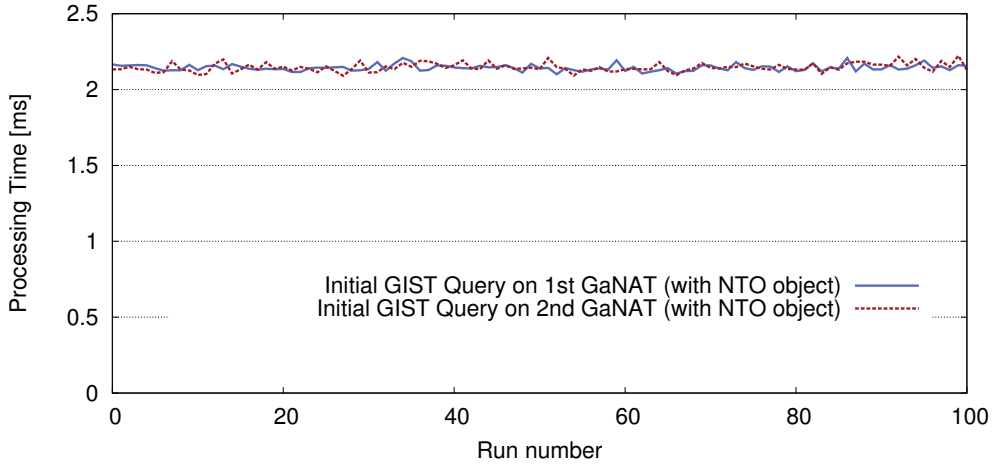


FIG. 6.2. *Processing time of initial* QUERY *messages when NAT traversal objects are included*

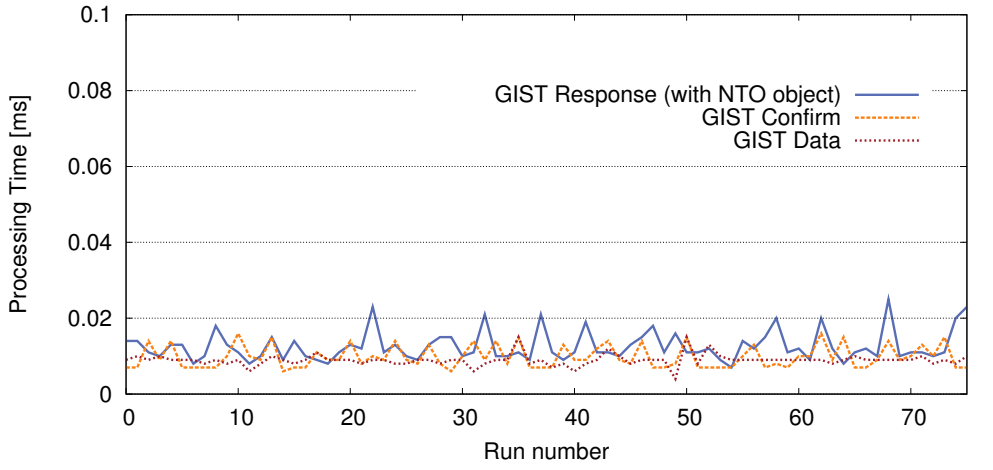


FIG. 6.3. *Processing time of GIST* RESPONSE, CONFIRM *and* DATA *messages on the first GIST-aware NAT gateway using TCP*

Figure 6.2 shows the processing time for initial QUERY messages on both GIST-aware NAT gateways for 100 consecutive runs. As outlined above, these initial QUERY messages must be processed by a GIST-aware NAT gateway by translating MRI and NLI objects and including a new NAT traversal object that carries the original MRI of the Querying Node.

The measurement results show a fairly stable processing time of about 2.153 ms on average with a very small standard deviation of 0.15 ms (cf. Table 6.1) and a 95% confidence interval in the range of (2.1476, 2.1542) ms. Note, that the results for the first and the second GIST-aware NAT gateway are also almost identical.

Measurement results for the processing time of 75 consecutive runs of the remaining GIST PDUs on the first GIST-aware NAT gateway are shown in Figures 6.3 and 6.4. While QUERY messages must always be sent in Q-mode encapsulation, i. e., by using UDP, subsequent GIST messages can be exchanged either via UDP or via TCP, depending on the negotiated protocol stack configuration data.

Note that the results of Figures 6.3 and 6.4 seem to indicate a rather unstable behavior, but it actually stems from the high resolution of the plotted data sets. The absolute time values for all of these three GIST message types are very small ranging from 0.008 ms and 0.026 ms on average and would otherwise not be visible compared to the processing time of initial QUERY messages.

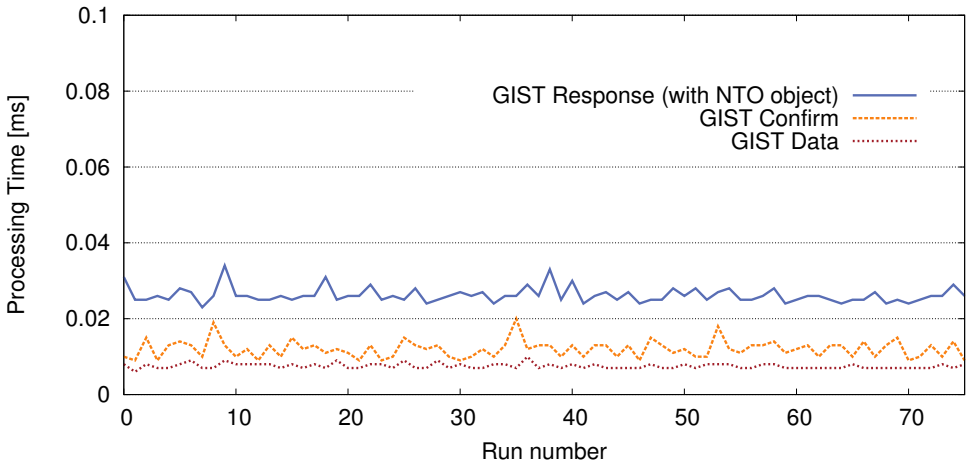


FIG. 6.4. *Processing time of GIST* RESPONSE, CONFIRM, *and* DATA *messages on the first GIST-aware NAT gateway using UDP*

Table 6.1 summarizes the results for the first GIST-aware NAT gateway. The small values of all standard deviations suggest a very stable behavior.

TABLE 6.1
Evaluation results for the overall processing time of different GIST PDUs on the first GIST-aware NAT gateway

Processing time on the first GIST-aware NAT gateway	Avg [ms]	Median [ms]	StdDev [ms]
UDP Query (with NTO)	2.153	2.161	0.152
TCP Response (with NTO)	0.012	0.011	0.004
UDP Response (with NTO)	0.026	0.026	0.002
TCP Confirm	0.010	0.009	0.003
UDP Confirm	0.012	0.012	0.002
TCP Data	0.009	0.009	0.001
UDP Data	0.008	0.007	0.001

Besides measuring the pure processing costs induced on a GIST-aware NAT gateway, we also measured the duration of complete GIST handshakes between the two end points with one consecutive DATA message. We conducted tests for GIST handshakes with and without NAT gateways in-between, in order to obtain a resulting overhead. Figure 6.5 shows the results obtained for complete GIST handshakes in case C-mode was requested, i. e., when TCP connections were used. In this case we only picked traces from our data sets that established an entirely new TCP connection. Therefore, each trace consists of an initial GIST QUERY as starting point until the TCP acknowledgement for the first DATA message is received by the Querying Node.

The results are again fairly stable and show a time difference for the complete handshake duration of about 5 ms between a NAT-free setup (lower two curves) and the use of two GIST-aware NAT gateways on the path (upper two curves). Using delayed state installation induces a small processing overhead in case NAT gateways are used, but no difference can be observed by using no NAT gateways.

Figure 6.6 uses the same setups and measurements in case only D-mode, i. e., UDP, is used. In this case the first time stamp was again the initial QUERY message, whereas the second timestamp had to be the emitting point of the final DATA message. Again, we observe a very stable behavior and this time we cannot detect a difference between using delayed state installation and using normal state installation. The GIST handshake duration is about 1 ms faster when using UDP instead of TCP.

Table 6.2 summarizes the results obtained for complete GIST handshakes for UDP and TCP connections, as well as by using delayed state installation (DSI) or by not using it.

We note that the results obtained for complete GIST handshakes using our evaluation setup are perfectly in-line with the results obtained for the processing overhead of single GIST PDUs. As outlined above, a UDP QUERY induces an overhead of about 2.15 ms on average, whereas subsequent RESPONSE, CONFIRM, and DATA

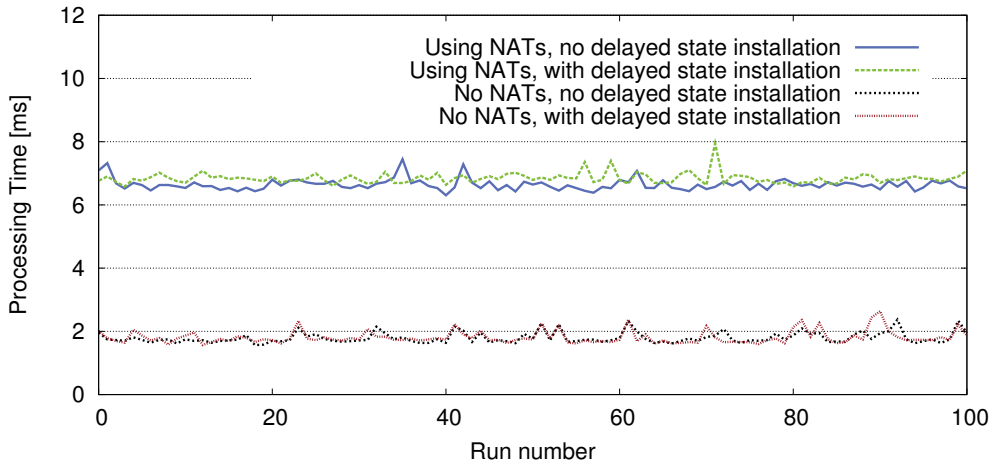


FIG. 6.5. *Duration of complete GIST handshakes with one subsequently sent* DATA *message measured on the Querying Node using TCP*

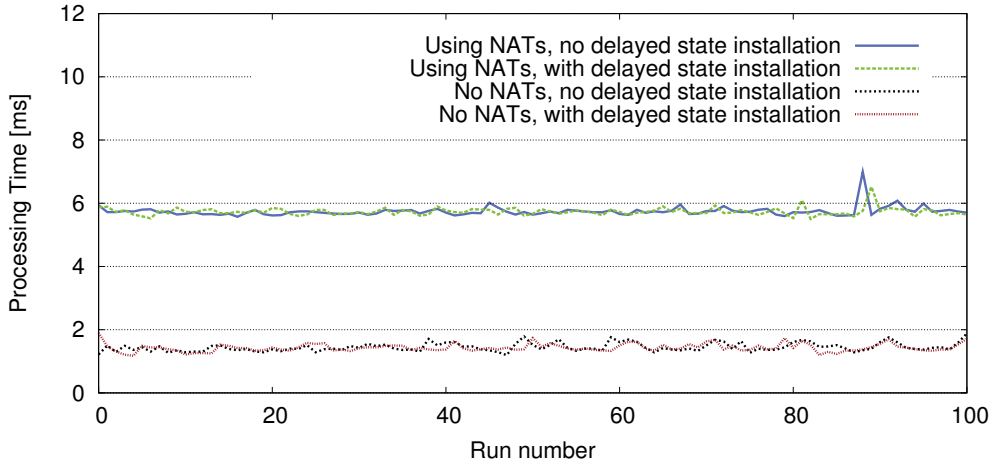


FIG. 6.6. *Duration of complete GIST handshakes with one subsequently sent* DATA *message measured on the Querying Node using UDP*

TABLE 6.2
Evaluation results for the durations of GIST handshakes with one subsequently sent DATA *message*

GIST Handshake duration using TCP	Avg [ms]	Median [ms]	StdDev [ms]
Using NATs, with DSI	6.843	6.820	0.178
Using NATs, without DSI	6.659	6.630	0.182
No NATs, with DSI	1.816	1.746	0.210
No NATs, without DSI	1.797	1.732	0.176

GIST Handshake duration using UDP	Avg [ms]	Median [ms]	StdDev [ms]
Using NATs, with DSI	5.737	5.722	0.127
Using NATs, without DSI	5.744	5.720	0.154
No NATs, with DSI	1.432	1.413	0.124
No NATs, without DSI	1.449	1.407	0.136

messages do only produce an overhead of (0.012 + 0.010 + 0.009) ms in case a TCP connection is used and (0.026 + 0.012 + 0.008) ms in case UDP is used (c.f. Table 6.1). As our setup uses two GIST-aware NAT gateways on the path, the plain processing overhead sums up to 2·2.15 ms+2·(0.012+0.010+0.009) ms = 4.3 ms+2·0.062 ms = 4.362 ms for a TCP connection and $2 \cdot 2.15\,\text{ms} + 2 \cdot (0.026 + 0.012 + 0.008)\,\text{ms} = 4.3\,\text{ms} + 2 \cdot 0.046\,\text{ms} = 4.392\,\text{ms}$ for UDP connections.

7. Conclusions. In this paper we presented the design of a NAT application level gateway for the General Internet Signaling Transport protocol. Following this approach NSIS signaling messages can safely traverse such NAT gateways and routing state can be established even across NATs. The design of the GIST-aware NAT gateway followed mostly the proposed design outlined in the GIST specification. This showed again how a design of a formal specification can be directly adopted in real-world implementations if it was well-defined.

The evaluation results of our implementation show only a slight overhead for processing initial QUERY messages on a GIST-aware NAT gateway in the range of about 2.15 ms on average. All subsequent GIST messages show almost no processing overhead and do not exceed 0.026 ms on average.

Using GIST-aware NAT gateways has also only a small impact on the duration of complete GIST handshakes from end-to-end. While a GIST handshake and a subsequent DATA message can be exchanged in a NAT-free setup within at most 1.82 ms on average, the complete duration of a handshake with two GIST-aware NAT gateways on the path does not exceed 6.84 ms.

Furthermore, the measurement results showed that the use of GIST's delayed-state installation mechanism, which can be used as a denial-of-service attack prevention, does not induce a notable performance overhead, compared to a normal state installation. The processing time of GIST messages could even be further lowered if the implementation runs in kernel-mode or is supported by the hardware chip itself. However, as NAT gateways are commonly used only at the very edge of the network, we would not expect the gateway to handle a big amount of concurrent signaling sessions.

Even though the authors do in general not support the idea of NATs—as they break the end-to-end principle of the Internet architecture—we note that the implementation of the presented approach follows only some basic design mechanisms, which is quite positive. Therefore, an application-level gateway support for NSIS signaling traffic may easily be incorporated into existing NAT implementations, especially the ones which are already based on Linux. Future research in this area may be directed towards the support of SCTP as an alternative transport layer protocol as opposed to UDP or TCP, as SCTP is currently, to the best of our knowledge, not supported by existing NAT gateways.

Acknowledgments. The authors thank Yun Lin for his contributions to the conceptual elaboration of this work and the resulting implementation.

REFERENCES

[1] X. Fu, H. Schulzrinne, A. Bader, D. Hogrefe, C. Kappler, G. Karagiannis, H. Tschofenig, and S. V. den Bosch, *NSIS: A New Extensible IP Signaling Protocol Suite,* Communications Magazine, IEEE, vol. 43, no. 10, pp. 133–141, Oct. 2005.

[2] H. Schulzrinne and R. Hancock, *GIST: General Internet Signalling Transport,* RFC 5971 (Experimental) Internet Engineering Task Force, Oct. 2010. [Online]. Available: `http://www.ietf.org/rfc/rfc5971.txt`

[3] J. Manner, G. Karagiannis, and A. McDonald, *NSLP for Quality-of-Service Signaling,* RFC 5974 (Experimental) Internet Engineering Task Force, Oct. 2010. [Online]. Available: `http://www.ietf.org/rfc/rfc5974.txt`

[4] P. Srisuresh and K. Egevang, *Traditional IP Network Address Translator (Traditional NAT),* RFC 3022 (Informational), Internet Engineering Task Force, Jan. 2001. [Online]. Available: `http://www.ietf.org/rfc/rfc3022.txt`

[5] F. Audet and C. Jennings, *Network Address Translation (NAT) Behavioral Requirements for Unicast UDP,* RFC 4787 (Best Current Practice), Internet Engineering Task Force, Jan. 2007. [Online]. Available: `http://www.ietf.org/rfc/rfc4787.txt`

[6] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing, *Session Traversal Utilities for NAT (STUN),* RFC 5389 (Proposed Standard), Internet Engineering Task Force, Oct. 2008. [Online]. Available: `http://www.ietf.org/rfc/rfc5389.txt`

[7] R. Mahy, P. Matthews, and J. Rosenberg, *Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN),* RFC 5766 (Proposed Standard), Internet Engineering Task Force, Apr. 2010. [Online]. Available: `http://www.ietf.org/rfc/rfc5766.txt`

[8] B. Carpenter, *Internet Transparency,* RFC 2775 (Informational), Internet Engineering Task Force, Feb. 2000. [Online]. Available: `http://www.ietf.org/rfc/rfc2775.txt`

[9] D. Raz, J. Schoenwaelder, and B. Sugla, *An SNMP Application Level Gateway for Payload Address Translation,* RFC 2962 (Informational), Internet Engineering Task Force, Oct. 2000. [Online]. Available: `http://www.ietf.org/rfc/rfc2962.txt`

[10] J. C. Han, W. Hyun, S. O. Park, I. J. Lee, M. Y. Huh, and S. G. Kang, *An Application Level Gateway for Traversal of SIP Transaction through NATs,* in Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference, vol. 3, Phoenix Park, Gangwon-Do, Republic of Korea, Feb. 2006.

[11] T.-C. Huang, S. Zeadally, N. Chilamkurti, and C.-K. Shieh, *A Programmable Network Address Translator: Design, Implementation, and Performance,* ACM Transactions on Internet Technology, vol. 10, no. 1, pp. 1–37, 2010.
[12] A. Pashalidis and H. Tschofenig, *GIST NAT Traversal,* IETF, Jul. 2007, Internet Draft draft-pashalidis-nsis-gimps-nattraversal-05, `http://tools.ietf.org/id/draft-pashalidis-nsis-gimps-nattraversal`.
[13] ———, *GIST Legacy NAT Traversal,* `http://tools.ietf.org/id/draft-pashalidis-nsis-gist-legacynats`, IETF, Jul. 2007, Internet Draft draft-pashalidis-nsis-gist-legacynat-02.
[14] P. McHardy, H. Welte, J. Kadlecsik, M. Josefsson, Y. Kozakai, and P. N. Ayuso, *Firewalling, NAT, and Packet Mangling for Linux,* Jun. 2010. [Online]. Available: `http://www.netfilter.org/`
[15] J. Salim, H. Khosravi, A. Kleen, and A. Kuznetsov, *Linux Netlink as an IP Services Protocol,* RFC 3549 (Informational), Internet Engineering Task Force, Jul. 2003. [Online]. Available: `http://www.ietf.org/rfc/rfc3549.txt`
[16] Institute of Telematics, *NSIS-ka—A free C++ implementation of NSIS protocols,* Dec. 2010. [Online]. Available: `http://nsis-ka.org/`

Edited by: Jemal H. Abawajy, Mukaddim Pathan, Al-Sakib Khan Pathan, and Mustafizur Rahman
Received: October 11th, 2010
Accepted: November 10th, 2010

Scalable Computing: Practice and Experience
Volume 11, Number 4, pp. 345–358. http://www.scpe.org
ISSN 1895-1767

AN EFFICIENT AND USER PRIVACY-PRESERVING ROUTING PROTOCOL FOR WIRELESS MESH NETWORKS

JAYDIP SEN*

Abstract. Wireless mesh networks (WMNs) have emerged as a key technology for next generation wireless broadband networks showing rapid progress and inspiring numerous compelling applications. A WMN comprises of a set of mesh routers (MRs) and mesh clients (MCs), where MRs are connected to the Internet backbone through the Internet gateways (IGWs). The MCs are wireless devices and communicate among themselves over possibly multi-hop paths with or without the involvement of MRs. User privacy and security have been primary concerns in WMNs due to their peer-to-peer network topology, shared wireless medium, stringent resource constraints, and highly dynamic environment. Moreover, to support real-time applications, WMNs must also be equipped with robust, reliable and efficient routing protocols so as to minimize the end-to-end latency. Design of a secure and efficient routing protocol for WMNs, therefore, is of paramount importance. In this paper, we propose an efficient and reliable routing protocol that also provides user anonymity in WMNs. The protocol is based on an accurate estimation of the available bandwidth in the wireless links and a robust estimation of the end-to-end delay in a routing path, and minimization of control message overhead. The user anonymity, authentication and data privacy is achieved by application of a novel protocol that is based on Rivest's ring signature scheme. Simulations carried out on the proposed protocol demonstrate that it is more efficient than some of the existing routing protocols.

Key words: wireless mesh network, user anonymity, bandwidth estimation, end-to-end delay, Rivest ring signature scheme, routing

1. Introduction. Wireless mesh networking has emerged as a promising concept to meet the challenges in next-generation wireless networks such as providing flexible, adaptive, and reconfigurable architecture while offering cost-effective solutions to service providers. WMNs are multi-hop wireless networks formed by mesh routers (which form a wireless mesh backbone) and mesh clients. The mesh routers provide a rich radio mesh connectivity which significantly reduces the up-front deployment cost of the network. Mesh routers are typically stationary and do not have power constraints. However, the clients are mobile and energy-constrained. Some mesh routers are designated as gateway routers which are connected to the Internet through a wired backbone. A gateway router provides access to conventional clients and interconnects ad hoc, sensor, cellular, and other networks to the Internet. A mesh network can provide multi-hop communication paths between wireless clients, thereby serving as a community network, or can provide multi-hop paths between the client and the gateway router, thereby providing broadband Internet access to the clients.

As WMNs become an increasingly popular replacement technology for last-mile connectivity to the home networking, community and neighborhood networking, it is imperative to design an efficient resource management system for these networks. Routing is one of the most challenging issues in resource management for supporting real-time applications with stringent QoS requirements. However, most of the existing routing protocols for WMNs are extensions of protocols originally designed for *mobile ad hoc networks* (MANETs) and thus they perform sub-optimally.

This paper presents an efficient and secure routing protocol for WMNs that is able to handle stringent QoS requirements of real-time applications while providing user privacy in a secure way. It involves a very low control overhead and hence provides a high network throughput when the number of data sources in the network is large. While issues such as reduction of control overhead of routing and enhancement of network throughput have been addressed for WMNs in [1], the protocol proposed in this paper is more efficient than those schemes as observed in the simulation results.

The key contributions of the paper are as follows: (i) It exploits network topological information to increase the efficiency of route discovery process and uses *multi-point relay* (MPR) nodes and *circular routing* (discussed in Section 4) to enhance the network throughput by reducing the control overhead. (ii) It computes a reliable link quality estimator and utilizes it in route selection. (iii) It provides a framework for reliable and robust estimation of available bandwidth and end-to-end delay in a routing path so that flow admission with guaranteed QoS for applications can be made. It also ensures that the number of retransmission required is minimized. (iv) It provides a simple mechanism to identify selfish nodes who consume network resources but do not cooperate

*Innovation Lab, Tata Consultancy Services Ltd, Bengal Intelligent Park, Salt Lake Electronics Complex, Kolkata — 700091, INDIA. (jaydip.sen@tcs.com). Questions, comments, or corrections to this document may be directed to that email address.

with other nodes in forwarding packets for others. (v) It presents a novel user anonymization scheme that enables secure authentication of the users while protecting their privacy.

The rest of this paper is organized as follows. Section 2 describes related work on routing in WMNs. Section 3 discusses some important challenges in routing in WMNs. Section 4 describes the details of the proposed routing protocol. Simulation results are presented in Section 5. Finally, Section 6 concludes the paper while highlighting some future scope of work.

2. Related Work. Although significant amount of work has been done on routing in MANETs, very little work has been done for WMNs. Most of the routing protocols for MANETs such as AODV and DSR use hop-count as the routing metric. However, this is approach not well-suited for WMNs. The basic idea in minimizing the hop-count is that it reduces delay and maximizes the throughput. But the assumption here is that the links in the path are either perfect or do not work at all, and all links are of equal bandwidth. A routing scheme that uses the hop-count metric does not take link quality into consideration. A minimum hop-count path has, on the average, longer links between the nodes present in the path compared to a higher hop-count path. This reduces the signal strength received by the nodes in that path and thereby increases the loss ratio at each link [2]. Hence, it is always possible that a two-hop path with a good link quality provides higher throughput than a one-hop path with a poor link quality. Moreover, wireless links usually have asymmetric loss rate [3]. Hence, new routing metrics based on link quality are proposed such *expected transmission count* (ETX),*per-hop round-trip time* (RTT), and per-hop packet pair.

Different approaches have been suggested by researchers for designing routing protocols for WMNs. In [4], a QoS routing over OLSR protocol has been proposed that takes into account metrics such as bandwidth and delay where the source node proactively changes a flow's next hop in response to the change in available bandwidth on its path. In [5], the authors have proposed a *link quality source routing* (LQSR) protocol. It is based on DSR and uses ETX as the routing metric. A new routing protocol called *multi-radio link quality source routing* (MR-LQSR) is proposed in [6]. The process of neighbor node discovery and propagation of link metric are same as those in DSR. However, assignment of link weight and computation of the path weight is different. A QoS enabling routing algorithm for mesh-based wireless LAN architecture has been proposed in [7], where the wireless users form an ad hoc peer-to-peer network. The authors also have proposed a protocol for MANET called *ad hoc QoS on-demand routing* (AQOR) [8]. In [9], the authors have shown that if a *weighted cumulative expected transmission time* [5] is used in a link state routing protocol, it does not satisfy the *isotonicity* property of the routing protocol and leads to formation of routing loops. To avoid routing loops, an algorithm is proposed that uses *metric of interference and channel switching* (MIC) as the routing metric. The *MeshCluster* architecture [10] addresses important issues in WMNs such as auto-configuration of mesh and client nodes, routing and load balancing in the infrastructure. The routing is performed via AODV-ST, a protocol that proactively maintains spanning trees rooted at the gateways. The mobility of the clients is managed by DHCP protocol.

Some routing protocols for WMNs have been developed by extending the existing routing protocols for MANETS with gateway discovery functionality [11][12][13][14][15] [16] [17]. Since these protocols provide unicast routes, individual routes must be maintained between every mobile node and one of the gateways. Therefore, these protocols scale poorly to the number of nodes in the mesh network.

In [18], scalability to the number of mesh nodes is improved with the uses of location information. However, this kind of information is typically not available in scenarios where the mobile nodes in the mesh network are commodity laptops or hand-held devices.

A problem that is common to most of the routing protocols for MANETs and WMNs is that gateway announcements or gateway requests are prone to vanish due to route breaks, and the recovery procedure is often as expensive as establishing a new route. In contrast, [19] proposes an efficient mechanism to fix broken routes locally. Mosko et al. [20] propose to establish multiple non-disjoint paths for better performance, but again the established routes are unicast and this protocol is not scalable to the number of mesh nodes.

IN [21], a single-hop mesh network architecture has been proposed where mobile clients connect directly to the gateways. However, this approach requires a much higher mesh node density for a comparable wireless coverage. In [22], the authors have proposed an anycast routing (i. e. routing form any mobile node to any gateway in the network) protocol that is designed to scale to the network size and to be robust to node mobility.

In contrast to the above approaches, the proposed protocol performs an on-demand route discovery using multiple metrics like bandwidth, delay, and reliability of the links and provides a routing framework that can support high network throughput with a minimum control overhead.

3. Routing Challenges in WMNs. This section first presents the generic architecture of a WMN and then discusses some specific challenges in designing routing algorithms for such networks.

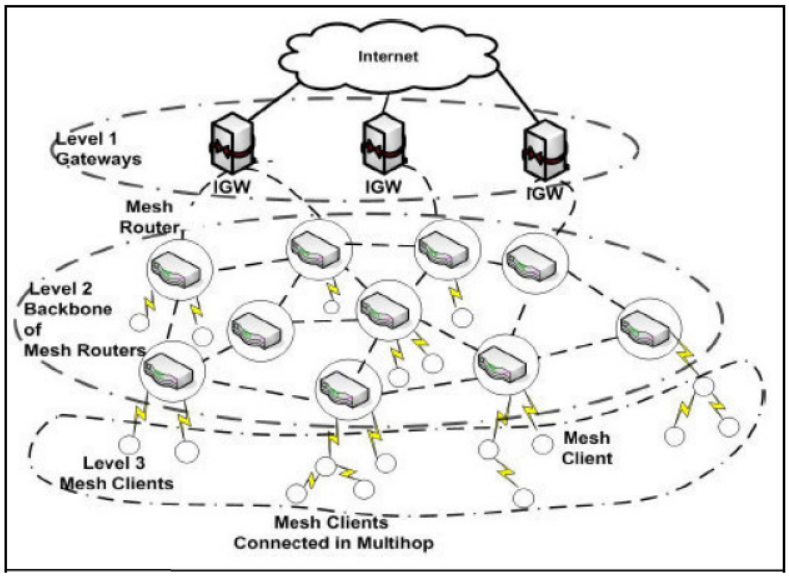


FIG. 3.1. *The three-tier architecture of a wireless mesh network (WMN)*

The architecture of a hierarchical WMN consists of three layers as shown in Fig. 3.1. At the top layers are the *Internet gateways* (IGWs) that are connected to the wired Internet. They form the backbone infrastructure for providing Internet connectivity to the elements in the second level. The entities at the second level are called wireless *mesh routers* (MRs) that eliminate the need for wired infrastructure at every MR and forward their traffic in a multi-hop fashion towards the IGW. At the lowest level are the *mesh clients* (MCs) which are the wireless devices of the users. Internet connectivity and peer-to-peer communications inside the mesh are two important applications for a WMN. Therefore design of an efficient and low-overhead routing protocol that avoids unreliable routes, and accurately estimate the end-to-end delay of a flow along the path from the source to the destination is a major challenge.

(i) *Measuring link reliability*: It has been observed that in wireless ad hoc networks nodes receiving broadcast messages introduce *communication gray zones* [23]. In such zones, data messages cannot be exchanged although the *hello* messages reach the neighbors. This leads to a disruption in communication among the nodes. Since the routing protocols such as AODV and WMR [7] rely on control packets like RREQ, these protocols are highly unreliable for estimating the quality of wireless links. Due to communication gray zone problem, nodes that are able to send and receive bidirectional RREQ packets sometimes cannot send/receive data packets at high rate. These fragile links trigger link repairs resulting in high control overhead.

(ii) *End-to-end delay estimation*: An important issue in a routing protocol is end-to-end delay estimation. Current protocols estimate end-to-end delay by measuring the time taken to route RREQ and RREP packets along the given path. However, RREQ and RREP are different from normal data packets and hence they are unlikely to experience the same levels of delay and loss as data packets. It has been observed through simulation that a RREP-based estimator overestimates while a hop-count-based estimator underestimates the actual delay experienced by the data packets [24]. The reason for the significant deviation of a RREP-based estimator from the actual end-to-end delay is interference of signals. The RREQ packets are flooded in the network resulting in a heavy burst of traffic. This heavy traffic causes inter-flow interference in the paths. The unicast data packets do not cause such events. Moreover, as a stream of packets traverse along a route, due to the broadcast nature of wireless links, different packets in the same flow interfere with each other resulting in per-packet delays. Since the control packets do not experience per-packet delay, the estimate based on control packet delay deviate widely from the actual delay experienced by the data packets.

(iii) *Reduction of control overhead*: Since the effective bandwidth of wireless channels vary continuously, reduction of control overhead is important in order to maximize throughput in the network. Reactive protocols

like AODV and DSR use flooding of RREQ packets for route discovery. This consumes a high proportion of the network bandwidth and reduces the effective throughput. An important challenge in designing a routing protocol for WMNs is to optimize the communication and computation overhead of the control messages so that the bandwidth of the wireless channels may be used for applications as efficiently as possible. Security and privacy issues bring another dimension of complexity. The goal of the protocol designer would be to design the security framework in such a way that it involves minimum computational and message overhead.

4. The Proposed Routing Protocol. The goal of the proposed routing protocol is to establish a route from a source node to a destination node that allows traffic flow within a guaranteed end-to-end latency and with a guaranteed available bandwidth in the wireless channel using the minimum control overhead. The salient features of the proposed algorithm in this paper are now discussed in the following subsections.

4.1. Estimating Reliability of Routing Paths. Every node estimates the reliability of each of its wireless links to its one-hop neighbor nodes. For computing the reliability of a link, the number of control packets that a node receives in a given time window is used as a base parameter. An *exponentially weighted moving average* (EWMA) method is used to update the link reliability estimate. If the percentage of control packets received by a node over a link in the last interval of measurement of link reliability is N_t, and if N_{t-1} is the historical value of the link reliability before the last measurement interval, $\alpha = 0.5$ is the weighting parameter, then the updated link reliability (R) is computed as in (4.1):

$$R = \alpha.N_t + (1 - \alpha).N_{t-1} \tag{4.1}$$

Every node maintains estimates the reliability of each of its links with its neighbors in a *link reliability table*. The reliability for an end-to-end routing path is computed by taking the average of the reliability values of all the links on the path. The use of path with the highest reliability reduces the overhead of route repair. The paths with reliability values less than 0.5 are never selected for routing.

4.2. Use of Network Topological Information in Route Discovery. The proposed protocol makes use of the knowledge of network topology by utilizing selective flooding of control messages in a portion of the network. In this way, broadcasting of control messages is avoided and thus the chances of network congestion and disruption to the flows in the network are reduced. If both the source and the destination are under the control of the same mesh router (refer Fig. 3.1), the flooding of the control messages are confined within the portion of the network served by the mesh router only. However, if the source and the destination are under different mesh routers, the control traffic is limited to the two mesh groups.

To further reduce the overhead of control message and enhance the reliability in routing, the nodes accept broadcast control messages from only those neighbors which have the link reliability value greater than 0.5 (i. e., on average 50% of the control packets sent from those nodes have been received by the node). This ensures that path with less reliability values are not discovered and therefore not considered for routing.

4.3. Estimation of End-to-End Delay in a Routing Path. For accurate estimation of end-to-end delay in a routing path, an approach similar to the one proposed in [24] has been taken. For addressing the issue of differential delays experienced by the control and the data packets, the proposed protocol makes use of some *probe packets* during the route discovery phase. When a source node receives RREP packets from the destination in response to its RREQ, it stores in a table, the records for all the RREP packets together with the path through which the packets have arrived at it. However, instead of randomly selecting a path to send probe packets to the destination as suggested in [24], the packets are sent along the path from which the RREP messages have arrived at the source first. This ensures that the probe packets are sent along the path which is likely to induce less end-to-end delay resulting in a better performance of the protocol as observed from the simulation results presented in Section 5. The probe packets are identical to data packets so far as their size, priority and flow rates are concerned. The objective of sending probe packets is to simulate the data flow and observe the delay characteristics in the routing path. Number of probe packets is kept limited to $2H$ for a path consisting of H hops to make a tradeoff between control overhead and measurement accuracy.

A destination node sets on a timer after it receives the first probe packet from the source node. The timer duration is based on the estimated time for receiving all the probe packets and is computed statistically. The destination computes the average delay experienced by all the probe packets it has received, and sends the computed value to the source node piggybacking it on a RREP message. If the computed value is within the

limit of tolerance of the application QoS, the source selects the route and sends data packets. If the delay exceeds the required limit, the source selects the next best path (based on the arrival of RREP packets) from its table and tries once again. Since the routing path is set up based on the probe packets rather than the naïve RREP packets, the proposed protocol has higher route establishment The proposed algorithm has higher setup time due to sending of the probe packets and selection of the best path based on the estimated end-to-end delay. However, since the selected paths have high end-to-end reliability, the delay and the control packet overhead are reduced because of minimal subsequent route breaks.

4.4. Use of Multi-Point Relay Nodes. The proposed routing protocol uses the *multi-point relay* (MPR) nodes like the *optimized link state routing* (OLSR) protocol [25] in order to reduce the control overhead in routing. In order to under the concept of MPR let us consider Fig. 3.1.

The control messages sent by an IGW, called GW_INFO messages are never flooded throughout the entire WMN; they are transmitted inside the corresponding subnet (under a particular MR) only. A GW_INFO message is processed by a node if and only if the neighbor which forwarded it has been validated as bi-directional (i. e., the sender is reachable by the receiver via the reverse link). The bi-directionality of a link is determined by appending the list of neighbors in the periodic *hello* messages. In this way, if a node finds itself in the list of neighbors advertised by its own neighbor, the link is considered bi-directional.

This additional list of neighbors in the hello messages is used to compute the MPR of a node. The objective of identifying the MPRs is to minimize control packet overhead. When MPRs are used, it is not necessary to send a message to all the nodes in a network when that message is required to reach all the nodes. If we visualize the WMN as a connected graph, the objective is to find the minimum subset of nodes which covers the whole graph. With a denser network, the benefits of using MPRs are more prominent. The protocol presented in this paper exploits the advantages of MPRs in order to reduce the control overheads of RREQ messages.

4.5. Estimating Available Network Bandwidth. In addition to computation of path reliability and use of MPRs, it is also necessary that the effective bandwidth in a routing path is reliably estimated. This is extremely important to support real-time applications since these applications require a guarantee for a minimum available bandwidth. In the proposed protocol, the available bandwidth in a wireless link is estimated using its end-to-end delay and loss of packets due to congestion. The packet-loss due to congestion in the link is estimated as follows.In addition to computation of path reliability and use of MPRs, it is also necessary that the effective bandwidth in a routing path is reliably estimated. This is extremely important to support real-time applications since these applications require a guarantee for a minimum available bandwidth. In the proposed protocol, the available bandwidth in a wireless link is estimated using its end-to-end delay and loss of packets due to congestion. The packet-loss due to congestion in the link is estimated as follows.

In a wireless link packet loss may happen due to two reasons: (i) loss due to faulty wireless links and (ii) loss due to network congestion. The *radio link control* (RLC) layer segments an IP packet into several RLC frames before transmission, and reassembles them into an IP packet at the receiver side. An IP packet loss occurs when an RLC frame belonging to an IP packet fails to be delivered. When this happens, the receiver knows the RLC frames reassembly has failed and the IP packet has been lost due to wireless error. Meanwhile, the sender detects *retransmission time out* (RTO) of the frame and discards all the RLC frames belonging to the IP packet. This enables the sender to compute packet drop rate in the wireless links. Moreover, using the sequence numbers of the IP packets received at the receiver, it possible to differentiate the packet loss due to link error and packet loss due to congestion [26]. For example, while receiving two incoming packets with sequence number i and $i + 2$, if the receiver finds an IP packet assembly failure in RLC layer, the packet with sequence number $i + 1$ is lost due to wireless channel. Once the packet loss ratio due to congestion ($P_{congestion}$) is estimated, the available bandwidth in the wireless link, *estrat*, is given by (4.2) as computed in [26]:

$$estrat = \frac{PacketSize}{(X + Y)} \tag{4.2}$$

X and Y in (4.2) are computed using (4.3) and (4.4) as follows:

$$X = RTT\sqrt{\frac{2P_{congestion}}{3}} \tag{4.3}$$

$$Y = RTO * Min(1, 3\sqrt{\frac{3P_{congestion}}{8}}P_{congestion}(1 + 32P^2_{congestion}) \tag{4.4}$$

In (4.2), RTT is the average round trip time for a control packet. RTO is the retransmission time out for a packet, and is computed using (4.5).

$$RTO = \overline{RTT} + K.\overline{RTT_{Var}} \tag{4.5}$$

$\overline{RTT}$ and $\overline{RTT_{Var}}$ in are the mean and variance respectively of RTTs and k is set to 4. This bandwidth estimator is employed to dynamically compute the available bandwidth in the wireless links on a routing path so that the guaranteed minimum bandwidth for the flow is always maintained throughout the application life-time.

4.6. Routing through the Fixed Network. In the proposed algorithm, the routing efficiency is further enhanced by occasional routing of packets through the fixed wired network backbone. Since the wired network backbone provides higher available bandwidth with more reliable links, it is advantageously exploited for intra-mesh message communication.

Since the IGWs (refer Fig. 3.1) periodically announce their presence in the network through beacon messages, every mesh client knows the hop count from itself to its selected gateway. In the proposed protocol, the RREQ messages include this hop count information. When the destination receives the RREQ, since it also knows its distance from its gateway, it checks whether it is better (in terms of number of hops) to route the packet through the wireless nodes (mesh) or through the fixed network.

In the proposed protocol, if a destination node finds that the better route is through the fixed network, the RREP message is routed through the wired network using the default route. Therefore, in such situations, the forward route is established between the source and the destination through the wired network, while the reverse route is set up through the WMN. This approach is known as *circular routing* [1]. This approach, improves the performance of bi-directional flows between a source and destination pair (as in a TCP connection) since the nodes in the forward and in the reverse routes are on node-disjoint paths and do not contend for access of the wireless medium.

4.7. Identification of Selfish Nodes. The proposed routing protocol also enforces cooperation among the nodes by identifying the selfish nodes in the network and isolating them. Selfishness is an inherent problem associated with any capacity-constrained multi-hop wireless networks like WMNs. A mesh router can behave selfishly owing to various reasons such as: (i) to obtain more wireless or Internet throughput, or (ii) to avoid path congestion. A selfish mesh router increases the packet delivery latency, and also increases the packet loss rate. A selfish node while utilizing the network resources for routing its own packet, avoids forwarding packets for others to conserve its energy. Identification of selfish nodes is therefore, a vital issue.

Several schemes proposed in the literature to mitigate the selfish behavior of nodes in wireless networks such as credit-based schemes, reputation-based schemes and game theory-based schemes [27]. However, to keep the overhead of computation and communication at the minimum, the proposed protocol employs a simple mechanism to discourage selfish behavior and encourage cooperation among nodes. To punish the selfish nodes, each node forwards packets to its neighbor node for routing only if the link reliability of that node is greater than a threshold (set at 0.5). Since the link reliability of a selfish node is 0, the packets arriving from this node will not be forwarded. Therefore, to keep its link reliability higher than the threshold, each node has to participate and cooperate in routing. The link reliability serves a dual purpose of enhancing reliability and enforcing node cooperation in the network.

4.8. User Anonymity and Privacy. As mentioned in Section 1, the proposed protocol has been augmented with a security module that provides user anonymity and privacy. An *authentication server* (AS) has been used in the network that authenticates the users in the WMN while preserving their privacy. To enable user authentication and anonymity, a novel protocol has been designed extending the improved ring signature authentication scheme in [28].

It is assumed that a symmetric encryption algorithm E exists such that for any key k, the function E_k is a permutation over b-bit strings. We also assume the existence of a family of *keyed combining functions* $C_{k,\nu}(y_1, y_2, \ldots\ldots, y_n)$ [29], and a publicly defined collision-resistant hash function $H(.)$ that maps arbitrary

inputs to strings of constant length which are used as keys for $C_{k,\nu}(y_1, y_2, \ldots\ldots, y_n)$. Every keyed combining function $C_{k,\nu}(y_1, y_2, \ldots\ldots, y_n)$ takes as input the key k, an initialization b-bit value ν, and arbitrary values y_1, $y_2 \ldots\ldots, y_n$. A user U_i who wants to generate a session key with the authentication server, uses a ring of n logged-on-users and performs the following steps.

Step 1: U_i chooses the following parameters: (i) a large prime p_i such that it is hard to compute the discrete logarithms in $GF(p_i)$, (ii) another large prime q_i such that q_i / (p_i -1), and (iii) a generator g_i in $GF(p_i)$ with order q_i.

Step 2: U_i chooses $x_A \in Z_{q_i}$ as his private key, and computes the public key $y_{A_i} = {g_i}^{x_{A_i}} \bmod p_i$.

Step 3: U_i defines a trap-door function:

$$f_i(\alpha, \beta) = \alpha.y_{A_i}^{\alpha mod q_i}.g_i^{\beta} mod p_i$$

Its inverse function $f_i^{-1}(y)$ is defined as: $f_i^{-1} = (\alpha,\ \beta)$, where α and β are computed as in (4.6), (4.7), and (4.8). In these equations, K is a random integer Z_{q_i}).

$$\alpha = y_{A_i}.g_i^{-K.(g_i^k mod q_i)} mod p_i \tag{4.6}$$

$$\alpha^* = \alpha mod q_i \tag{4.7}$$

$$\beta = K.(g_i^K mod p_i) - x_{A_i}.\alpha^* mod q_i \tag{4.8}$$

U_i makes p_i, q_i, g_i and y_{A_i} public, and keeps x_{A_i} as secret.

The *authentication server* (AS) chooses: (i) a large prime p such that it is hard to compute discrete logarithms in $GF(p)$, (ii) another large prime q such that q / (p -1), (iii) a generator g in $GF(p)$ with order q, (iv) a random integer x_B from Z_q as its private key. The AS computes its public key $y_B = g^{x_B} \bmod p$ and publishes (y_B,p,q,g).

Anonymous authenticated key exchange: The key-exchange is initiated by the user U_i and invloves three rounds to computes a secret session key between U_i and AS. The operations in these three rounds are as follows:

Round 1: When U_i wants to generate a session key on the behalf of n ring users U_1, U_2,$\ldots\ldots U_n$ where, ($1 \leq i \leq n$), U_i does the following:

(i) U_i chooses two random integers x_1, $x_A \in Z_q^*$ and computes the following: $R = g^{X_1} mod\ p$, $Q = y_B^{X_1} mod\ p\ mod\ q$, $X = g^{x_A} mod\ p$ and $l = H(X, Q, V, y_B, I)$.

(ii) U_i chooses a pair of values (α_t, β_t) for every other ring member U_t, $(1 \leq t \leq n, t \neq k)$ in a pseudorandom way, and computes $y_t = f_t(\alpha_t, \beta_t) mod\ p_t$.

(iii) U_i randomly chooses a b-bit initialization value ν, and finds the value of y_i from the equation: $C_{k,\nu}(y_1, y_2, \ldots\ldots, y_n) = \nu$.

(iv) U_i computes $(\alpha_i, \beta_i) = f_i^{-1}(y_i)$ by using the trap-door information of f_i. First, it chooses a random integer $K \in Z_{q_i}$, computes α_i using , and keeps K secret. It then computes α_i^* using (4.7), and finally computes β_i using (4.8) .

(v) $(U_1, U_2, \ldots\ldots U_n, \nu, V, R, (\alpha_1, \beta_1), (\alpha_2, \beta_2) \ldots., (\alpha_n, \beta_n))$ is the ring signature σ on X.

Round 2: AS does the following to recover and verify X from the signature σ.

(i) textitAS computes $Q = R^{X_B} mod\ p\ mod\ q$, recovers X using $X = V.g^Q mod\ p$, and hashes X, Q, V and y_b to recover l, where $l = H(X, Q, V, y_B, I)$.

(ii) textitAS computes $y_t = f_i(\alpha_t, \beta t) mod\ p_i$, for $t = 1, 2, \ldots., n$.

(iii) AS checks whether $C_{k,v}(y_1, y2, \ldots\ldots y_n) = \nu$. If it is true, AS accepts X as valid; otherwise, AS rejects X. If X is valid, AS chooses a random integer x_b from Z_q^*, and computes the following: If it is true, AS accepts X as valid; otherwise, AS rejects X. If X is valid, AS chooses a random integer xb from , and computes the following: $Y = g^{x_b} mod\ p$, $K_s = X^{x_b} mod\ p$, and $h = H(K_s, X, Y, I)$. AS sends $\{h, Y, I^{'}\}$ to U_i.

Round 3: U_i verifies whether $K_{S'}$ is from the server AS. For this purpose, U_i computes $K_S^{'} = Y^{x_a} mod\ p$, hashes K, X, Y to get h' using $h^{'} = H(K_S^{'}, X, Y, I)$. If $h^{'} ? = h$, U_i accepts K_S as the session key.

Security analysis: The key exchange scheme satisfies the following requirements.

(i) *User anonymity*: For a given signature X, the server can only be convinced that the ring signature is actually produced by at least one of the possible users. If the actual user does not reveal the seed K, the

server cannot determine the identity of the user. The strength of the anonymity depends on the security of the pseudorandom number generator. It is not possible to determine the identity of the actual user in a ring of size n with a probability greater than $1/n$. Since the values of k and ν are fixed in a ring signature, there are $(2^b)^{n-1}$ number of $(x_1, x_2, \ldots\ldots x_n)$ that satisfy the equation $C_{k,\nu}(y_1, y_2, \ldots\ldots y_n) = \nu$, and the probability of generation of each $(x_1, x_2, \ldots\ldots x_n)$ is the same. Therefore, the signature can't leak the identity information of the user.

(ii)*Mutual authentication*: In the proposed scheme, not only the server verifies the users, but the users can also verify the server. Because of the hardness of inverting the hash function $f(.)$, it is computationally infeasible for the attacker to determine (α_i, β_i) and hence it is infeasible for him to forge a signature. If the attacker wants to masquerade as the AS, he needs to compute $h = H(K_S, X, Y)$. He requires x_B in order to compute X. However, x_B is the private key of AS to which the attacker has no access. (iii)*Forward secrecy*: The forward secrecy of a scheme refers to its ability to defend leaking of its keys of previous sessions when an attacker is able to catch hold of the key of a particular session. The forward secrecy of a scheme enables it to prevent *replay attacks*. In the proposed scheme, since x_a and x_b are both selected randomly, the session key of each period has not relation to the other periods. Therefore, if the session key generated in the period j is leaked, the attacker can not get any information of the session keys generated before the period j. The proposed protocol is, therefore, resistant to replay attack.

5. Performance Evaluation. The proposed protocol has been implemented in the *Qualnet* network simulator, version 4.5 [30]. The reason for the choice of Qualnet is its ability of handle simulation of complex networks at multiple layers of the protocol stack.It is also suitable for simulation of large-scale dense networks like WMNs. The simulated network consists of 50 and 75 static nodes randomly distributed in the simulation area forming a dense WMN. The WMN topology is shown in Fig. 5.1 where 50 nodes are deployed in the network. out of these 50 nodes, 5 are MRs and remaining 45 are MCs. Each MR has 9 MCs associated with it. In 75 nodes deployment scenario (which is not shown), each of the 5 MRs has 14 MCs under it.

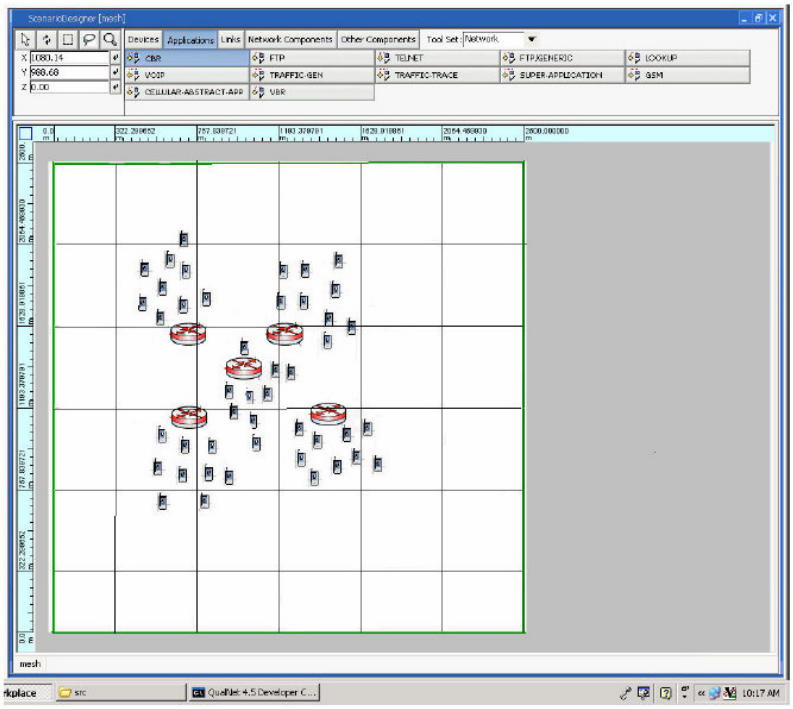


FIG. 5.1. *The simulated WMN topology in Qualnet network simulator*

For connectivity with the backbone Internet, 5 IGWs are placed at locations (100, 100), (100, 1400), (1400, 100) and (1400, 1400) and (700, 700) so as to provide uniform connectivity to backbone Internet with the WMN. The simulation parameters are presented in Table 5.1. The choice of the parameters is made similar to that in [1], and the performances of the two protocols (the proposed protocol and in one presented in [1]) are compared

with respect to two important metrics- control packet overhead and network throughput.The overhead due to security and privacy module in the proposed protocol is not considered for the purpose of comparison, since unlike the protocol presented in this paper, the protocol in [1] does not have any security feature. The security and privacy module involves well-known symmetric key encryption, and computations of efficient hash functions and message digests.The computation and communication overhead invloved in these operations are well-known and hence they are not studied in the simulation. The simulation results are presented for the modules which are mainly responsible for efficient operation of the protocol, such as reduction of control message overhead, reduction of end-to-end latency and increasing the network throughput.

TABLE 5.1
Simulation Parameters

Parameter	Values
Simulated netwoork area	1500m * 1500m
Propagation channel frequency	2.4 GHz
Raw channel bandwidth	2 Mbps
MAC protocol	802.11b
Simulation duration	900 s
Radio range of each node	250 m
Traffic type	CBR UDP
Packet size	512 bytes
Data rate in the network	32 Kbps
IGW hello packet broadcast interval	200 ms
No. of source nodes	15, 25, 35
Node mobility	None
Wireless fading model	None
IP queue scheduler	Strict priority
Propagation model	Two-ray ground
Wired network bandwidth	100 Mbps
Delay in wired links	11.8 ms

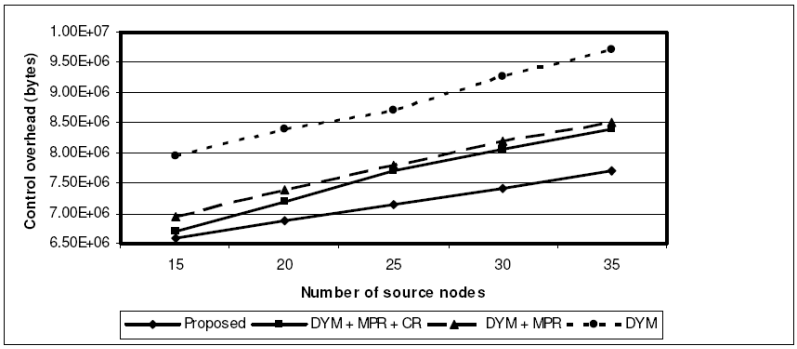


FIG. 5.2. *Control overhead (bytes) vs. number of data source nodes (50 nodes in the networks)*

5.1. Control Overhead. For studying the control overhead, four algorithms are considered. The DYNMOUM algorithm in [31], the DYNMOUM with MPR, in [1] DYNMOUM with MPR and *circular routing* (CR) [1] and the proposed algorithm compared with respect to their control overhead in routing. The number of data source nodes is varied from 15 to 35 and the control overhead in bytes is studied for 50 nodes and 75 nodes networks respectively. The results are presented in Fig. 5.2 and Fig. 5.3.

It may be easily observed that the proposed protocol has the least control overhead among all the four protocols. The reason for the less control overhead in the proposed protocol is the less number of route errors and route repairs due the reliable link quality estimation and bandwidth estimation technique used in the protocol, which were absent in the other three protocols. In addition, it exploits the advantages of using MPRs

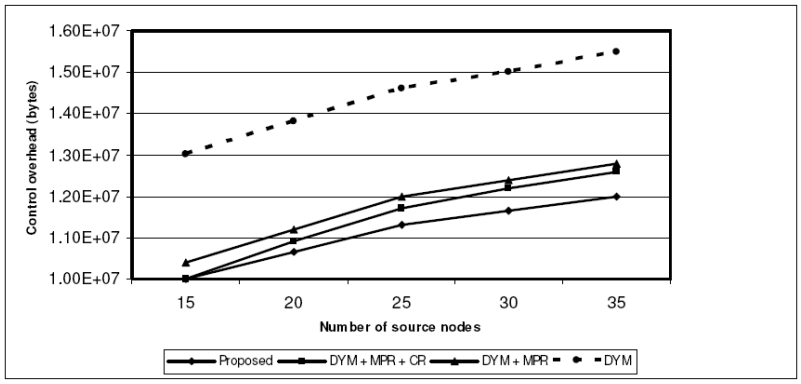


FIG. 5.3. *Control overhead (bytes) vs. number of data source nodes (75 nodes in the networks)*

and the circular routing. The MPRs reduce the overhead by controlled flooding and the circular routing reduces the overhead by routing some of the RREPs through the fixed network.

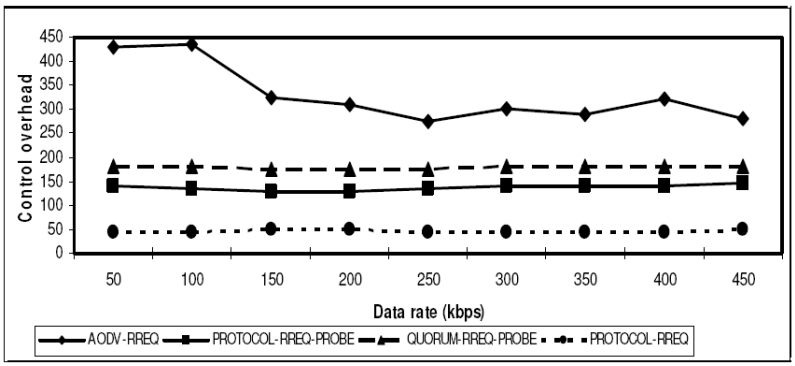


FIG. 5.4. *Control overhead (bytes) vs. data rate (kbps). Comparison of performance of the proposed protocol with the QUORUM protocol in [24]*

To further demonstrate the efficiency of the proposed protocol, the control packets overhead of the protocol is also compared with that of the protocol presented in [24]. For the purpose of comparison, some changes in the simulation parameters are made. The raw channel bandwidth is set at 11 Mbps. The application traffic is assumed to be CBR UDP. Each flow source is assumed to be sending a maximum of 10,000 packets to its destination node. Each flow is alive for 10 minutes and each simulation run is executed for 15 minutes. The link robustness value is computed once per second and the value of α in EWMA is taken as 0.5. All these parameters are set as per simulation environment presented in [24]. For the purpose of comparison, only the RREQ messages and the probe packets in the protocols are considered since these broadcast messages largely contribute to the control overhead. Fig. 5.4 shows the overhead due to RREQ packets in AODV and the proposed protocol for different data rates. The control overhead of the proposed protocol is first evaluated only with the RREQ packets and then with RREQ packets and the probe packets together. This also gives an idea of the additional overhead introduced due to the probe packets. It can be easily observed that the proposed protocol has very low overhead even with the probe packets when compared with the naive AODV protocol. It is also worth observing that the proposed protocol has about 20% less control overhead that the protocol proposed in [24] due to its robust bandwidth estimation of the wireless links.

Fig. 5.5 shows that AODV has a very high overhead due to control packets for increasing number of flows in the system. AODV always tries to establish routing paths based on minimum hop-counts. It does not consider the aspect of link reliability. This leads to frequent selection of unreliable links and consequent link-breaks and consequent re-discovery of routes resulting in high overhead of control packets. In contrast, the proposed protocol has a very limited control overhead since paths with higher link reliability only are selected for routing purpose. The algorithm proposed in [24] has similar performance as the proposed protocol in this case. The

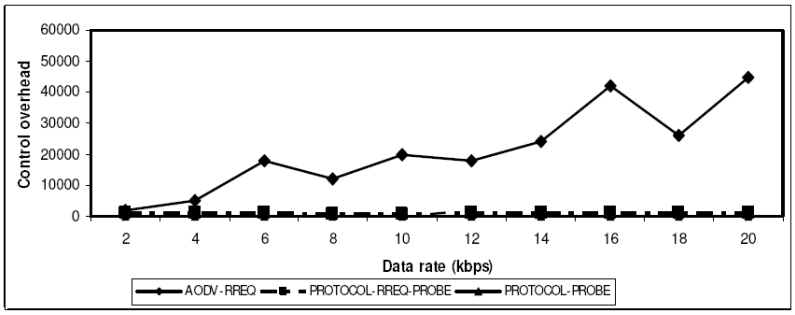


FIG. 5.5. *Control overhead vs. number flows in the network. Proposed protocol has much less control overhead than AODV protocol and has similar performance as that of QUORUM protocol in [24]*

results clearly demonstrate while the proposed protocol has similar trends as the protocol in [24] as far as control overhead with number of flows in the network are concerned, it has almost 20% reduction in control overhead for a particular value of network flow when compared with the protocol in [24].

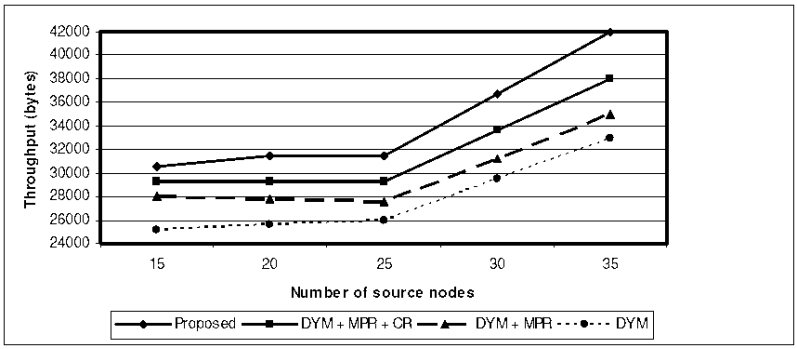


FIG. 5.6. *Network throughput in bits per second (BPS) vs. number of data source nodes (50 nodes in the network)*

5.2. Network Throughput. The performance of the protocol is also studied with respect to its ability to enhance network throughput. It may be intuitively clear that the reduction in control overhead should lead to a corresponding increase in the network throughput. Fig. 5.6 and Fig. 5.7 represent the data throughput in the network under varying number of source nodes with total number of nodes in the network being 50 and 75 respectively.

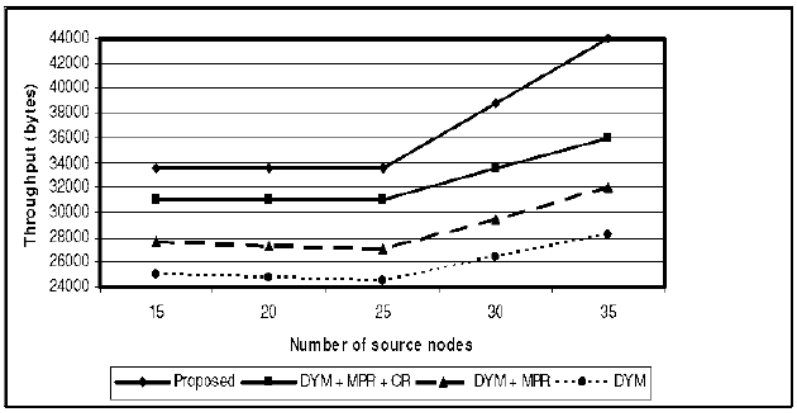


FIG. 5.7. *Network throughput in bits per second (BPS) vs. number of data source nodes (75 nodes in the network)*

It may be observed that the proposed protocol produces maximum network throughput among all the four protocol studied. There are various factors that contribute to the enhanced throughput with the proposed protocol. First, the throughput significantly increases when MPRs are used due to reduced number of collision in the wireless medium and fewer retransmissions. Moreover, circular routing improves the performance further due to use of fixed network that has higher effective bandwidth. Accurate estimate of link quality also contributes to higher throughput, since packets are always forwarded through the link that has the highest effective bandwidth. Finally, efficient bandwidth estimation ensures that there will be minimum packet retransmission.

5.3. End-to-End Delay Estimation. To demonstrate the effectiveness of the end-to-end delay estimation mechanism by probe packets mentioned in Section 4.3, delays estimated by naïve RREP approach and the probe packet approach are compared with the actual end-to-end delay in the routing paths. The records are observed for different flow rates with each flow having a minimum bandwidth requirement of $B_{min} = 50$ Kbps.

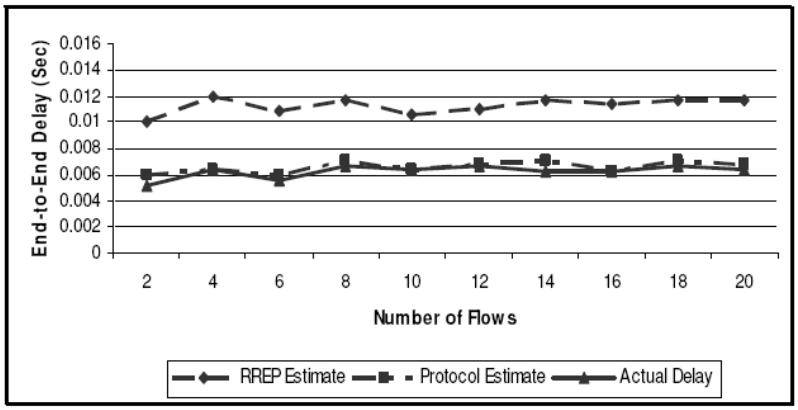


FIG. 5.8. *End-to-end delay estimation by different protocols for different number of nodes in the network*

Fig. 5.8 shows that the probe packet-based mechanism very accurately estimates the actual delay. The naive RREP approach is very poor in estimation of the delay as explained in Section 4.3.

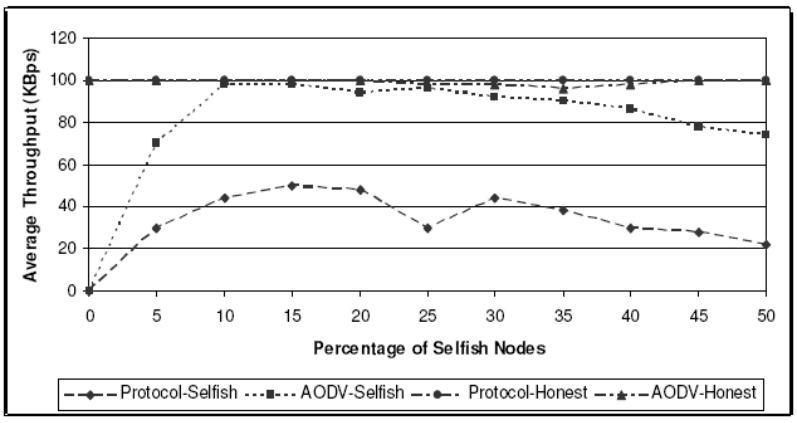


FIG. 5.9. *Average throughput for different protocols with varying number of selfish nodes*

5.4. Detection of Selfish Nodes. As mentioned in Section 4.7, the proposed protocol has the ability to detect selfish nodes which try to use network resources without contributing to the cooperative framework of routing. To evaluate its detection capability of selfish nodes, two types of flows are distinguished: *selfish* and *honest*. A flow is considered selfish if either its source or destination is a selfish node, otherwise the flow is considered to be honest. Some mesh clients are selected randomly and configured as selfish nodes. In each of the 20 run of the simulation, 10 flows of 50 kbps data rate are generated randomly. The throughput in the network is measured under four different types of flows: (i) honest flows using proposed protocol, (ii) selfish

flows (flows where selfish nodes are involved) using the proposed protocol, (iii) honest flows using simple AODV protocol [32], and (iv) selfish flows using AODV protocol.

It may be observed from Fig. 5.9 that AODV cannot restrict the traffic along the selfish flows. The selfish nodes can fully exploit the routing process to have their packets routed in the network. However, the proposed protocol reduces the flows along the selfish paths. In fact, the performance of AODV is not very much affected by presence of selfish nodes, since it never establishes routing path based on hello packets. Since the proposed protocol establishes route based on hello packets received form neighbors, its performance is affected by the presence of selfish nodes. However, its performance is not substantially affected, since most in most cases, these nodes are not allowed to participate in the routing, because of the low values of their link reliability. It may also be mentioned that the proposed protocol is able to maintain, on average, 35% more throughput in presence of selfish nodes when compared with the protocol proposed in [24]. The large difference is due to its ability to detect selfish nodes faster by its effective bandwidth estimation in the route where non-forwarding of packets is treated as packet drops due to congestion.

6. Conclusion and Future Work. WMNs have become an important focus area of research in recent years owing to their great promise in realizing numerous next-generation wireless services. Driven by the demand for rich and high-speed content access, recent research has focused on developing high performance communication protocols, while security and privacy issues have received relatively little attention. However, given the wireless and multi-hop nature of communication, WMNs are subject to a wide range of security and privacy threats. Accordingly, designing a high-performance, efficient, secure and user privacy-preserving routing protocol for WMN is a very challenging task due to involvement of a number of complex factors. This paper has presented a routing protocol that has very low control overhead and high network throughput when the number of source nodes in a WMN is large. By robust estimation wireless link quality and the available bandwidth in the wireless route and exploiting the benefits of using MPRs and circular routing technique, the protocol is able to sustain a high level of throughput with a low control overhead. The user privacy is protected by using a novel anonymized authentication protocol. Simulation results have shown the protocol is more efficient than some of the existing routing protocols for WMNs. Future work includes developing a security module with the routing protocol that will be able to defend against tunnelling attack [33], in which two malicious nodes advertise in such a way as if they have a very reliable ink between them. This is achieved by tunnelling AODV messages between the nodes. No security scheme exists so far that can detect this attack promptly and efficiently.

REFERENCES

[1] F. J. Ross and P. M. Ruiz, A Low Overhead Architecture for Infrastructure-Based Wireless Mesh Networks, *Proceedings of the First International Workshop on Wireless Mesh: moving towards Applications (WiMeshNets'06*, Waterloo, Ontario, Canada, August, 2006.

[2] D. D. Couto, D. Aguqayo, J. Bricket, and R. Morris, A High-Throughput Path Metric for Multi-Hop Wireless Routing, *Proceedings of the 9th ACM Annual International Conference on Mobile Computing and Networking (MOBICOM'09)*, pp. 134–146, September, 2003.

[3] D. Aguqayo, J. Bricket, S. Biswas, G. Judd, and R. Morris, Link-Level Measurements from an 802.11b Mesh Network, *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'04)*, pp. 121–132, 2004.

[4] H. Badis, I. Gawedzki, and K. Al Agha, QoS Routing in Ad Hoc Networks Using QOLSR with No Need of Explicit Reservation, *Proceedings of the 60th IEEE Vehicular Technology Conference (VTC-Fall'04)*, Vol. 4, pp. 2654–2658, September, 2004.

[5] R. Draves, J. Padhye, and B. Zill, Comparison of Routing Metrics for Static Multi-Hop Wireless Networks, *Proceedings of ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'04*, pp. 133–144, 2004.

[6] R. Draves, J. Padhye, and B. Zill, Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks, *Proceedings of the 10th ACM Annual International Conference on Mobile Computing and Networking (MOBICOM'04)*, pp. 114- 122, 2004.

[7] Q. Xue and A. Ganz, QoS Routing for Mesh-Based Wireless LANs, *International Journal of Wireless Information Networks*, Vol. 9, No. 3, pp. 179–190, 2002.

[8] Q. Xue and A. Ganz, Ad Hoc QoS On-Demand Routing (AQOR) in Mobile Ad Hoc Networks, *Journal of Parallel and Distributed Computing*, Vol. 63, No. 2, pp. 154–165, February, 2003.

[9] Y. Yang, J. Wang, and R. Kravets, Interference-Aware Load Balancing for Multi-Hop Wireless Networks, *Technical Report, TR: UIUCDCS-R-2005*, University of Illinois, Arbana Champaign, 2005.

[10] K. Ramachandran, M. Buddhikot, G. Chandranmenon, S. Miller, E. Belding-Royer, and K. Almeroth, On the Design and Implementation of Infrastructure Mesh Networks, *Proceedings of IEEE Workshop on Wireless Mesh Networks (WiMesh'05)*, IEEE Press, 2005.

[11] J.-C. Chen, S. Li, S.-H. Chan, and J.-Y. He, WIANI: Wireless Infrastructure and Ad-Hoc Network Integration, *Proceedings of IEEE International Conference on Communications (ICC'05)*, Vol. 5, pp. 3623–3627, May, 2005.
[12] R.-H. Hwang, C.-Y. Wang, C.-Y. Li, Y.-S. Chen, Mobile IPv6-Based Ad Hoc Networks: Its Development and Applications, *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 11, pp. 2161–2171, November, 2005.
[13] C. Ahlund and A. Zaslavsky, Extending Global IP Connectivity for Ad Hoc Networks, *Kluwer Telecommunication Systems, Modeling, Analysis, Design and Management*, Vol. 24, No. 2, pp. 221–250, 2003.
[14] P. Ratnachandani and R. Kravets, A Hybrid Approach to Internet Connectivity for Mobile Ad Hoc Networks, *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC'03)*, Vol. 3, pp. 1522–1527, New orleans, Los Angeles, USA, March, 2003.
[15] Y. Sun, E. Belding-Royer, and C. Perkins, Internet Connectivity for Ad Hoc Mobile Networks, *International Journal of Wireless Information Networks*, Vol. 9, No. 2, pp. 75–88, 2002.
[16] M. Michalak and T. Braun, Common Gateway Architecture for Mobile Ad-Hoc Networks, *Proceedings of the 2nd Annual Conference on Wireless On-Demand Network Systems and Services (WONS'05)*, pp. 70–75, Washington DC, USA, January, 2005.
[17] R. Baumann, VANET: Vehicular Ad Hoc Networks, *Master's Thesis*, ETH, Zurich, 2004.
[18] B.- N. Cheng, M. Yuksel, and S. Kalyanaraman, Orthogonal Rendezvous Routing Protocol for Wireless Mesh Networks, *IEEE / ACM Transactions on Networking*, Vol. 17, No. 2, pp. 542–555, April, 2009.
[19] M. Mosko and J. J. Garcia-Luna-Aceves, Ad Hoc Routing with Distributed Ordered Sequences, *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM'06)*, pp. 1–12, Barcelona, Spain, April, 2006.
[20] M. Mosko and J. J. Garcia-Luna-Aceves, Multipath Routing in Wireless Mesh Networks, *Proceedings of the First IEEE Workshop on Wireless Mesh Networks (WiMesh)*, Santa Clara, California, USA, September, 2005.
[21] H. Ju and I. Rubin, Backbone Topology Synthesis for Meshed Wireless LANs, *Proceedings of IEEE International Conference on Computer Communications (INFOCOM'06)*, Barcelona, Spain, April, 2006.
[22] R. Baumann, S. Heimlicher, V. Lenders, and M. May, HEAT: Scalable Routing in Wireless Mesh Networks Using Temperature Field, *Proceedings of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'07)*, pp. 1–9, Espoo, Finland, June, 2007.
[23] H. Lundgren, E. Nordstrom, and C. Tschudin, The Gray Zone Problem in IEEE 802.11b Based Ad Hoc Networks, *ACM SIGMOBILE Mobile Computing and Communications Review, (MC2R)*, Vol. 6, No. 3, pp. 104–105, July, 2002.
[24] V. Kone, S. Das, B. Y. Zhao, and H. Zhang, QUORUM: Quality of Service in Wireless Mesh Networks, *Journal of Mobile Networks and Applications*, Vol. 12, No. 5–6, pp. 358–369, 2007.
[25] T. Clausen and P. Jacquet, Optimized Link State Routing Protocol, *IETF RFC 3626*, 2003.
[26] F. Yang, Q. Zhang, W. Zhu, and Y.- Q. Zhang, End-to-End TCP-Friendly Streaming Protocol and Bit Allocation for Scalable Video over Wireless Internet, *IEEE Journal of Selected Areas in Communications*, Vol. 22, No. 4, pp. 777–790, May, 2004.
[27] L. Santhanam, B. Xie, and D. Agrawal, Selfishness in Mesh Networks: Wired Multi-Hop MANETS, *IEEE Journal of Wireless Communications*, Vol. 15, No. 4, pp. 16–23, August, 2008.
[28] T. Cao, D. Lin, and R. Xue, Improved Ring Authenticated Encryption Scheme, *Proceedings of the 10th Joint International Computer Conference*, pp. 341–346, Kunming, China, 2004.
[29] R. Rivest, A. Shamir, and Y. Tauman, How to Leak a Secret, *Advances in Cryptology, ASIACRYPT, LNCS*, Vol. 2248, pp. 552–565, Springer, Heidelberg, 2001.
[30] Network Simulator Qualnet, URL: `http://www.scalable-networks.com`
[31] F. Ross, Dymoum Implementation, URL: http://masimum.dif.um.es/?Software:DYMOUM.
[32] C. E. Perkins and E. M. Royer, Ad-Hoc On-Demand Distance Vector Routing, *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systes and Applications (WMCSA'99)*, pp. 90–100, New Orleans, Los Angeles, USA, February, 1999.
[33] C. Li, Z. Wang, and C. Yang, Secure Routing for Wireless Mesh Networks, *International Journal of Network Security*, Vol. 13, No. 2, pp. 1–12, September, 2011.

Edited by: Jemal H. Abawajy, Mukaddim Pathan, Al-Sakib Khan Pathan, and Mustafizur Rahman
Received: October 10th, 2010
Accepted: November 10th, 2010

Scalable Computing: Practice and Experience
Volume 11, Number 4, pp. 359–371. http://www.scpe.org
ISSN 1895-1767

ERCTP: END-TO-END RELIABLE AND CONGESTION AWARE TRANSPORT LAYER PROTOCOL FOR HETEROGENEOUS WSN*

ATIF SHARIF, VIDYASAGAR M. POTDAR AND A. J. D RATHNAYAKA†

Abstract. Other than hardware optimization, the communication protocols for heterogeneous Wireless Sensor Network (WSN) are currently playing an important role for achieving the longevity of the network. Among other layers of heterogeneous WSN communication protocol stack, researchers are putting efforts in developing transport layer protocols in order to avoid congestion in WSN and provide data or application level reliability support thereby ensuring the QoS objectives of the heterogeneous WSN application. In this paper we have envisaged a light weight transport protocol design, End-to-End Reliable and Congestion Aware Transport Layer Protocol (ERCTP), which achieves high data reliability by the introduction of distributed memory concept within network and minimum packet drop due to congestion by the effective implementation of congestion detection and rate adjustment scheme that uses stochastic control framework. The proposed scheme is evaluated extensively against the TCP-Westwood+ (TCP-WW+), TCP-Westwood (TCP-WW), TCPNewReno(TCP-NR), and TCPReno(TCP-R). The ERCTP has been tested for 24 mote topology and results reveal that the ERCTP effectively controls congestion and exhibits highest good throughput of 0.2941 Mbps, $\leq$ 100 msec average End-to-End (E-2-E) data packet latency for heterogeneous packet information, $>$ 99% data packet reliability and overall energy efficient behavior (lowest per packet communication cost) in comparison to TCP-WW+, TCP-WW, TCP-NR and TCP-R.

Key words: transport layer, IEEE 802.11, MAC layer, congestion, NACK, cross-layer design, WSN, reliability

1. Introduction.

Heterogeneous WSN is comprised of tiny embedded devices termed as "motes" that have inbuilt features for multitude sensing, processing and communicating information over wireless channel [7, 25, 18]. These devices sense information from the targeted area and communicate it wirelessly to remote base station or sink. There are a range of application scenarios for such ad-hoc heterogeneous networks[1] that range from modern health care to military applications by involving multiple disciplines of control, signals processing and embedded computing [7, 9, 21].

Transport layer of heterogeneous WSN, in general provide E-2-E network connection and the key objective of a transport protocol is to attain reliable data transport, while avoiding congestion and achieving high energy efficiency. Energy efficiency is considered to be the biggest concern for such networks and achieving high energy efficiency is of paramount importance for the longevity of such networks [26, 9]. To combat against these challenges energy efficient hardware design along with efficient communication protocol design for such networks is heavily investigated in the research community [7, 20].

In WSN, the transport layer protocol is responsible for reliably communicating the sensed information form source mote to the sink. In WSN the term "data reliability" is defined in terms of successful data delivery probability from the source mote to the sink mote. The main reason for packet drop includes congestion, collisions due to hidden motes, poor Signal-to-Noise Ratio (SNR) caused by bad channel conditions and link breakage due to mote failure. Undelivered data packets dropped as a result of any of the above listed conditions may lead to unwanted data retransmission that consume considerable amount of mote's energy budget. Although link reliability mechanisms like MAC layer protocol and Automatic Repeat Request (ARQ)[21] keeps the E-2-E packet loss ratio within acceptable limits, but mission critical applications like military, surveillance and industrial automation demand high E-2-E data reliability, which requires the use of highly reliable transport functionality for heterogeneous WSN. The reliability level is enhanced in WSN by the introduction of distributed intermediate storage or buffer motes. The buffer mote is not only meant for buffering the sensed packet information for some defined time interval (depending upon the link conditions, congestion severity etc) but also responsible for prioritized forwarding of information and possible data packet retransmissions as a result of packet drops in forward hops.

Congestion control is an important design parameter for heterogeneous WSN transport protocol. In WSN the term "congestion control" is defined as a way of achieving uncongested network that results in drop-less transmission of data packets from source to sink within the defined E-2-E packet latency threshold. Congestion occurs when:

*This paper is an enhanced version of the paper accepted in 24^{th} IEEE International Conference on Advanced Information Networking and Applications, AINA 2010, Australia, 20–13 April 2010.

†DEBII, Curtin University of Technology, Perth, WA. {a.sharif, v.potdar, a.rathnayaka}@curtin.edu.au wsn.debii.curtin.edu.au

[1]throughout the paper this term means network having scalar and multimedia sensing capability

1. Mote transmit more combined upstream traffic resulting in packet-arrival rate to exceed the packet processing rate at the receiving mote,
2. Mote's data throughput exceeds the link's available data threshold limit, and
3. Due to wireless link issues such as contention, interference, and blind mote problem.

Congestion causes packet drops and unnecessary packet retransmissions followed by significant network's energy depletion. The congested network regions also termed as "hot-spots" may lead to packet drops and retransmitting the lost packet information may result in considerable energy loss.

The congestion control feature of WSN transport protocol [15] is comprised of three functional modules i. e. Congestion Detection, Congestion Notification and Rate Adjustment. Congestion Detection module detects the level of congestion within the WSN network by either taking the E-2-E or H-b-H reliability information. The measured level of congestion then dictates the new rate adjustment for data communication. Upon detecting congestion, the Congestion Notification module informs the neighboring motes about the severity of the congestion in order to avoid extreme congested scenarios, which may lead to enormous data packet drop. Based on the notified congestion level the Rate Adjustment module then defines the new rate adjustment for the source motes in order to mitigate congestion. In most scenarios it is the sink mote that broadcasts this piece of information to the entire network. Given the importance of reliability and congestion control in WSN communication the contribution of this paper is four fold:

1. Firstly, we develop a stochastic control framework for congestion control.
2. Secondly, we design a new transport protocol that is capable of handling both reliability and congestion simultaneously for heterogeneous WSN.
3. Thirdly, we develop data prioritization module for assigning priorities to diverse sensors to allocate more bandwidth for critical applications, while achieving weighted fairness [2].
4. Finally, we provide simulation results that show the performance of the ERCTP in terms of system's good throughput, average E-2-E data packet latency in mili seconds (msec), average data packet drop and average per packet energy consumption for the entire data transmission monitored at source, intermediate/relay and sink motes.

The rest of the paper is organized as follows. After introduction the related work is presented in Section 2 followed by Section 3 and Section 4 where we describe the overview and details of the proposed transport protocol. Section 5 describes the simulation setup used for observing the behavior of the proposed protocol. In Section 6 we outline the simulation results and then we conclude the paper in Section 7.

2. Related Work. In this section we discuss the existing transport protocol schemes for WSN that targets both the congestion control and reliability aspects. The Table 2.1 includes the fundamental differences of these protocols based on the congestion control and reliability support.

In [7], the STCP (Sensor Transmission Control Protocol) solely depends on the buffer occupancy for congestion detection. It notifies congestion implicitly and the congestion avoidance is performed using rate adjustment as well as traffic redirection. STCP achieves controlled variable packet reliability for different flows in the network. STCP offers E-2 E reliability and the loss recovery mechanisms for continuous and event-driven dataflow utilizing NACK and ACK respectively. However, STCP does not incorporate multimedia flow of data, which is an imperative demand for heterogeneous applications.

DST (Delay Sensitive Transport Protocol) [13] detects the congestion considering mote delay estimation in addition to the buffer occupancy and uses implicit (Imp) congestion notification messages for informing the severity of the congestion. It avoids congestion by source rate adjustment policy. DST supports E-2-E event reliability but does not offer an explicit (Exp) loss recovery mechanism for lost packets.

Flush [16] detects congestion based on buffer size, route length and link quality and notifies the congestion implicitly to initiate rate adjustments. Flush ensures the Event reliability in upstream direction and enable E-2-E loss recovery mechanism by the Exp use of NACK signaling.

PORT (Price-Oriented Reliable Transport Protocol) [27] on the other hand detects congestion based on link loss rates and "Node price". The term "Node price" is defined as the number of transmission attempts made before a packet is successfully transferred to the sink. PORT uses Imp congestion notification messages for informing sink about the severity of the congestion and avoids congestion by using traffic redirection and source readjustment policy. It supports reliability in the reverse direction (from sink to source, E-2-E) only.

[2] is a data flow technique allowing different scheduling priorities to statistically multiplexed data flows

It does not support Exp mechanism for packet retrieval (for reliability assurance), as it relies heavily on its congestion control and optimal routing functionality to prevent congestion and packet drop.

CTCP (Collaborative Transport Control Protocol) [11] attains active congestion detection via transmission error losses and buffer overflow. It informs the congestion severity by the Exp use of congestion notification packets and avoids congestion using rate adjustment policy. CTCP achieves controlled variable packet reliability for different applications in network and accomplishes Hop by Hop (H-b-H) reliability by sending ACK and double ACK to each mote. CTCP uses considerable control signaling overhead for ensuring reliability.

ART (Asymmetric and Reliable Transport Protocol) [24] and RCRT (Rate Control Reliable Transport Protocol) [17] detect congestion based on the successful delivery of packets and use Imp congestion notification. ART identifies congestion if ACK is not received from essential dominating sensor motes within specific time interval. However, RCRT on the other hand identifies congestion based on time estimation to recover the loss. Both support E-2-E loss recovery mechanism by the Exp use of ACK and NACK control signaling. ART ensures bidirectional reliability by the use of ACK and NACK for event and query reliability, whereas RCRT uses NACK for ensuring packet reliability in the upstream direction and cumulative ACK to safely delete the memory. RT^2 (Real-Time and Reliable Transport) [14] achieves reliable H-b-H loss recovery between actors using Exp selective-acknowledgments (SACK) and also addresses cross-layer feature and intermediate mote feedback information to obtain the route failures, congestion notifications and transmission rate feedbacks. It uses information in the form of mote delay and buffer size for congestion detection and uses Imp means to notify congestion.

3. Proposed Protocol Overview:ERCTP. As discussed above majority of communication from source to sink in WSN is of H-b-H nature. Congestion, poor link quality, mote failure, collisions due to hidden motes etc. are some of the major factors of packet drop in WSN. As WSN is an energy scarce network, there is a price to be paid in terms of energy for every packet drop, which is given by:

$$E_{price} = N_h \times (E_{NACK} + E_{send})\ joules \tag{1}$$

where,

E_{price} = Total energy required for dropped packet retransmission (joules),

N_h = Number of hops between the sink and the source motes,

E_{NACK} = Energy consumed by a mote to transmit the NACK packet (joules), and

E_{Send} = Energy consumed by the mote to transmit the actual missing data packet (joules).

Hence in order to conserve the mote's energy budget (or network's energy budget) packet drop should be minimized to an extent such that the applications specific QoS is satisfied. It is therefore transport protocol's responsibility to control congestion, and if at all it occurs then notify the source motes and readjust their sending rate in order to mitigate congestion effectively. The heterogeneous WSN transport protocol also has an additional feature of distributed network storage which is invoked when packet recovery is desired. As WSN is comprised of energy scarce motes so loss of information due to packet drop caused by congested network condition is inevitable not only for certain application scenarios but also from the energy consumption point of view. In the next section we will describe our proposed transport protocol scheme, named ERCTP, that ensures Congestion detection, Congestion avoidance and Data loss recovery. The block diagram of the ERCTP is shown in Figure 3.1. The proposed scheme is comprised of the following functional modules: Congestion Control Module and Reliability Module.

3.1. Congestion Control Module. Congestion control module is composed of three separate sub-modules i. e. congestion detection module, congestion notification module and congestion avaoidance using source rate adjustment module. The basic functionality of these modules was discussed in Section 1 earlier and hence we have not repeated it again. For congestion control we envisage a scheme based on the stochastic control framework theory since we are considering the entire network communication in probabilistic terms. Stochastic control framework provides an effective mean to control congestion by observing the average E-2-E packet latency at the sink mote. Here the packet latency is given by:

$$T_{delay} = T_{pr} + T_Q + T_{PP} \tag{2}$$

where,

T_{delay} =average E-2-E packet latency (msec),

$T_{PP} = \tau$ =average packet processing time at a given mote(msec),

T_Q =average mote interface queue latency(msec), and

T_{pr} =average 1-hop propagation time (msec).

TABLE 2.1
Comparison for Transport Layer Protocols

Protocols	RCRT	FLUSH	STCP	PORT	ART	CTCP	RT^2	DST
Congestion Control	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Congestion Detection	Time to recover loss	Route length and link quality	Buffer size	Mote price and link-loss rates	ACK received to "Essential motes"	Transmission error packet loss and Buffer size	Mote delay/ Buffer size	Mote delay/ Buffer size
Congestion Notification	Implicit	Implicit	Implicit	Implicit	Implicit	Explicit	Implicit	Implicit
Congestion Avoidance	Rate Adjustment	Rate Adjustment	Traffic redirection / Rate adjustment	Traffic redirection / Rate adjustment	Reduce traffic of "Non-essential motes"	Rate Adjustment	Rate Adjustment	Rate Adjustment
Reliability Support	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Reliability Direction	Up	Up	Up	Up	Both	Up	Up	Up
Reliability Measure	Packet	Packet	Packet	Event information	Event	Packet	Packet	Event
Loss Recovery	E-2-E	E-2-E	E-2-E	-	E-2-E	H-b-H	H-b-H	E-2-E
Loss Notification	NACK & cumulative ACK	NACK	NACK/ ACK	-	ACK (events)/ NACK (queries)	ACK/ Double ACK	SACK	-

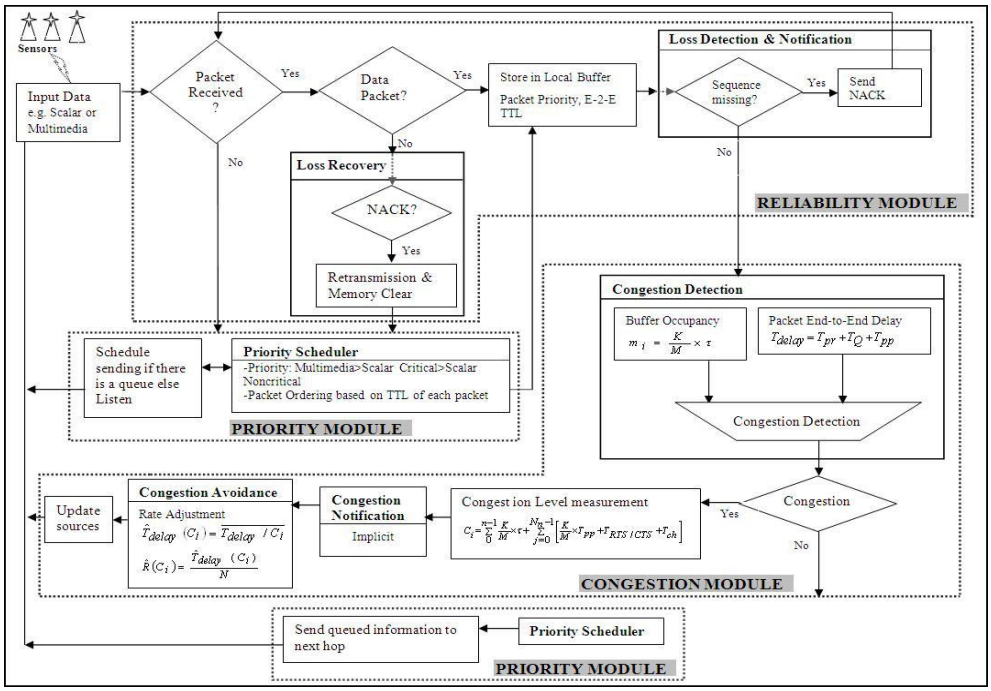


FIG. 3.1. *Proposed Transport protocol Scheme*

To detect congestion occurrence we define an index termed as the Congestion Index (C_i) in msec, which represents the congestion state of the network and is governed by the intermediate mote's buffer occupancy level and E-2-E propagation delay. Buffer Occupancy Index (m_i) for any intermediate mote is defined as 'ratio of occupied storage locations to the maximum available storage locations multiplied by the time required to process one data packet information'. The buffer mote uses the local queue for storing the packet and will read the information based on the prioritized nature of the data and the TTL information for that particular packet ID.

$$m_i = \frac{Total\ Free\ Space\ in\ the\ local\ storage}{Total\ Memmory\ space\ for\ local\ storage} \times \tau \tag{3}$$

or

$$m_{i-local} = \frac{K_{local}}{M_{local}} \times \tau_{local} \tag{4}$$

where,

τ = average local processing time for one data packet at any given intermediate mote.

For n number of intermediate storage motes each having congestion state m_i (for i^{th} mote), then C_i is mathematically given by:

$$C_i = \sum_{i=0}^{n-1} m_{i-local} + T_{E-2-E\ Pr\ delay} \tag{5}$$

If 'N_h' is the number of hops between source and sink mote, T_Q as the interface queue delay and per mote per link propagation delay (including the local queue latency) of T_{pr} then E-2-E propagation latency is given by:

$$T_{E-2-E\ Pr\ delay} = N_h \times T_{pr} \tag{6}$$

where,

$$T_{pr} = T_Q + T_{MAC} \tag{7}$$

$$T_Q = m_{i-interface} \times \tau_{inetrface} \tag{8}$$

and T_{MAC} is given by:

$$T_{MAC} = T_{RTS/CTS} + T_{ch} \tag{9}$$

where,

T_{MAC}= MAC access delay,

$T_{RTS/CTS}$ = Latency due to ongoing transmission as indicated by RTS/CTS, and

T_{ch} = Channel access delay.

So Eq.(5) i. e. congestion index, can now be expressed as:

$$C_i = \sum_{i=0}^{n-1} \frac{K_{local}}{M_{local}} \times \tau_{local} + \sum_{j=0}^{N_h-1} [\frac{K_{interface}}{M_{interface}} \times \tau + T_{RTS/CTS} + T_{ch}] \quad (10)$$

Eq. (10) represents the congestion index, which is helpful in deciding the future rate adjustments for the source motes. Now that we have calculated the congestion index, we move on to the next stage i. e. congestion detection.

3.1.1. Congestion Detection. In WSN, Congestion Detection is defined as "the means of detecting the congestion state of the network based on the link capacity and intermediate motes buffer occupancy". Eq. (10) is used for measuring the congestion state of the network which can then be used for the Mean-Square Estimation (MSE) of a new T_{delay} or new rate value for source motes (as we are taking the entire E-2-E scenario). Here we limit ourselves to C_i computation related to congestion only, avoiding any wait state for carrier sensing at the MAC level.

The joint density function[3] that relates the E-2-E data packet latency (T_{delay}) and Congestion index C_i (both are dependent variables) is given by $f_{T_{delay}C_i}(t_{delay},c_i)$, so the posteriori estimate for T_{delay}, $\hat{T}_{delay}(C_i)$ changes as C_i changes and is fixed for some constant z i. e. $C_i = z$. At the start we assume no congestion i. e. $C_i = C_0 = z_0$. Hence, the estimated error (J) [10] is given by:

$$J = \int_{-\infty}^{\infty} \int \left[T_{delay} - \hat{T}_{delay}(C_i)\right]^T \left[T_{delay} - \hat{T}_{delay}(C_i)\right] \, dT_{delay}.dC_i \quad (11)$$

or using the Baye's rule [10] we can write Eq.(11) as follows:

$$J' = \int_{-\infty}^{\infty} f_{C_i}(C_i) \int_{-\infty}^{\infty} \left[T_{delay} - \hat{T}_{delay}(C_i)\right]^T \left[T_{delay} - \hat{T}_{delay}(C_i)\right] f_{T_{delay|C_i}}(T_{delay}|C_i).dT_{delay}.dC_i \quad (12)$$

Here $f_{C_i}(C_i)$and the inner integral is non-negative so the value of conditional mean-square error J can be minimized by treating C_i as a constant and minimizing J for every such C_i. The value of J becomes:

$$J' = \int_{-\infty}^{\infty} \left[T_{delay} - \hat{T}_{delay}(C_i)\right]^T \left[T_{delay} - \hat{T}_{delay}(C_i)\right].f_{T_{delay|C_i}}(T_{delay}|C_i) \, dT_{delay}.dC_i \quad (13)$$

Now considering the conditional case and using the fact that $\hat{T}_{delay}(C_i)$changes as C_i changes and is fixed for $C_i = z$, where z is a constant it comes out to be:

$$\frac{\partial J'}{\partial T_{delay}(C_i)} = -\overline{2\hat{T}_{delay}/C_i} + 2\hat{T}_{delay}(C_i) = 0 \quad (14)$$

or finally MSE of the new $T_{delay}(C_i)$ is given by:

$$\hat{T}_{delay}(C_i) = \overline{T_{dealy}/C_i} \quad (15)$$

where for channel load threshold limit of λ_{Thresh} ,

$$\hat{T}_{delay}(C_i) \leq \lambda_{Thresh} \quad (16)$$

Hence the new estimated rate value can be expressed as:

$$\hat{R}(Ci) = \frac{\hat{T}_{delay}(Ci)}{N} \quad (17)$$

3.1.2. Congestion Notification. The purpose of the congestion notification module is to inform the neighboring motes about the severity of the congestion in order to avoid extreme congestion scenarios, which may lead to enormous data loss. Based on the computed $T_{delay}(C_i)$ value from Eq. (15), the sink periodically informs the source about the new rate value using Eq. (17) and the level of congestion C_i using Eq. (10).

3.1.3. Source Rate Adjustment. The purpose of the Source Rate Adjustment Module is to define the new transmission rate adjustment for child motes in order to mitigate congestion. The protocol monitors the instantaneous network statistics which helps sink to explicitly and periodically send the estimated value of rate adjustment to source motes, which is obtained based on congestion index calculation. Here this packet is generated separately for each source for every $'x'$ number of packets received from each source at the sink. This rate adjustment value indicates how much transmission rate should be increased or decreased for prospective data transmissions. For example, the rate adjustment figure would be either a negative value in case of foreseen congestion or a positive value if the channel is underutilized. Source motes then start sending the data packets with this new rate value $R(C_i)$, adjusted based on this feedback, in order to reduce the network's congested state and to avoid channel under utilization.

[3]function of two or more randomly distributed variables from which a single probability can be obtained indicating the range within which all the variables in the function falls.

3.2. Reliability Module. The purpose of the reliability module is to:

1. Retain the sensed information at designated intermediate motes for a defined interval of time,
2. Resend the sensed information upon receipt of NACK packet i. e. when packet loss occurs, and
3. Free up the motes memory upon receipt of ACK packet or after interval expiry.

To ensure reliability the ERCTP has introduced the concept of Distributed Memory Storage (DMS). In DMS the designated intermediate motes[4] are used to temporarily store packet information (thereby ensuring the transport level reliability), which can then be used for retransmission in case the sink fails to receive the designated packet information. In case the sink breaks its communication link with the immediate storage mote due to poor channel condition or due to storage motes energy depletion, then the NACK for the missing packet is propagated back to the next storage mote for corresponding information retrieval thus ensuring reliability of the proposed light weight transport protocol. The probability that a data packet is dropped is basically the sum of various probabilistic failures and is given by:

$$P_f = P_{lf} + P_{nf} + P_{hn} \tag{18}$$

where,

P_f = Total probability of data packet being dropped,

P_{lf} = Probability of packet drop due to link failure,

P_{nf} = Probability of packet drop due to mote failure, and

P_{hm} = Probability that a packet is dropped due to a hidden mote.

The idea of P_{lf}, P_{nf}, P_{hm} comes from the MAC/Wireless-Physical layer, by looking at motes energy, Bit Error Rate (BER) and packet collisions. So the total probability (success probability,$P_{success}$) with which data packet arrives at sink station is given by:

$$P_{success} = 1 - P_f \tag{19}$$

Hence there always exists a probability of packet failure and basically the index of P_f i.e α depicts the time to retain the data packets at the intermediate storage motes memory which is given by:

$$\frac{10 \times RTT}{\alpha}(msec) \tag{20}$$

where

α = storing time factor in this case, and

RTT = Round Trip Time in msec.

Based on the formulation, we are now going to simulate ERCTP by taking into consideration the possible real world effects of hidden mote transmissions and individual mote energy consumption for packet communication.

4. Simulation Setup. In this section we will describe the network topology and the parameters used for extensive testing of ERCTP for heterogeneous WSN. The aim of the simulation setup is to monitor the average system good throughput, average data packet drop, average E-2-E data packet latency and average per packet energy consumed by the source, relay and sink motes.

4.1. Performance Metrics. The ERCTP is evaluated against the following performance metrics:

1. Average Good throughput,
2. Average Data packet drop,
3. Average E-2-E Data packet latency, and
4. Average per packet energy consumed.

4.1.1. Average Good throughput . Average Good Throughput (AGT) is defined as the percentage ratio of sum of data sent by all sources to the data received by the sink. Mathematically it can be written as:

$$AGT = \frac{Total\ sent\ data}{Total\ received\ data} \times 100 \tag{21}$$

4.1.2. Average Data packet drop. Average packet drop (PD) is defined as the % of the data loss (difference between sent and received data) to the sent data. The main contributors of data loss are collisions at the receiving end in the presence of blind motes, congestion and link failure due to mote energy depletion. Mathematically it can be written as:

$$PD = \frac{Total\ sent\ data\ packets - Total\ received\ data\ packets}{Total\ sent\ data} \times 100 \tag{22}$$

[4]also called as "storage motes or buffer motes

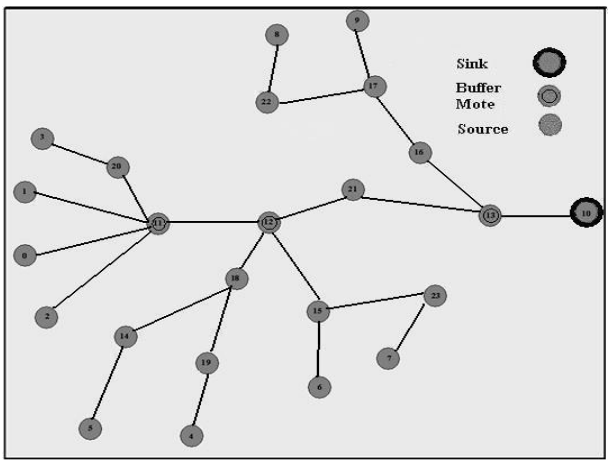


FIG. 4.1. *Network Topology*

4.1.3. Average E-2-E Data packet latency. E-2-E data packet latency $T_{delay}(E-2-E)$ is defined as the total time a packet would take from the source to sink (E-2-E). This includes all the possible delays as a result of queuing, retransmissions at the MAC layer, propagation delays and transfer time. Mathematically it can be written as:

$$T_{delay}(E-2-E) = (T_{rec} - T_{send}) \times 1000\, msec \tag{23}$$

4.1.4. Average per packet energy consumed. Average per packet energy E_{avg} consumed by source, relay or sink motes is defined as the % of total energy consumed in packet processing (that includes transmit, receive, relaying etc) to the total number of packets that it handles. Mathematically it can written as:

$$E_{avg} = \frac{Total\, Energy\, consumed}{Total\, packets} \times 100\, mJ \tag{24}$$

4.2. Network Topology. The network topology used for evaluating the ERCTP is shown in the Figure 4.1. The motes $0-9$ are considered as source motes while motes 11, 12 and 13 are considered as intermediate storage motes and mote 10 is considered to be a sink mote. The remaining network parameters, source nature and their priorities are listed in the Table 4.1 and Table 4.2 respectively.

Let us consider that the network is comprised of N motes placed in open/free space environment. The separation between the sending mote and receiving mote given by d_{tx-rx} is maintained to avoid the issue of hidden motes[5]. As outlined in Eq. 25 & Eq. 26 this distance is calculated as follows:

$$d_{int} \geq 1.778 * d_{tx-rx} \tag{25}$$

$$d_{tx-rx} \leq 0.5624 * d_{int} \tag{26}$$

where,

d_{int} = receiver interference region, and is given by:

$$d_{int} \geq 4\sqrt{SNR_{Thresh}} * d_{tx-rx} \tag{27}$$

$$SNR_{rec} = \frac{P_{rx}}{P_{int}} = \left(\frac{d_{int}}{d_{tx-rx}}\right)^4 \geq SNR_{Thresh} \tag{28}$$

For present simulations we have used a value of 10 for SNR_{Thresh}.

4.3. Comparative Transport Layer Protocol Standards . In this section we will briefly describe the transport layer protocols which we have used for the detailed evaluation against ERCTP and they are:

1. TCP-R,
2. TCP-NR,
3. TCP-WW, and
4. TCP-WW+.

ERCTP is evaluated against TCP-WW+, TCP-WW, TCP-NR and TCP-R. The source codes of these protocols were freely available but we were unable to find the codes for many recent protocols that we have mentioned in the related work section. TCP-NR is an extended version of TCP-R. TCP-NR improves retransmission during the

[5]for more detail refer to [22]

TABLE 4.1
Network Parameters

Parameter	Values
Frequency (Hz)	$914e^{+6}$
Transport Protocols	ERCTP, TCP-WW+ [5], TCP-WW [8], TCP-NR[3], TCP-R[4]
MAC	IEEE802.11 [1]
RX and CS Threshold (W)	$3.6252e^{-10}$ & $1.559e^{-11}$
Routing agent	Ad hoc On-Demand Distance Vector (AODV) [19]
CP Threshold	10
Ifqlen (packets)	50
Mote Initial power (W)	100
Mote Idle power (W)	$712e^{-6}$
Mote Rx power (W)	$35.28e^{-3}$
Mote Tx power (W)	$31.32e^{-3}$
Mote Sleep power (W)	0.001
Data Packet Size (Bytes)	512
x	10
α	$0 < \alpha \leq 1$
SNR_{Thresh}	10

TABLE 4.2
Source Priority

Source	Priority	Nature of Source
$1, 4, 7$	1	Multimedia
$3, 5, 6$	2	Scalar-critical
$0, 2, 8, 9$	3	Scalar-less critical

fast recovery phase of TCP-R. TCP-WW is a sender-side-only modification to TCP-NR and TCP-WW+ is an evolution of TCP-WW. The main difference in congestion avoidance feature of TCP-WW variants with respect to TCP-R variants is that Reno variants halves the congestion window after three duplicate acknowledgments or after a timeout where as Westwood variants attempt to select a slow start threshold and a congestion window considering the bandwidth estimation. In TCP-WW+, the original bandwidth estimation algorithm in TCP-WW has been modified to cope with ACK compression effect [12], as the TCP-WW fails to function properly in the presence of ACK compression.

5. Simulation Results and Discussions. Network Simulator (NS-2) [2] is used as a simulation platform for testing and comparing the functionality of the ERCTP with TCP-WW+, TCP-WW, TCP-NR and TCP-R. We have considered heterogeneous traffic flow including multimedia with a packet size of 512 bytes. The simulation has been performed five times and the average values of the results are plotted to ensure reliability in the given results.

5.1. Average Good Throughput Comparison. Good throughput comparison of the ERCTP with TCP variants like TCP-WW+, TCP-WW, TCP-NR and TCP-R is shown in Figure 5.1. From the Figure 5.1 we can see that the TCP-R and the ERCTP offer high system good throughput i. e. 0.2941 and 0.2927 Mbps in comparison to TCP-WW+, TCP-WW and TCP-NR whose good throughputs are 0.2874, 0.2902 and 0.2668 Mbps respectively. The ERCTP, which is basically a sink enabled E-2-E congestion control, utilizes the stochastically estimated values of T_{delay} (function of C_i) for defining the new transmission rate values for every

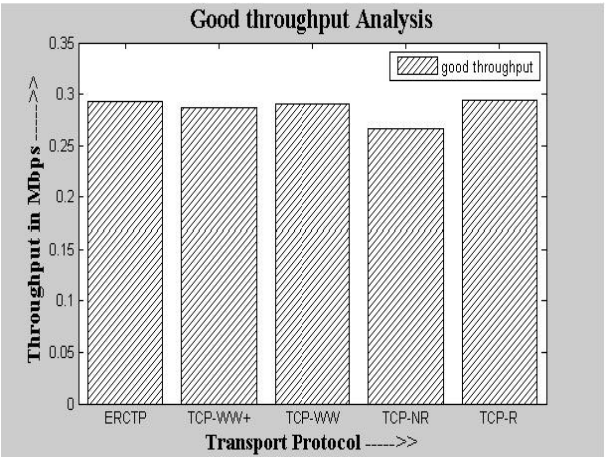
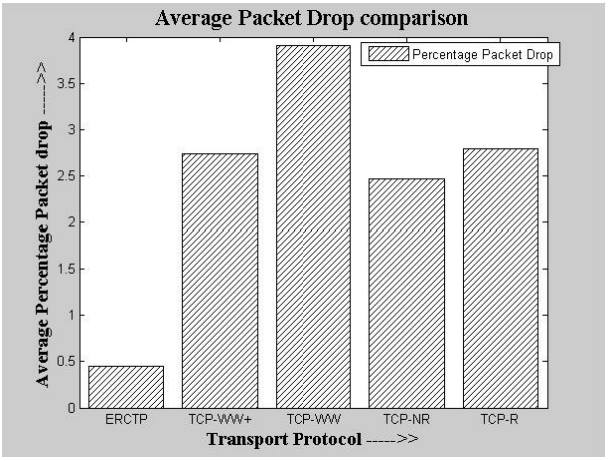


FIG. 5.1. *Average E-2-E Good Throughput Comparison*

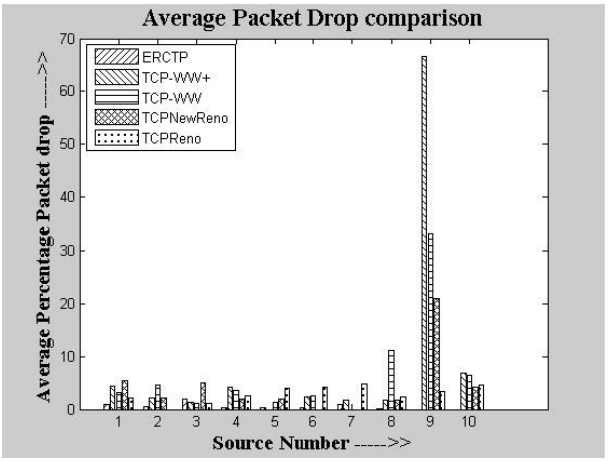
source and updates itself by the feedback parameter also called estimation error 'J'. Since the rate estimation involves the real time monitored system statistics in form of 'C_i' that is directly coupled to T_{delay} (the time gap between successive packet transmissions in other words the transmission frequency) which is why ERCTP shows high good throughput. By intelligently employing the system statistics in a stochastic control framework, not only prevents congestion but also the associated packet drops thus resulting in energy efficiency. In comparison, TCP-WW+ and TCP-WW (also called the sender-side variants of TCP-NR) relies on mining of the ACK control signaling for setting the congestion control parameters like slow-start threshold limit and congestion window which are then used in estimating the source transmission rate values. As mining of ACK stream involves the close monitoring of per packet ACK besides the actual data flow, it includes the E-2-E T_{delay} for data packet with an addition of the ACK reception latency (which is variable) at the source side, which would increase the time gap between the successive transmissions and hence results in a drop of system good throughput.

5.2. Average Data Packet Drop Comparison. Figure 5.2.*a* and 5.2.*b* shows the average E-2-E data packet drop comparison of the ERCTP with TCP variants like TCP-WW+, TCP-WW, TCP-NR and TCP-R. From these figures it is clear that among all protocols only the ERCTP offers high packet reliability for both the high and low priority packets. Of the total communication, the ERCTP exhibits approximately 0.46% packet drop in comparison to 2.74%, 3.91%, 2.47% and 2.79% packet drops for TCP-WW+, TCP-WW, TCP-NR and TCP-R respectively. Other than the use of storage motes for storing packets at intermediate buffer motes another major reason the ERCTP offers high packet reliability is its congestion control dependency over the real time/ instantaneous monitored network statistics that actually gives true idea of network congestion. In comparison the TCP variant like TCP-WW+ and TCP-WW avoids packet drops by the use of an additive increase/adaptive decrease (AIMD) paradigm to enhance the classic AIMD algorithm. When it experiences packet loss, it performs E-2-E bandwidth estimation (at the sender side) for future packet transmissions. As this sender based dynamic phenomenon of rate adjustment is entirely based on received ACK monitored statistics it could result in enormous packet drops before the new bandwidth estimation has taken place, which is why their average percentage data packet drop ranges between $2.74 - 3.91\%$ of the entire communication.

5.3. Average E-2-E Data Packet Latency Comparison. Figure 5.3 shows the E-2-E data packet latency comparison of the ERCTP with TCP-WW+, TCP-WW, TCP-NR and TCP-R. From this comparison it is obvious that the ERCTP, which uses real time instantaneous monitored statistics of the packet E-2-E latency coupled to the network congestion index, outperforms the rest. For the packet information coming from sources that uses TCP-WW+ and TCP-WW as transport agents, which probes the incoming ACKs for every sent data packet information for defining their rate plan based on the estimation of bandwidth, actually suffers from a large variable delay. For all TCP variants including TCP-NR and TCP-R, for large number of hops between sources and sink motes, this variable delay is in between $370 - 700$ msec E-2-E whereas for the ERCTP with similar number of hops the average E-2-E data packet latency is in between $70 - 85$ msec. For all data packet priorities the E-2-E data packet latency offered by ERCTP is < 100 msec while it is $\gg 100$ msec for TCP



(a) E-2-E percentage overall packet drop comparison



(b) Percentage packet drop comparison for various sources

FIG. 5.2. *Average Packet Drop Comparison*

variants and is highly variable depending upon the number of hops between the source and sink mote. Thus, these results clearly shows the use of intermediate storage mote's buffer occupancy and instantaneous network channel statistics, that defines the E-2-E congestion index of the entire network coupled to the rate adjustment scheme, for achieving acceptable E-2-E data packet latency confined to the QoS objectives of the application [23, 6].

5.4. Average Per Packet Energy Consumed Comparison. Figure 5.4 shows the per packet energy consumed (in mili Joules, mJ) by the source, relay and sink motes that uses various transport layer protocols. From the Figure 5.1 we notice that only TCP-R and ERCTP offer high throughput i. e. 0.2941 and 0.2927 Mbps while lowest of 0.2668 Mbps offered by TCP-NR. So if we start relating this system good throughput statistics with per packet energy consumption statistics as shown in Figure 6 then it is obvious that among all transport layer protocols only the ERCTP shows energy efficient behavior. For ERCTP, per packet energy consumed by the source, relay and sink motes is 0.3364, 0.4793 and 0.6942 mJ. Since TCP-NR has lowest throughput among all, its per packet energy consumption by the source motes is small i. e. 0.3126 mJ while highest being consumed by TCP-WW+, TCP-WW and TCP-R i. e. 0.4475, 0.4410 and 0.4394 mJ respectively. The reason why TCP-WW+ and TCP-WW offers high per packet energy cost is being justified by its control channel probing (mining of ACK control signaling for bandwidth estimation for future transmissions). Finally, in the case of TCP-R, which offers approximately similar system good throughput as offered by the ERCTP, has shown high per packet energy consumption behavior for source, relay and sink motes (i. e. on average per

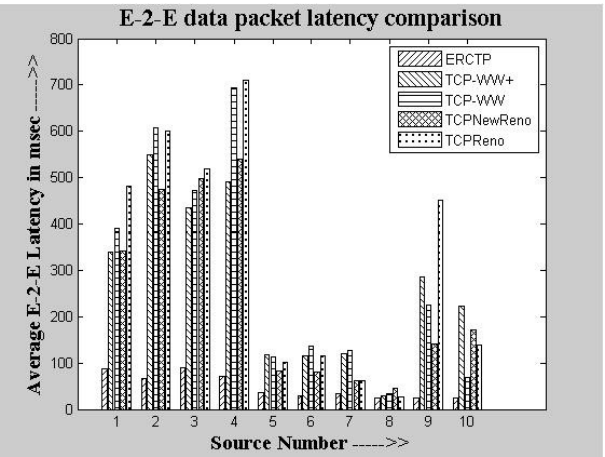


FIG. 5.3. *Average E-2-E Data Packet Latency Comparison*

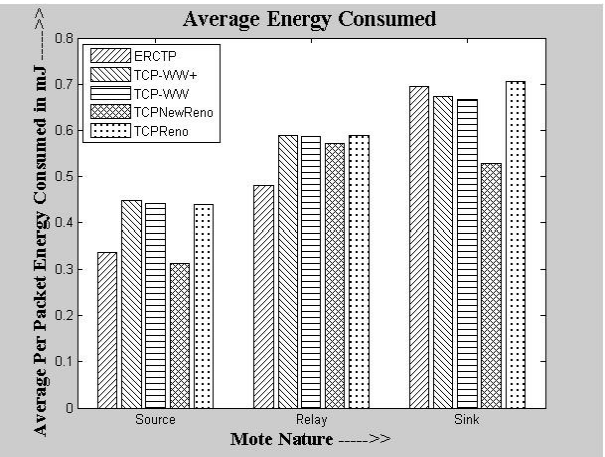


FIG. 5.4. *Average Per Packet Energy Consumed Comparison*

packet energy consumed by source, relay and sink motes are 0.1030, 0.1101 and 0.0114 mJ) and it is caused by frequent retransmissions and associated control signaling.

6. Discussions and Conclusions. Longevity of WSN is an important design challenge and was addressed in multiple disciplines of control, signal processing, and communication protocol design for WSN. It can be achieved by ensuring data reliability and congestion control, which are the two vital aspects of any transport protocol design for WSN. In this paper, we have surveyed various transport protocol schemes for WSN and proposed a light weight transport protocol, ERCTP, which is capable of detecting congestion, notifying it, source rate readjustment and data retrieval for any possible data loss that may happen either due to congestion or due to poor channel conditions i. e. packet is considered to be dropped for high BER, collisions at the receiver side due to transmissions from a hidden mote (hidden mote problem).

We extensively simulated the ERCTP against TCP-WW+, TCP-WW, TCPNewReno and TCPReno and the results reveal that considerable reduction in E-2-E data packet latency has been observed for the ERCTP around < 100 msec for heterogeneous packet information. Also for the ERCTP and TCP-R highest average good throughput is achieved i. e. 0.2927 and 0.2941 Mbps while effectively maintaining the $> 99\%$ achieved source priority (at sink) for various sources with only 3 buffer motes. For the entire communication ERCTP exhibits the energy efficient behavior in comparison to rest all and exhibits only $< 0.45\%$ packet drop which is minimum of all. The results reveal that the rate adaptation mechanism prevents major data packet drop if the stochastically distributed network conditions are known. Under uncongested/congested network conditions the ERCTP performs well in comparison to TCP variants.

In the next phase we will try to melt the prevalent or reciprocal functionalities of Transport layer and underlying MAC and Wireless-Physical layers, as we have seen that Transport layer functionality is mostly dependent on MAC and Wireless-Physical layers [22]. Cross-layering the transport with MAC and Wireless-Physical layers enables an idea of low level signal and channel details like received signal strength, packet ACK at MAC level, channel access time, channel loading, packet retransmission, intermediate queue occupancy, mote energy updates etc at the transport level, thereby helping in the design of more robust transport layer protocol for achieving high network throughput while simultaneously addressing the issues of congestion detection and data reliability.

REFERENCES

[1] *Ieee p802.11, main general info page.*
[2] *The network simulator:ns-2, http://www.isi.edu/nsnam/ns/index.html, 2008.*
[3] *Tcp new reno, http://www.faqs.org/rfcs/rfc3782.html, 2008.*
[4] *Tcp reno, http://tools.ietf.org/html/rfc1644, 2008.*
[5] *Tcp westwood+, http://c3lab.poliba.it/index.php/westwood, 2008.*
[6] *How delay and packet loss impact voice quality in voip,*, 2008.
[7] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, *A survey on wireless multimedia sensor networks*, Comput. Netw., 51 (2007), pp. 921–960.
[8] C. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, *Tcp westwood: end-to-end congestion control for wired/wireless networks*, Wirel. Netw., 8 (2002), pp. 467–479.
[9] S. Ci, H. Wang, and D. Wu, *A theoretical framework for quality-aware cross-layer optimized wireless multimedia communications*, Adv. MultiMedia, 2008 (2008), pp. 1–10.
[10] L. L. Frank, L. Xie, and D. Popa, *Optimal and Robust Estimation: With an Introduction to Stochastic Control Theory, Second Edition*, CRC Press, 2007, ch. 1-7, pp. 1–552.
[11] E. Giancoli, F. Jabour, and A. Pedroza, *Ctcp: Reliable transport control protocol for sensor networks*, in International Conference on Intelligent Sensors, Sensor Networks and Information Processing, ISSNIP, dec. 2008, pp. 493 –498.
[12] L. A. Grieco and S. Mascolo, *Tcp westwood and easy red to improve fairness in high-speed networks*, in PIHSN '02: Proceedings of the 7th IFIP/IEEE International Workshop on Protocols for High Speed Networks, London, UK, 2002, Springer-Verlag, pp. 130–146.
[13] V. Gungor and O. Akan, *Dst: delay sensitive transport in wireless sensor networks*, in International Symposium on Computer Networks, 2006, pp. 116 –122.
[14] V. C. Gungor, O. B. Akan, and I. F. Akyildiz, *A real-time and reliable transport (rt) 2 protocol for wireless sensor and actor networks*, IEEE/ACM Trans. Netw., 16 (2008), pp. 359–370.
[15] Y. Iyer, S. Gandham, and S. Venkatesan, *Stcp: a generic transport layer protocol for wireless sensor networks*, oct. 2005, pp. 449 – 454.
[16] S. Kim, R. Fonseca, P. Dutta, A. Tavakoli, D. Culler, P. Levis, S. Shenker, and I. Stoica, *Flush: a reliable bulk transport protocol for multihop wireless networks*, in SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems, New York, NY, USA, 2007, ACM, pp. 351–365.
[17] J. Paek and R. Govindan, *Rcrt: rate-controlled reliable transport for wireless sensor networks*, in SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems, New York, NY, USA, 2007, ACM, pp. 305–319.
[18] Q. Pang, V. W. S. Wong, and V. C. M. Leung, *Reliable data transport and congestion control in wireless sensor networks*, Int. J. Sen. Netw., 3 (2007), pp. 16–24.
[19] C. Perkins, E. Royer, and S. Das, *Ad hoc on-demand distance vector (aodv) routing rfc 3561, 2003.*, 2003.
[20] V. Potdar, A. Sharif, and E. Chang, *Wireless sensor networks: A survey*, in WAINA '09: Proceedings of the 2009 International Conference on Advanced Information Networking and Applications Workshops, Washington, DC, USA, 2009, IEEE Computer Society, pp. 636–641.
[21] Z. Rosberg, R. Liu, A. Dong, L. Tuan, and S. Jha, *Arq with implicit and explicit acks in wireless sensor networks*, nov. 2008, pp. 1 –6.
[22] A. Sharif, V. Potdar, and A. Rathnayaka, *Dependency of transport layer protocols on the ieee 802.11 and ieee 802.15.4 mac/phy layer protocols for wsn: A step towards cross-layer design*, Int J. of Busn. Data Comm. and Netw., IJDBCN, 3 (2010), pp. 1–30.
[23] E. D. Siegel, *Applications and infrastructure issues*, Wiley Computer Publishing John Wiley & sons.
[24] N. Tezcan and W. Wang, *Art; an asymmetric and reliable transport mechanism for wireless sensor networks*, Int. J. Sen. Netw., 2 (2007), pp. 188–200.
[25] C. Wang, K. Sohraby, B. Li, M. Daneshmand, and Y. Hu, *A survey of transport protocols for wireless sensor networks*, Network, IEEE, 20 (2006), pp. 34 – 40.
[26] N. Zhao and L. Sun, *Research on cross-layer frameworks design in wireless sensor networks*, mar. 2007, pp. 50a –50a.
[27] Y. Zhou, M. Lyu, J. Liu, and H. Wang, *Port: a price-oriented reliable transport protocol for wireless sensor networks*, nov. 2005, pp. 117 –126.

Edited by: Jemal H. Abawajy, Mukaddim Pathan, Al-Sakib Khan Pathan, and Mustafizur Rahman
Received: September 24th, 2010
Accepted: November 10th, 2010

Scalable Computing: Practice and Experience
Volume 11, Number 4, pp. 373–383. http://www.scpe.org
ISSN 1895-1767

IMPLEMENTING SELF-PROTECTION IN DISTRIBUTED GRID ENVIRONMENT*

INDERPREET CHOPRA† AND MANINDER SINGH‡

Abstract. Grids encourage the dynamic addition of resources that are not likely to be benefited from the security that static, centrally administered commercial firewall's or IDS's provide. On the contrary, Grids need to enforce their own security policies that are self automated, as the manual management techniques are time consuming, insecure and error prone. This paper presents a new self-protection model—SPM based upon few principles of genetic algorithm.

Key words: genetic algorithm, SPM, Snort, GT4

1. Introduction. The Grid is constantly growing and it is being used by more and more applications. However the grid services are increasing in complexity [4, 5] in modern era. These can vary over a large number of parameters: from the network (topology, speed and security policies), the resources (number of machines, processors and machine usage) to the scheduler (fork or more complex once) [3].

The complexity of today's distributed computing environment is such that the presence of bugs and security holes is unavoidable [2]. Before a user runs an application,he needs assurance that the machine has not been compromised which could subject his application from being stolen. When a user's job executes, the job may require confidential message passing services [9].Hence, security is a major concern for any IT infrastructure.

Overall,most of the problems stem from the fact that human administrators are unable to cope up with the amount of work required to secure the computing infrastructure properly in the age of Internet [12]. Autonomic systems and self-managed environments provide security policies establishing promising trust between the server and the client. Hence, autonomic computing, [7, 8] presented and advocated by IBM, suggests a desirable solution to this problem [6].

Self-Protection enables the system with an ability to secure itself against attacks i. e. detect illegal activities and trigger counter measures in order to stop them. It also helps to overpower the limitations of manual management of the system i. e. slow speed, increasing chances of errors and unmanageability by human administrators.

This paper proposes a Self-Protection Model (SPM)- an agent enabling autonomic computing providing a promising solution to the system management troubles caused by increased complexity of large scale distributed systems. SPM is based upon few implications of the genetic algorithm (GA). GAs are robust, inherently parallel, adaptable and suitable for dealing with the classification of rare classes. Moreover, due to its inherent parallelism, it offers a possibility to implement the system without requiring any additional resource. This new model adopts intelligent agents to dynamically organize system management with centralized control. At the system level each element contributes its capabilities on the functions of system management and cooperate with each other to implement autonomic computing for the grid system. Cooperative works are organized by dynamically associated relationships among autonomic elements (agents), including acquaintance, collaboration and notification.

This self-organized model is more suitable for the grid environment as it is characterized by distributed, open and dynamic properties.

2. Self-Protection. A self-protecting system helps to detect and identify hostile behavior and take autonomous actions to protect itself against intrusive behavior. The main goal of self-protection system is to defend grid environment against malicious intentional actions by scanning the suspicious activities and react accordingly without the users being awared that such protection is in process [2].

2.1. Current Security Vulnerabilities. Kind of vulnerabilities, against which we need self-protection are:

QoS Violation Grid service must provide the best service with respect to the service level agreement (SLA) signed between the user and the provider [15]. Different types of Qos violation attacks include:

*This work is done at Thapar University, India to automate the security process in grid computing.

†Research Scholar, Computer Science Department, Thapar University, Patiala, India (inderpreet@thapar.edu).
‡Associate Professor, Computer Science Department, Thapar university, Patiala, India (msingh@thapar.edu).

- *Dropping Packets:* In these types of vulnerabilities, attacker chooses to drop the packets flowing through the grid networks by taking control over some component in the grid network.
- *Delaying Packets:*In this type of attacks, attackers reduce the flow rate of the packets so as to reduce the effective QoS.

DoS attacks These continue to be the main threat. In such attacks, a set of attackers generates a huge traffic, saturating the victim's network, and causing significant damage. Sun Grid was brought to its knees by a distributed denial-of-service (DDOS) attack [19], necessitating an emergency login procedure change. While grid computing may very well revolutionize enterprise computing, the incident underscores the security risks that could prove quite harrowing for enterprises that rely on grid computing [17].These include:

- *TCP floods:* A stream of TCP packets with various flags set are sent to the victim IP address. The SYN, ACK, and RST flags are commonly used.
- *ICMP echo request/reply (e.g., ping floods):* A stream of ICMP packets are sent to a victim IP address.
- *UDP floods:* A stream of UDP packets are sent to the victim IP address.
- *Source IP address:* In some cases, a false source IP address, a method commonly called IP spoofing, is used to conceal the true source of a packet stream. In other cases, IP spoofing is used when packet streams are sent to one or more intermediate sites in order to cause responses to be sent toward a victim. The latter example is common for packet amplification attacks such as those based on IP directed broadcast packets (e.g., "smurf" or "fraggle").
- *Source/destination ports:* TCP and UDP based packet flooding attack tools sometimes alter source and/or destination port numbers to make reacting with packet filtering by service more difficult.
- *Other IP header values:*At the extreme, we have seen DoS attack tools that are designed to randomize most all IP header options for each packet in the stream, leaving just the destination IP address consistent between packets.

Insider Attacks Attacks from the inside carry the potential for significant damage that can reveal or even exceed the damage caused by external forces [13]. As an integral and trusted member of the organization, the perpetrator carries valid authorization and typically enjoys relatively unchallenged presence and movement within the organization's IT infrastructure. The attacks typically target specific information and exploit established entry points or obscure vulnerabilities. In many respects, insider attacks can be more difficult to detect than penetration attempts from the outside.

Remote to local(R2L) Attacks in which an unauthorized user is able to bypass normal authentication and execute commands on the target [9].

User to root (U2R) Attacks in which a user with login access is able to bypass normal authentication to gain the privileges of another user, usually root. This can also be the case when the user try to use more or some other resources assigned to him [9].

2.2. Related Work. Few of the self-protection tools/techniques available to handle the attacks on grid system are:

- **Firewall**: The firewall is configured to filter out the packets to enter inside the grid environment. Firewalls basically separate the protected network from the unprotected network [26] [27]. They screens and filters all connections coming from internet to the protected (corporate) network and vice versa, through a single concentrated security checkpoint.
- **Intrusion Detection System**: Intrusion Detection Systems add an early warning capability for the suspicious activity that mostly occurs before and during an attack. Intrusion detection systems (IDSs) often work as misuse detectors, where the packets in the monitored network are compared against a repository of signatures that define characteristics of an intrusion.
  Many grid based IDS systems have also been conceived, designed and implemented. [24] The basic components for grid based IDS systems include: Sensors, which are able to monitor the state of grid systems. The information collected by sensors are then collected and analyzed by IDS system like SNORT [25]. Based upon the analysis, alarm is raised.
  Many approaches for Grid based IDS prevailed in the market. Kenny and Coghlan [28] describe a system that allows the querying of log files through the Relational Grid Monitoring Architecture (R-GMA), which can be used to build a Grid-wide intrusion detection system. Their implementation of this system, called SANTA-G (Grid enabled System Area Networks Trace Analysis), queries Snort log files by using SQL. SANTA-G is composed of three elements: A Sensor, a QueryEngine and a Viewer

GUI. Schulter et al. [29] describes different types of IDS systems. The authors point out that present Grid-IDS approaches do not fulfil the important qualities (completeness—recognition of all attack-types, scalability and Grid-compatibility) for protection of Grid systems. They propose a high-level GIDS that utilizes functionality of lower-level HIDS and NIDS provided through standard inter-IDS communication.
Michal Witold [30] in his thesis investigates the possibility of Grid-focused IDS. The main stress is put on feature selection and performance of the system. Leu and Li [31] proposed Fault-tolerant Grid Intrusion Detection System (FGIDS) which exploits grid's dynamic and abundant computing resources to detect malicious behaviors from a massive amount of network packets. In FGIDS, a detector can dynamically leave or join FGIDS anytime.

3. Self-Protection Model (SPM) for Grid. We have discussed the various problems and the related work that has already been done to avoid the grid system services to be compromised. All of these areas are under constant investigation by researchers and have been so for a long time. There is still no grid middleware that is intelligent enough to handle the failures and faults automatically. We are trying an attempt to unify related research on these by providing the self-management environment to manage the security and faults in grid. SPM model has been given in this paper, providing a way for automatically creating a security wall to protect against external attacks like DDoS. Main advantages of using SPM are:

1. Robust Agent based architecture is used in SPM.
2. Genetic approach used by filter agent to automatically monitor the malicious activities. Many features of GA make it very suitable for intrusion detection. Like parallelism, self-learning capabilities, and the fact that initial rules can be built randomly so there is no need of knowing the exact way of attack machinery at the beginning.
3. Easy to use user interface for job submission and monitoring.

3.1. SPM Architecture. SPM has the layered architecture based upon MVC. MVC stands for Model (Database layer), View (User Interface) and Controller (Business Logic). In the (Figure: 3.1), the application layer is View, the agents and autonomic unit is Controller and the database layer defines the model. Advantages of using this architecture is to achieve:

- Flexible: Can be used with any database, application server with very small effort to change.
- Customizable: Add new features and screens without affecting any existing functionality.

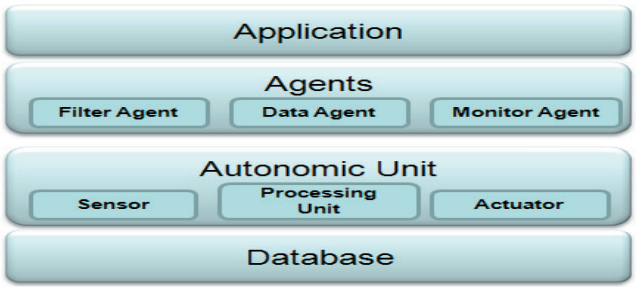


FIG. 3.1. *SPM Architecture*

In this section the details of SPM architecture are described.
Application Layer: This layer provides the user interface for the user to interact with the grid environment. This layer authenticates the originality of the user before letting the user to access the grid. After the user is authenticated, it shows the different options to the user to submit, monitor and cancel the job.
Agents Layer: Agent layers consists of three types of agents and each perform their respective tasks. These are:

- *Filter Agent*: It helps to safeguard the system from unwanted requests and various security threats (Figure: 3.2). Only the authenticated users can submit the jobs. These agents are configured to allow only a certain type of traffic to pass through them. Whenever they detect the illegal packets, they simply discard them. After all inspection is done, they publish the job request to the queues. For this agent, we use Snort to setup the distributed IDS [33][34]. To make filter agent intelligent enough to handle new threats, we have used the Genetic Algorithm (GA) approach to define new rules.

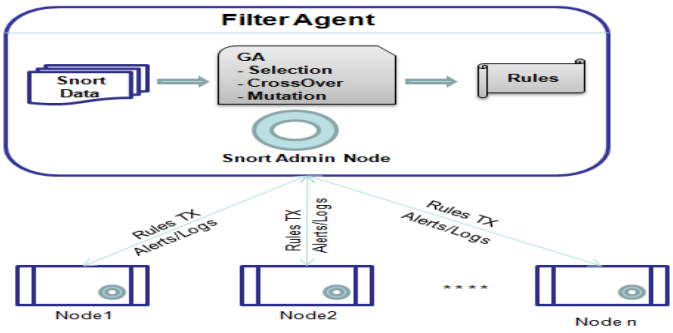


FIG. 3.2. *SPM Filter Agent*

Genetic Algorithm

GA is based upon the principles of evolution and natural selection (Figure: 3.4). GA converts the problem in specific domain into model by using chromosome like data structure and evolve the chromosome using selection, crossing over (Figure: 3.3) and mutation operators [10]. The process of genetic algorithm usually begins with the population of chromosomes representing the gene pool. Different positions of each chromosomes are encoded as bits, characters or numbers and referred to as *Genes*. Genes can be randomly changed within a range during evolution of the species. An evaluation function is used to calculate the "goodness" of each chromosome. During evaluation, two basic operators—Crossing Over and Mutation are used, simulating the natural reproduction and mutation of species. In our

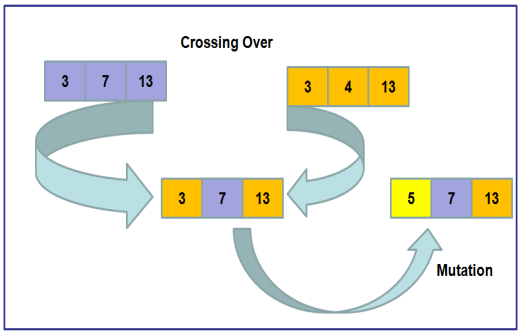


FIG. 3.3. *Crossing Over and Mutation*

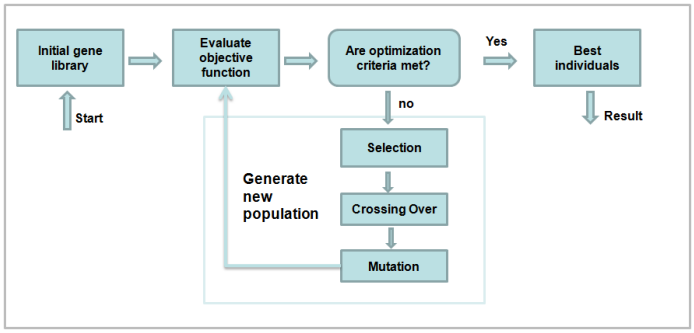


FIG. 3.4. *Structure of GA [11]*

model we use this approach for intelligently defining the new rules which are later transferred to various child resources attached to the central node (Figure: 3.5). Our filter agent GA for snort work like this:

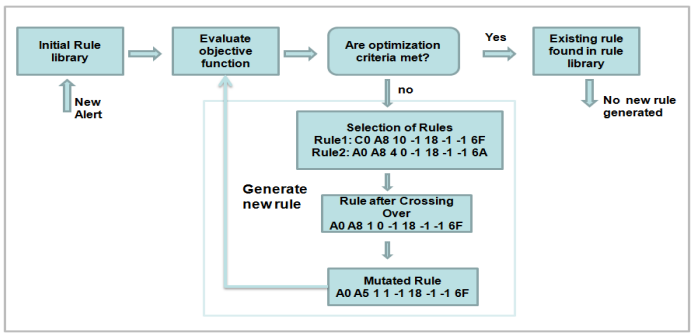


FIG. 3.5. *GA based Rule Generation*

```
SPM Filter Agent GA()
{
      Initialization;
      Evaluation;
      while termination criterion has not been reached
      {
            Selection and Reproduction;
            Crossover;
            Mutation;
            Evaluation;
      }
}
```

Initialization: It involves setting the parameters for the algorithm and creating the first generation of chromosomes. In this benchmark, each chromosome is represented by the set of genes. And the genes contains the description of the rules Let's take an example. Consider one sample rule *alert tcp any any -> 192.168.1.0/24 111* and it is represented as hexadecimal chromosome set (C0 A8 10 -1 18 -1 -1 6F)
Evaluation: Each of the chromosomes in a generation is evaluated for the selection process. This is accomplished by looking up the success rate of the rule representing the chromosome. If the rule is able to find the anomalous behaviour, its fitness is increased else decreased.
Selection and Reproduction: We use the simplest of all i. e. roulette wheel selection[32]. In roulette wheel selection, individuals are given a probability of being selected that is directly proportionate to their fitness. Two individuals are then chosen randomly based on these probabilities and their capability of producing offspring.
Crossover: In the crossover phase, all of the chromosomes with similar lengths are paired up, and with a probability they are crossed over. The crossover is accomplished by randomly choosing a site along the length of the chromosome, and exchanging the genes of the two chromosomes for each gene past this crossover site.Say if we have the two parent chromosomes represented by (C0 A8 1 0 -1 18 -1 -1 6F) and (A0 A8 4 0 -1 18 -1 -1 6A), then the child chromosome can be (A0 A8 1 0 -1 18 -1 -1 6F)
Mutation: After the crossover, for each of the genes of the chromosomes, the gene can be mutated. With the crossover and mutations completed, the chromosomes are once again evaluated for another round of selection and reproduction.We have skipped the details related to performance and working of SPM Filter agent GA to keep things simple for SPM understanding.

- *Data Agent*: Picks the job request from the queues. Store the data into the database. Jobs can be from high priority clients or normal priority. This information is also saved into the jobs database so as to later use this information for deciding the execution based upon the priority. Also these agents are responsible for data transfer from the user machine to the centralized server, so that when the execution unit needs the data for executing submitted jobs, it can pick it from the one single location. For data transfer, ftp protocol is configured that requests the users to upload the necessary data that their job will require for execution. The packets while transfer has to pass through the filter agent that keeps on scanning each packet for any harmful content.
- *Monitor Agent*: It keeps on monitoring the participating nodes in the grid. With the information thus obtained, it updates the resources states in the database. This monitor agent keeps on running and update information about the resources after certain interval of time. If a system is using some

metascheduler, then this agent also have the feature to pick the information from that scheduler rather than from the individual head nodes.

Autonomic Unit Layer: This layer will help in providing the autonomic capabilities to self heal and self-protect the system without any user intervention. This layer consists of:

- *Sensor*: Sensor keeps on monitoring any changes in the jobs or resources database. Whenever any change is introduced, it updates the knowledge bank of the processing unit.
- *Processing Unit*: PU deals with the analysis and planning of job execution. It takes into consideration the job requirements and the available resources wrt those requirements. It is the PU, which based upon the users priority level, makes the plan for job execution. Another things taken into consideration while making the plan are: job status, job resource requirement, time of submission, state of resources involved, etc
- *Actuator*: This takes the plan from the PU, and started working to execute it. Another task that the actuator performs in the updation of the job state in the database.

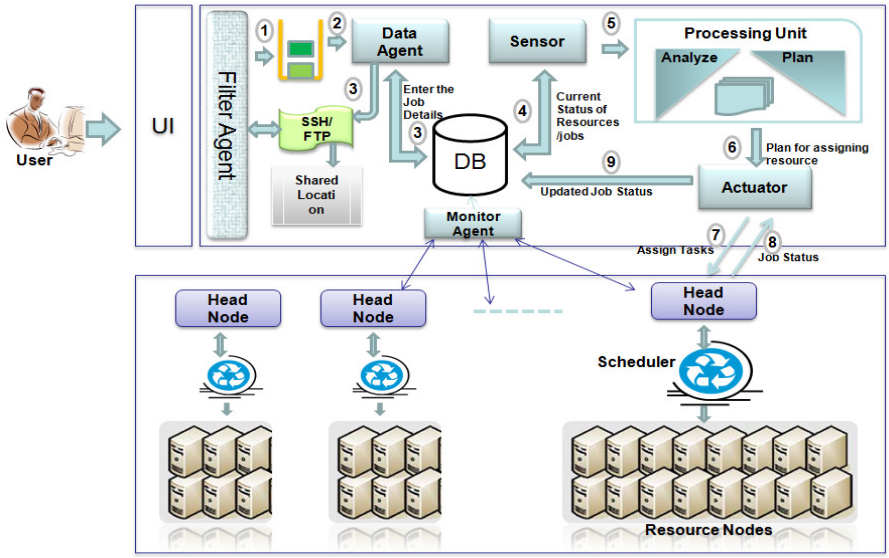


FIG. 3.6. *SPM Detail Working*

3.2. SPM Working. Figure: 3.6 describes the complete process behind the SPM working. This describes the complete process, starting from user's submitting the job from its machine, all the way through to the user receiving the result of submission. Figure: 3.7 discusses the basic flow in few cases. The basic process is as follows:

1. User submits the job through the user interface. UI passes the job request to the *filtering unit*, to scan for any malicious activities. After filtering, jobs are published into the queues.
2. *Data agent* keeps on monitoring the job queue for any new requests. Once discovered any new request, the fetcher in the data agent, fetches the job details and inserts the details into the database.
3. The job data is pushed into the JOB table. Along with this, the data needed by this job is transferred to the central location using FTP. Filtering unit plays its role again to scan the packets before downloading the data to shared location. This helps in avoiding frequent security checks.
4. Sensor monitors the JOB and RESOURCES database and passes the information to the processing unit. The RESOURCES table is updated by the *monitor agent*. This helps in reducing the failure rate caused by invalid old information about the resources.
5. *Processing unit* analyzes the information provided by sensor, and publishes the plan describing which jobs are going to be executed first, which job's need to be resubmitted, what is the current system cpu usage, etc. This plan is shared to Actuator by the processing unit.
6. The *Actuator*, based upon the plan, starts passing the jobs to the meta scheduler (not shown in figure).
7. *Meta scheduler* then schedules the jobs to different head nodes (GRAM nodes in case of globus).

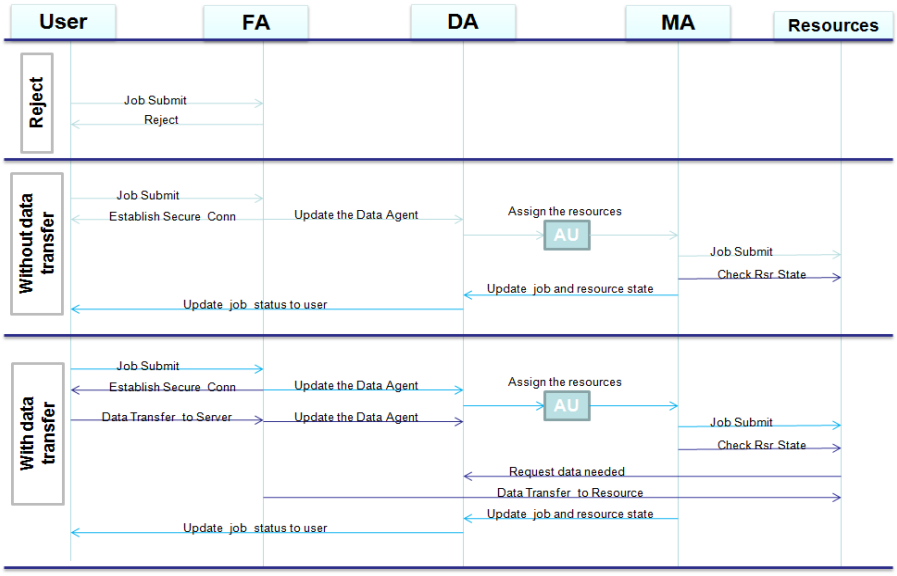


FIG. 3.7. *SPM Work Flow*

8. Actuator keeps monitoring the job status from the scheduler and updates the status into the JOB table.

4. Implementation and Results.

4.1. Softwares Used. The implementation uses: a standard IDS called Snort, ACID monitoring tool, apache web server to host ACID, MySql database and Java SDK6 for programming the tool.

4.2. Grid Environment. Grid Environment setup using Globus Toolkit 4 [37]. The major components involved are:

- *GRAM*: Enables resource allocation through job submission, staging of executable files, job monitoring and result gathering [35].
- *Scheduler*: Helps to schedule the jobs to resources available. Here we use the GridWay MetaScheduler [20] to schedule the jobs.
- *Grid FTP*: Extension of standard FT protocol that provides the secure, efficient and reliable data movements in grid environment [36]. In addition to standard FTP features, it provides GSI support for authenticated data transfer.

4.3. Experimental Setup. The SPM was implemented using the Snort. Snort is used to leverage existing distributed IDS capability to one step ahead by automating its rules generation process by the use of Genetic Algorithm. For testing the SPM in grid environment, we installed snort sensors on different resource nodes participating in the Grid Environmnet, then installed the database(MySql) on one machine, and deployed a Web server and ACID onto another machine. The grid environment is setup inside the TU campus using GT4 middleware. Grid environment consists of 44 Intel Dual Core 2.2GHz processor Windows XP nodes, 20 dual 2.4GHz Xeon Linux nodes and 5 node dual 450MHz PII Linux clusters. Each node has 1GB RAM and 80GB HDD. TU Grid is exposed to the outer world with limited access.

4.4. Experimental Results. To test the SPM, we setup the environment as described above. After environment setup, we configure some default rules in Snort which are later used by all nodes participating in the grid. The rules impose conditions on various GT4 components, these includes port check on GRAM (2119), MDS (2135), Grid FTP(2811), GSI SSH(22) over the TCP protocol for the default users. This rules database keep on increasing with time. As the time passed, the effectiveness of rules is decided on the basis of the alerts raised by it. If the rule raises the correct alert, its impact value is increased and vice-versa. Figure 4.1 is a graphical representation of the number of alerts raised with passage of time with default configurations.

We see that with the passage of time, security of the grid environment with SPM keep on increasing. As the time passed, the number of rules auto generated with help of genetic features- crossing over and mutation

FIG. 4.1. *Time vs Alerts raised*

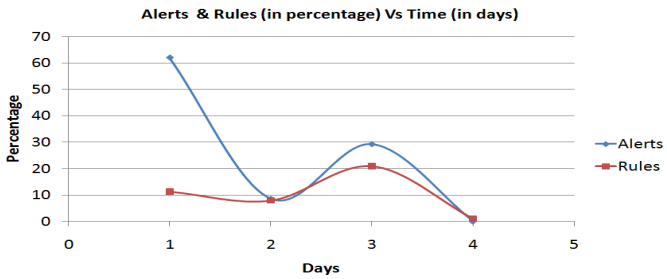


FIG. 4.2. *Alerts & Rules (in percentage) Vs Time (in days)*

also increases. We observed the SPM for the next few days after its setup in grid environment and found that based upon the alerts raised, rules keep on adding into the rule database (Figure 4.2).

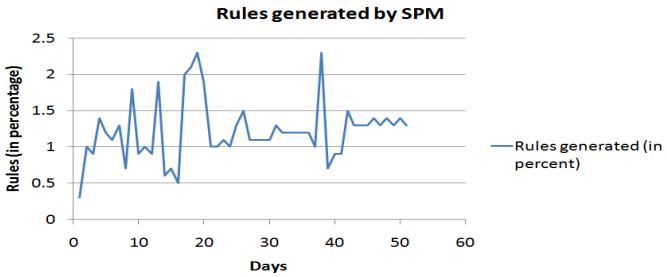


FIG. 4.3. *Rules generated by SPM*

As the rule database grow, more attacks are handled by the SPM. We have found that with respect to the manual system for configuring rules, our SPM reduces the attacks rate (Figure 4.4). After integrating with the current Grid environment, we have found that the attacks from the outside and inside world are reduces faster than the manual system (Figure 4.4). This is because of the increase in rules count due to the automated process implemented using the genetic algorithm.

5. Conclusions. As grid involves heterogeneous resources located in different geographical domains, secure and fault tolerant resource allocation services will increase its performance and efficiency. In this paper attempt has been made to list the most common security threats and the failures encountered by most of the present grid systems. The description of a new approach- SPM has been proposed to deal with failures and security threats with least manual administrative interference. This new approach has layered architecture comprising- Application Layer, Agent Layer and Autonomic Layer. SPM uses the Genetic Algorithm approach to provide the self-protection. The advantage of GA approach over existing approaches is that, SPM makes the system more robust. As central node transfers rules to all the child nodes once they are created through crossover or mutation, in case of central node failure, the system will still keep on protecting the local system. Another advantage is that this system is easily scalable. As the central node carries all the rules, once the new node is registered with the grid system, it will automatically receives all the existing rule set from the central node.

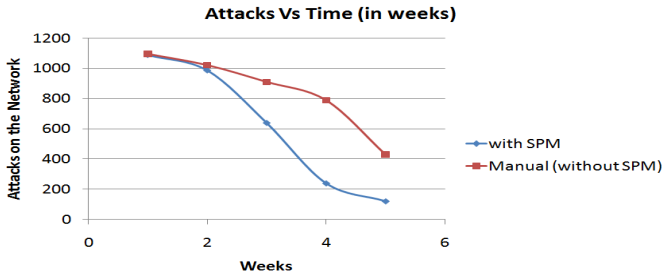


FIG. 4.4. *Attacks Vs Time (in weeks)*

The detailed working of SPM has been described. The goals achieved by the proposed system are summarized as:

- Reduction of administrative complexities because of decreased manual intervention.
- Automatic protection of the system from malicious activities like DoS, DDoS, etc by SPM's Self-Protection mechanism.
- Automatic monitoring of system performance.
- Scalability of the system to add new nodes easily into the grid.

With the implementation and results presented in the paper, we have tried to prove the prototype model for SPM. In the future work, there is the scope to test the feasibility and correctness of the automatically generated rules by GA.

SPM Algorithm

```
/*
Pass through the application layer
/
#VerifyUserData(){
        // validate the user
        IF user exist THEN
            IF PassThroughFilterAgent THEN
                Publish the job to queue
                JobId, UID, Priority, etc.
            ENDIF
        ENDIF
}
/*
filter agent working
/
#filterAgent(){
        GeneticRuleFormation();
        Transfer the rules to child nodes
}
#GeneticRuleFormation(){
        Initialization
            Based upon the existing rules, form the chromosome structure(hexadecimal format)
        Evaluation
            More the rule give the correct result, better is its fesibility factor
        WHILE termination criterion has not been reached
            Selection and Reproduction
                Based upon the feasibility factor select the set of chromosome
            Crossover
                two chromosomes are combined to reproduce the new offspring
            Mutation and Evaluation
                some major change is done to the existing chromosome to form the new chromosome
        ENDWHILE
        Update the rules database with new chromosome thus formed.
}
#monitorAgent(){
        Check the status of resources
        We will not be checking the resources actually, based
        upon the job statuses and time they are taking to execute
        we are predicting the grid load.
        IF NumberofJobspending/hr¿AvgNumberof Executing job/hr THEN
            Performance is degrading
        ENDIF
}
#dataAgent(){
        Collects the job info from queue
        IF job needs data from user THEN
```

```
            Form the ssh session
            Transfer the data to central location
        ENDIF
        Update the job information in the job's database
}
#raiseAlert(){
        At some time interval, agent will be called that will raise
        Alert for all the still cancelled jobs in the database
        Select job_id from job where state=cancelled
        IF count>0 THEN
            Raise the alert
        ENDIF
}
```

Acknowledgments. I thank Dr. Bhupinder Kaur, M.D.S. for clearing my doubts related to genetic algorithms.

REFERENCES

[1] Y. A. Mohamed and A. B. Abdulla, *Immune Inspired Framework for Securing Hybrid MANET*, ISIEA, IEEE (2009).
[2] Benoit Claudel and Noel De Palma and Renaud Lachaize and Daniel Hagimont, *Self-protection for Distributed Component-Based Applications*, Springer-Verlag Berlin Heidelberg (2006).
[3] Ramon Nou, Ferran Julia, and Jordi Torres, *The need for self-managed access nodes in grid environments*,IEEE, EASe (2007).
[4] M. Romberg, *The unicore grid infrastructure*, `http://www.unicore.org` (2002).
[5] B. Sotomayor and L. Childers, *Globus Toolkit 4: Programming Java Services*,(2005).
[6] Hang Guo, Ji Gao, Periyou Zhu, Fan Zhang, *A Self-Organized Model of Agent-Enabling Autonomic Computing for Grid Environmnet*, 6th Word Congress on Inteligent Control (2006).
[7] Alan Ganek, *Overview of Autonomic Computing: Origins,Evolution, Direction*,CRC Press(2004).
[8] Jeffrey O. Kephart and David M. Chess, *The Vision of Autonomic Computing*,IEEE Computer(2003).
[9] Marty Humphrey and Marry Thompson, *Security Implications of Typical Grid Computing Usage Scenarios*,Cluster Computing, Volume 5 , Issue 3(2002).
[10] Wei Li, *Using Genetic Algorithm for Network Intrusion Detection*,Proceedings of the United States Department of Energy Cyber Security Group (2004)
[11] Pohlheim Hartmut, *Genetic and Evolutionary Algorithms: Principles, Methods, and Algorithms*, Genetic and Evolutionary Algorithm Toolbox (2003)
[12] I. Chopra, M.Singh, *Analysing the need for autonomic behaviour in grid computing*, ICCAE P535-539, IEEE (Feb 2010)
[13] , *Stopping insider attacks: how organizations can protect their sensitive information*, ibm.com/services (2006)
[14] Ian Foster, Carl Kesselman, Jeffrey M. Nick, and Steven Tuecke, *The physiology of the Grid*, Global Grid Forum(2002)
[15] Salim Hariri, Guangzhi Qu, and et.al., *Quality-of-Protection (QoP)-An Online Monitoring and Self-Protection Mechanism*, IEEE Journal on selected areas in communications, vol. 23, no. 10 (2005)
[16] P. Horn, *Autonomic computing: IBM's perspective on the state of information technology*, `http://www.research.ibm.com/autonomic/` (2001)
[17] Kevin J. Houle, George M. Weaver, *Trends in Denial of Service Attack Technology*, CERT® Coordination Center (October 2001)
[18] Ian Foster, Carl Kesselman, and Steven Tuecke, *The anatomy of the Grid*,John Wiley and Sons (2003)
[19] Ju Wang, Xin Liu and Andrew Chien, *Empirical Study of Tolerating Denial-of-Service Attacks with a Proxy Network*, Proceedings of the 14th conference on USENIX Security Symposium (2005)
[20] Ruben S. Montero, *The GridWay Meta-Scheduler,Open Source Grid and Cluster*, Oakland, CA (May 2008).
[21] Manish Parashar and Salim Hariri, *Autonomic Computing: An Overview*,Springer-Verlag Berlin Heidelberg (2005).
[22] Gerald Tesauro and David M. Chess and William E. Walsh and Rajarshi Das and et.al, *A Multi-Agent Systems Approach to Autonomic Computing*, AAMAS'04 ACM (Jul 2004).
[23] Christian Kreibich, Jon Crowcroft, *Honeycomb - Creating Intrusion Detection Signatures Using Honeypots*, ACM SIGCOMM, January (2004).
[24] Anirban Chakrabarti, *Grid Computing Security*, Springer, Ch-6 pp105, (2007).
[25] Snort, *https://edge.arubanetworks.com/article/leveraging-centralized-encryption-snort-part-1*
[26] Mark Stephen, V.S. Sukumaran Nair, Jacob. Abraham, *Distributed Computing Grids- Safety and Security, Security in Distributed, Grid, Mobile, and Pervasive Computing*, Chapter 14
[27] Anirban Chakrabarti, *Grid Computing Security*, Springer, Ch-8 pp159, (2007).
[28] S. Kenny and B. Coghlan, *Towards a Grid-Wide Intrusion Detection System*, Advances in Grid Computing. Springer, pp. 275–284, (2005).
[29] A. Schulter, F. Navarro, F. Koch, and C. Westphall, *Towards Gridbased Intrusion Detection*, Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP, pp. 1–4, (2006).
[30] Michal Witold Jarmolkowicz, *A Grid-aware Intrusion Detection System, Technical University of Denmark*, IMM-THESIS (2007).
[31] Fang-Yie Leu, Ming-Chang Li, Jia-Chun Lin, *Intrusion Detection based on Grid*, ICCGI'06, (2006).
[32] M. Srinivas, Lalit M. Patnaik, *Genetic algorithms: A survey*, IEEE Computer, 27(6):17–26, June (1994).

[33] Michael P. Brennan, *Using Snort For a Distributed Intrusion Detection System*,SANS Institute (2002).
[34] Hassen Sallay, Khalid A. AlShalfan, Ouissem Ben Fred, *A scalable distributed IDS Architecture for High speed Networks*, IJCSNS, VOL.9 No.8, August (2009).
[35] M. Feller and I. Foster and S. Martin, *GT4 GRAM: A Functionality and Performance Study*, (2008).
[36] Data Management: Key Concepts, Grid FTP, `http:/www.globus.org/toolkit/docs/4.0/data/gridftp` (2008).
[37] Globus Alliance, *A Globus Primer: Describing Globus Toolkit Version 4*, `http:/www.globus.org/toolkit/docs/4.0/` (2008).

Edited by: Jemal H. Abawajy, Mukaddim Pathan, Al-Sakib Khan Pathan, and Mustafizur Rahman
Received: October 11th, 2010
Accepted: November 10th, 2010

Scalable Computing: Practice and Experience
Volume 11, Number 4, pp. 385–392. http://www.scpe.org
ISSN 1895-1767

REPUTATION-BASED MAJORITY VOTING FOR MALICIOUS GRID RESOURCES TOLERANCE

AHMED BENDAHMANE, MOHAMMAD ESSAAIDI, AHMED EL MOUSSAOUI AND ALI YOUNES*

Abstract. Recently, distributed computing infrastructures such as grid computing, have witnessed huge developments which have a very important impact on computationally intensive scientific, industrial and engineering problems. Collaborative computing, resources access and sharing facilitated by grid computing systems amplifies concerns about cyber attacks and misbehaviors of grid resources (resources provider). In many cases, an organization may send out jobs for remote execution on untrustworthy machines. Since resource sharing and cooperation among different administrative domains are among the central goals of grid computing, it is necessary to guarantee that, among the shared resources, there are no malicious resources interested in invalidating or corrupting the job results. In order to guarantee reliable jobs execution in grid systems, we have developed a new approach for malicious grid resources tolerance using reputation to improve the efficiency of majority voting mechanisms. The grid broker service investigates the trustworthiness of the grid resources used in the voting procedure; this investigation is based on distributed and hierarchical reputation management system which maintains a reputation value for each grid resource based on its historical behavior. The validation of the result is then decided based on the grid resources reputation level.

Key words: grid computing, malicious resources, tolerance, majority voting, reputation

1. Introduction. A grid system is a platform that provides access to various computing resources owned by different institutions through the creation of virtual organizations [1]. A grid virtual organization (VO) allows a seamless aggregation of computational resources in multiple administrative domains into a single pool. The users can lease for a certain time-frame bundles of software, CPU cycles, bandwidth and storage space in order to execute computationally intensive tasks, according to a service level agreement (SLA) between the user and the grid broker service. This broker is acting as a service intermediate between users and grid resources. This brokering role can either happened in strong structured grids, with long-lasting VOs inside the VO boundaries or in volatile grid environments, with very dynamic VOs, where the grid resources can dynamically join and leave in the middle of an execution without impediment. In the second scenario, which is very close to the grid design adopted by the XtreemOS project [15], the execution of a task may be halted arbitrarily. It is then mandatory for VO manager to tolerate these misbehaviors in order to assure the security of the user jobs execution.

Moreover, as the grid resources are connected through the Internet, and with the fusion of web services and grid technologies, and due to the inherently dynamic and complex nature of grid system, a range of new security threats and vulnerabilities are coming to the fore with the broadening scope of grid computing [3], and especially the grid resources has become a subject of several unknown attacks. Introduction of grids in the commercial sector has open up new ventures in the business arena [8]. However, the contemporary grid application domains require many protections from malicious entities. These applications and their underlying infrastructure have to protect the critical information about their owners; provide safeguards to the economic interests of the enterprises using the grid system; and win the confidence of the society which is already doubtful of the digital data processing [5].

Conventional grid security mechanisms [21] can ensure several security services. Some of them guarantee communication confidentiality between grid nodes [17, 18] while others investigate resources access [19, 20]. As far as data integrity is concerned, most mechanisms that have been proposed to guarantee it will do it only during the transmission; employing authentication and encryption technologies (as in Globus system) [22, 23, 24]. However in order to really guarantee data integrity in a grid system, it is also needed to ensure the integrity of the job result processed by the grid resources. Hence, it is necessary to protect the applications in process in the grid against possible malicious behavior of attacked resources that can affect the efficiency of the grid system by returning erroneous jobs results.

In addition, the existing attacks/intrusions detection and prevention mechanisms are not efficient against unknown attacks. Therefore, another kind of defense mechanism is needed to implement the precaution against all possible attacks/intrusions. This creates the requirement that grid environments should detects and tolerates the grid resources attacked by an adversary, which tamper with the computation and return corrupted results, in order to guarantee reliable execution in such a suspicious environment.

*Information and Telecommunication Systems Laboratory, Faculty of Science, Tetuan, Morocco, ({a.dahman, elmoussaoui, younes.ali}@uae.ma, essaaidi@ieee.org).

In this paper, we propose a new approach for tolerating malicious grid resources by using majority voting mechanisms, and then investigate the trustworthy of the grid resources which are used in voting procedure. This investigation is based on reputation management system. And the validation of result is decided based on the reputation level of the grid resources.

The remainder of this paper is organized as follow. Section 2 presents our motivation of using sabotage tolerance techniques in grid. Section 3 survey the background of these techniques. In section 4, we present our vision of basic components of grid system and the possibility of attacking these components. Section 5 describes our approach of using majority voting with reputation. Section 6 presents how the grid broker service will build the reputation of grid resources. At last, section 7 concludes the paper.

2. Motivation. Sabotage tolerance is gaining importance in grid environments notably in the situation where different grid domains have conflicting interests. The term sabotage tolerance was originally coined in the specific area of Desktop Grids, where voluntarily Internet users contribute to the grid with computing cycles. Because everyone can take part in such a grid, the environment started loosing its trustworthiness. Since computations run in an open and un-trustable environment, it is necessary to protect the integrity of data and validate the computation results. Sabotage tolerance techniques need to be employed mainly for the detection of malicious entities who may submit erroneous results.

In a classical grid, sabotage tolerance is not an issue because the grid environment is trustable, in the sense that someone (grid node administrator) controls strictly the grid resources and their ownership. When a grid resource is malfunctioning, the grid owner is notified. However, if the grid scales at the size of the Internet or spans over multiple administrative domains without a hierarchical subordination of control and ownership, an attacker might corrupt the grid resources by exploiting some of its vulnerabilities, then it is likely that grid resources report erroneous job results. Sabotage tolerance becomes mandatory, for the protection of both the grid users and other grid contributors.

Sabotage tolerance techniques are applied in grid systems that employ the master-worker computational model [13]. This grid model is not restrictive and maps well on the wide variety of grids. This model can be summarized as a server (referred further as the master) that distributes work units of an application to grid nodes (workers). A result error is any result returned by a worker that is not the correct value or within the correct range of values [10]. Sabotage-tolerance techniques imply detecting result errors, which are very important as they can undermine long computations that have been executing during weeks or even months by several workers [4]. The error rate ε is defined as the ratio of bad results or errors among the final results accepted at the end of the computation. Thus, for a batch of N work units with error rate ε, the master expects to receive εN errors. For every application, the master employs some sabotage-tolerance mechanism for obtaining an acceptable error rate ε_{acc} with regard to its application. If a grid user comes with an application divided on a big number of tasks (e.g. 10 batches of 100 work units each) and it requires a global error rate of 10^{-2}, the sabotage tolerance technique should provide with a work unit error rate of about 10^{-5} [10]. Many applications, especially the ones from computational biology and physics require bounds on the error rates, as the correctness of the computed results is essential for making accurate scientific conclusions.

3. Background and related work. In this section, we provide a short overview for the principal sabotage tolerance techniques in Desktop Grids. Sabotage tolerance techniques for Desktop Grids can be classified in four big classes [4]: reputation systems, replication with voting, sampling, and checkpoint-based verification.

Reputation systems [26] are commonly used to estimate the reliability of grid resources based on past interactions. The concept of trust-aware resource management for the Grid was proposed in [27], where a technique is presented for computing trust ratings in a Grid using a weighted combination of past experience and reputation. Zhao and Lo [16] propose augmenting peer-to-peer cycle sharing systems with a reputation system to reduce the degree of replication required to verify results. GridEigenTrust [28] combines this trust-computation technique with the EigenTrust reputation system [29] to provide a mechanism for rating resources in a Grid. These existing reputation systems have focused on correctness as the primary metric, and have dealt mainly with binary trust values. The key concept in our reputation model is the combination of the credibility, availability, and security level to compute the overall grid resources reputation.

Replication with majority voting [13] is widely used in the BOINC Desktop Grid platform [2]. The master distributes $2m-1$ replicas of a work unit to workers and when it collects m similar results, it accepts that result as being the correct one. Each collected result is seen as a vote in a voting pool with $2m-1$ voters and with majority agreement being the decision criteria. The error rate of this method is determined by the number of

identical results required (m), which is a measure of redundancy. High levels of redundancy provide very low error rates (less than 10^{-5}). The main benefit of the method is its simplicity, while the big drawback is the fact that it wastes a lot of resources.

Sampling-based techniques are developed to overcome limitations of replication, especially redundancy. Within sampling, the master determines the trustworthiness of the workers by verifying them only on a few samples of their results. The basic sampling is the naïve one [14], where the master sends probes with verifiable results to workers. If workers respond well to the probes, they are considered trustworthy. The main drawback is the workers can easily recognize the probes and respond well to them, while cheating on the ordinary work units. The probing by spot checking is introduced by Sarmenta [13]. A credibility based mechanism is presented to protect a volunteer computing environment against malicious users. Probes, named now spotters, are tasks with known results. If a worker fails to compute correctly a spotter, it will get blacklisted and all its results will be invalidated. Based on spot-checking, Sarmenta defines the credibility of a worker and a result. The credibility of a worker is an estimate of the probability the worker will return a correct result. The credibility of a result is the conditional probability that the result originating from a worker will be accepted as correct. Sarmenta approach is essentially a centralized one, the fact that adds an overhead to the central resource responsible for the tests (every spotter job is also processed by the central resource). In our approach, which is based on a distributed and hierarchical paradigm, we delegate the spot-checking operations to other resources known to be trustworthy; we assume that there are some highly pre-trusted resources in the grid system we call reputation agent. Moreover, we will use these highly pre-trusted resources to store the credibility value, and then to reduce the traffic and the processing effort by a centralized resource according to the tests results.

Quizzes [16] are an improvement to basic probing. With this method, the master sends to workers batches with work units and it places the probes inside of those batches. Given the actual required error rate, the master can compute the number of quizzes to place inside a batch. A drawback of probing is the fact the master should possess some heuristic in order to generate the probes and make them to resemble with the real work units. Because this is a difficult task, the master can use actual work units as probes [13]. The master verifies (using replication) only a sample of results and if a worker is caught cheating all its previous results are invalidated.

Checkpoint based verification addresses the problems with sequential computations that can be broken in multiple temporal segments (S_{t1}, S_{t2}, . . . , S_{tn}). At the end of each segment a checkpoint is submitted to a stable storage. The checkpoints are stored locally to allow recovering the long tasks from faults. After finishing a task, the worker sends back the result along with a list of hashes for the checkpoints. In the basic checkpoint verification [11], the master randomly selects a checkpoint time S_{ti} for a task and asks the worker to deliver its local checkpoint $C(S_{ti})$. Then the master computes the task from S_{ti} up to the next checkpoint and compares the results with the hash value submitted by the worker for $C(S_{ti+1})$. If the hash verification succeeds, then the worker passed the verification. In the distributed version of the checkpoint verification, the master selects a third worker for verification purposes and let the verifier to compute the checkpoint verification. The distributed version has the advantage of not overloading the master with a lot of verification tasks. The error rate of this method strongly depends on the number of verified checkpoints - i. e. a high percentage of verified checkpoints yield a low error rate, for the cost of increased computation (redundancy) and bandwidth.

4. Model and assumption. Fig. 4.1 shows the basic components of a grid system of interest, which consists of N Virtual Organizations (VOs), where each VO contain a set of resources, services and middleware, applications, and users, connected by a local network and the Internet (Fig. 4.2). From the service point of view, these components are considered as resources provider and services provider. The former owns computational nodes or other physical resources, and has root privilege. It decides whether and when resources should contribute and the quantity of available resources that are needed for job execution. The number of resources is dynamic and undetermined. Besides, each resource provider may require different authentication, authorization mechanisms and security policies. The second is the owner of the software solutions and the information databases that are deployed on a resource provider's assets. The software solution is an individual service or a combination of services to solve user's submitted jobs or requests. In combination situation, the service provider then comes to be a user that requests other services.

In grid computing environment, a client is a parallel job submitter who requests results, he/she can submit a job to grid broker service (after SLA establishment), which determines where and when to run the job. The broker primarily mediates user access to grid resources by discovering suitable services and resources provided by VOs and deploying and monitoring job execution. In the other hand, each VO in the grid has a single VO

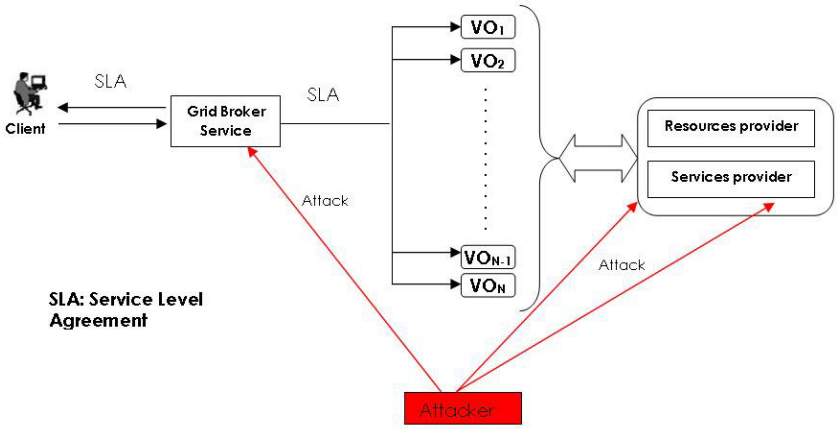


FIG. 4.1. *Basic components of a grid system*

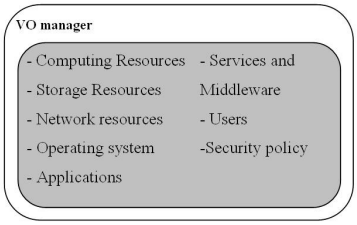


FIG. 4.2. *Virtual organization (VO) components*

manager, which searches for available services. When it finds an interesting service, it negotiates with the service provider to allow access to the service from their VO. Once the service provider authorizes the use of the service, the service becomes available at this VO. The job is then divided into subjobs that have their own specific input data. The subjob is called a task. The broker distributes tasks to selected grid resources (computing resources) using scheduling mechanisms. When each grid resource subsequently finishes its task, it returns the result of the task to the broker. Finally, the broker returns the final result of the job back to the client.

In our study, the VO manager includes a reputation agent component, responsible for spot-checking operations, and for the reputation value computation for each grid resource within the VO. The reputation values are then returned to the broker.

In this execution scenario of jobs, there are several possible attacks to grid resources which might tamper with the computation and return corrupted results (grid resources takeover by an unauthorized user can result to malicious acts like corruption of job data, job termination etc.). Such attacks are usually related to security holes of the services and resources provider [9]. Denial of Service (DoS) attacks [7] may also disrupt job completion or temporarily prevent access to job output by cutting a site off the Resource Broker that has delegated the job and is waiting for the output.

5. Malicious grid resources tolerance using reputation. Attacks on grid computing, as on any other system, rely on a large number of different vulnerabilities in grid security. Therefore, these attacks become an increasingly prevalent form of security threats, and the problem appears to be quite difficult to solve, especially for attacks which can corrupt the grid resources and make them provide erroneous results of execution. Nevertheless, there exist several defense mechanisms that have been proposed to detect and prevent these attacks, these mechanisms can stop known attacks, but can not defend against new attacks which can happen in grid system. To this end, tolerance mechanisms for malicious grid resources become necessary to provide a safe result of job execution. As a solution, we propose an approach of tolerating these types of attacks by using sabotage tolerance techniques.

The proposed method is based on reputation to improve the efficiency of majority voting mechanisms. With majority voting the grid broker service (i. e. master) decide about the trustworthiness of a result immediately

after having the result of all replicas of a task (i. e. work unit). This approach tolerates a certain number of incorrect results in a vote. However, it does not resist when, for example an adversary gets control through a virus over a large number of computing resources (i. e. workers), and make them return the same result. In our approach, the broker will postpone the decision until the moment it gathers enough information to infer the trustworthiness of the result. To achieve this information, we will use existing reputation lists of different computing resources in VO.

5.1. Problem formulation. Let us assume that each task is replicated n times and allocated to several computing resources C_i, so that a broker can collect m different results V_j, where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$. Each computing resource has its reputation value R_i, this reputation is collected by a grid broker service, which contains the reputation lists for all computing resources C_i. The reputation is a value in the range between 0 and 1. In Sect. 6 we will show how the reputation agent will compute the reputation value.

To make a decision about which result V_j is trustworthy (i. e. accepted result), the grid broker service utilizes a majority voting based on reputation decision criteria.

Let the relativity vector $Re(V_j)$ represents The aggregated vote of all computing resources that return the same result.

$$Re(V_j) = [T(V_j, C_1), T(V_j, C_2), \ldots, T(V_j, C_n)] \tag{5.1}$$

Where $T(V_j, C_i)$ is the relationship between the result V_j and the computing resource C_i. It is calculated as follow.

$$T(V_j, C_i) = \begin{cases} 1, & \text{if } C_i \text{ compute } V_j \\ 0, & \text{otherwise} \end{cases} \tag{5.2}$$

We define the result reputation $RR(V_j)$ of a given result V_j as the summation of reputations of the computing resources returning the result V_j.
For each result V_j:

$$RR(V_j) = \sum_{i=1}^{n} T(V_j, C_i) R_i \tag{5.3}$$

To make a decision we fixe a positif threshold value $\lambda < 1$ and we pick the maximum of $RR(V_j)$; $(j = 1, \ldots, m)$.

$$\begin{cases} if \quad max(RR(V_j)) > \lambda \sum_{i=1}^{n} R_i; & \text{the result corresponding to this maximum is accepted} \\ Otherwise; & \text{the grid broker service should decide for further replication} \end{cases}$$

To avoid the possibility that a set of computing resources with a low reputation could undermine the result, we impose the following condition.

$$R_i = \begin{cases} 0 & \text{, if } R_i < \Theta \\ R_i & \text{, otherwise} \end{cases} \tag{5.4}$$

Where Θ represents the minimum reputation value a computing resource should have for its results to be taken in consideration.

6. Computing reputation. In this section, we present how the grid broker service will build the reputation value of each computing resource (or grid resource).

The reputation of an entity is an expectation of its behavior based on other entities observations or information about the entities past behavior at a given time [6]. The reputation management services are responsible for evaluating the reputation of grid resources, services, and users inside a VO. In our vision, the grid broker service builds the reputation of each computing resource through its credibility, availability, and security level.

6.1. Credibility. The credibility represents the likelihood of a particular object of the system to be operating properly [12]. The reputation agent the credibility of the computing resource C_i by passing spot checking (see Sect. 3). If f is the proportion of compromised computing resources, the credibility of a computer resource which correctly computed k_i spotters (tasks), when blacklisting is not used, will be computed by [13]:

$$CR(C_i, k_i) = \begin{cases} 1 - \frac{f}{k_i}, & \text{if } K_i \neq 0 \\ 1 - f, & \text{otherwise} \end{cases} \tag{6.1}$$

With blacklisting policy [25], the malicious resources which return an erroneous result for the spotter job are immediately blacklisted, i. e. the malicious resources are never allowed to return results or to get any more jobs. In our study, we prefer to compute credibility without the blacklisting policy, because in some way it is possible that a malicious resource may change its behavior with the time. For instance, resources may be infected by a virus and after being repaired it will become reliable again. Thus, these repaired resources can participate in a computation and return a correct result, and can then improve the performance of the grid system.

After the computing resources pass enough tasks k_{min}, they succeed to obtain enough credibility (minimum credibility) to guarantee that their results are correct given the error rate ε_{acc}. First, we initialize the credibility value $CR_{init}(C_i)$ of all computing resources. The initial value is no more than the minimum credibility of the computing resource itself.

$$CR_{init}(C_i) = CR(C_i, k_{min}) \tag{6.2}$$

The credibility value is incremented in such a way that if a computing resource returns a validated result by the reputation-based majority voting described above. We propose to consider as a passed spotter each task successfully validated by the broker, using the reputation-based majority voting. Then, we update the credibility value using this equation:

$$CR(C_i, k_i) = CR(C_i, k_i + 1) \tag{6.3}$$

In the same manner, we do decrease the credibility of those computing resources whose result was not validates by the reputation-based majority voting. To this end, we update the credibility value using this equation:

$$CR(C_i, k_i) = CR(C_i, k_i - 1) \tag{6.4}$$

6.2. Availability. The Availability Ai is the ratio of the number of successful contacts to the computing resource C_i and the total number of requests. A non-available resource means that it is not accessible [30]. Thus, the computation of availability is given by:

$$A_i = \frac{N_S}{N_R} \tag{6.5}$$

Where N_S is the number of successful contacts to the computing resource, and N_R is the Number of the total number of requests.

When a computing resource joins the grid system, it may participate in the computation of several jobs. Every time, its availability value will be updated.

6.3. Security level. In our approach, we consider the security level of a computing resource as a self-protection capability.

As defined in [31], the self-protection capability of a computing resource is calculated by aggregating the security factors. The values of these factors differ in the range between 0 and 1. Based on their contributions to security, a proportion is given to each security factor as a final point aggregated to compute the self-protection capability. The security factors and their related proportions are listed in Table 6.1 [32].

Thus, the security level (SL_i) is calculated using the following equation:

$$SL_i = \frac{\sum_{s=1}^{n} P(s)L(s)}{n} \tag{6.6}$$

Where n is the total number of factors. $P(s)$ is the proportion and $L(s)$ is the value of the factor determined by the reputation agent trust strategy.

TABLE 6.1
Proportion of security factors.

Security Factors	Proportion (P)
IDS Capabilities	0.825
Anti-virus Capabilities	0.85
Firewall Capabilities	0.9
Authentication Mechanism	0.8
Secured File Storage Capabilities	0.7
Interoperability	0.6
Secured Job Execution	0.75

6.4. grid Resources reputation. The reputation R_i value of a computing resource C_i, that will be used by the reputation-based majority voting, computed by the reputation agent, is the product of the following three metrics: credibility, availability, and security level, and it is computed using the following equation:

$$R_i = CR(C_i, k_i) \times A_i \times SL_i \tag{6.7}$$

As the SL_i of a computing resource is usually stable for a period of time, when a new computing resource joins the grid, it will be calculated as the initial reputation value of the computing resource. Each result produced by the computing resource will enter the reputation-based majority voting competition. If the result is accepted, the reputation of the computing resource will be increased. If not, it will be decreased.

7. Conclusion. In this work, the proposed approach for tolerating attacks in grid resources is discussed. This approach is based on majority voting mechanisms and reputation management system. The voting result that is generated by the majority voting can be combined with the reputation of grid resources owned by VOs, and then it may be possible to decide on the authenticity of suspicious grid resources. Using reputation information with majority voting, our grid computing system can provide job results correctness. Thus, our tolerance scheme for detecting suspicious grid resources is more accurate and more reliable. And help to improve the security of grid computing system.

REFERENCES

[1] F. BERMAN, G. C. FOX, AND A. J. G. HEY, *Grid Computing: Making the Global Infrastructure a Reality*, Wiley, Chichester, 2003.

[2] P. A. B. DAVID, *Boinc: A system for public-resource computing and storage*, In GRID2004. The Fifth IEEE/ACM International Workshop on Grid Computing, IEEE Computer Society, Washington, 2004, pp. 4–10.

[3] Y. DEMCHENKO, L. GOMMANS, C. LAAT, AND B. OUDENAARDE, *Web services and grid security vulnerabilities and threats analysis and model*, In GRID2005. The 6th IEEE/ACM International Workshop on Grid Computing, IEEE Computer Society, Washington, 2005, pp. 262–267.

[4] P. DOMINGUES, B. SOUSA, AND L. M. SILVA, *Sabotage-tolerance and trust management in Desktop Grid computing*, Future Generation Computer System 23(7), 2007, pp. 904–912.

[5] J. ERMISCH, AND D. GAMBETTA, *People's trust: The design of a survey-based experiment*, IZA Discussion Papers 2216, Institute for the Study of Labor, 2006.

[6] A. FARAG, AND M. MUTHUCUMARU, *Towards Trust-Aware Resource Management in Grid Computing Systems*, In CCGRID2002. 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, IEEE Computer Society, Washington, 2002, p. 452.

[7] P. GALLI, *DoS attack brings down Sun Grid demo*,(2006, Mars). Available: `http://www.eweek.com/article2/0,1895,1941574,00.asp`

[8] ———, *Commercial grid solutions, Grid Computing Planet* , (2006, December, 18). Available: `http://www.gridcomputingplanet.com/resources/article.php/933781`

[9] N. JIANCHENG, L. ZHISHU, G. ZHONGHE, AND S. JIRONG, *Threat analysis and Prevention for grid and web security services*, InProc. of Eighth ACIS. International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007, pp. 526–531.

[10] D. KONDO, F. ARAUJO, P. MALECOT, P. DOMINGUES, L. M. SILVA, G. FEDAK, , AND F. CAPPELLO, *Characterizing result errors in Internet Desktop Grids*, In Euro-Par2007. LNCS, vol. 4641, Springer, Heidelberg, 2007 pp. 361–371.

[11] F. MONROSE, P. WYCKOFF, AND A. D. RUBIN, *Distributed execution with remote audit*, The Network and Distributed System Security Symposium, NDSS. The Internet Society, San Diego, 1999.

[12] A. C. OLIVEIRA, L. SAMPAIO, S. F. FERNANDES, AND F. BRASILEIRO, *Adaptive Sabotage-Tolerant Scheduling for Peer-to-Peer Grids*, In LADC2009. Fourth Latin-American Symposium on Dependable Computing, IEEE Computer Society, Joao Pessoa, 2009, pp. 25–32.

[13] L. F. G SARMENTA, *Sabotage-tolerance mechanisms for volunteer computing systems*, Future Generation Computer Systems 18(4), 2002, pp. 561–572

[14] D. Wenliang, J. Jing, M. Mangal, and M. Murugesan, *Uncheatable grid computing*, In ICDCS2004. The 24th International Conference on Distributed Computing Systems, IEEE Computer Society, Washington, 2004, pp. 4–11.

[15] E. Y. Yang, B. Matthews, A. Lakhani, and Y. Jégou, *Virtual organization management in XtreemOS: an overview*, Towards Next Generation Grids, Proc. of the CoreGRID Symposium, Rennes, France, Springer, Heidelberg , 2007.

[16] S. Zhao, V. Lo, and C. GauthierDickey, *Result verification and trust-based scheduling in peer-to-peer grids*, In P2P2005. The 5th IEEE International Conference on Peer-to-Peer Computing, IEEE Computer Society, Washington, 2005, pp. 31–38.

[17] A. Setiawan, D. Adiutama, J. Liman, A. Luther, and R. Buyya, *Gridcrypt: High performance symmetric key cryptography using enterprise grids*, In PDCAT 2004. Parallel and Distributed Computing - Applications and Technologies, 2004, pp. 872–877.

[18] S. Song, K. Hwang, and J. Lv, *Grid security enforcement with trust integration over minimal vpn tunnels*, Technical Report TR 2004-10, USC Internet and Grid Computing Lab, 2004.

[19] J. Silva and L. Gaspary, *Uma arquitetura de controle de acesso baseado em politicas para grades computacionais peer-to-peer*, In Workshop of Peer-to-Peer in the 23rd Brazilian Symposium on Computer Networks, 2005, pp. 37–48

[20] A. Goldchleger et al, *Integrade object-oriented grid middleware leveraging idle computing power of desktop machines*, In Concurrency and Computation: Practice and Experience, 2004, pp. 449–454.

[21] A. Bendahmane, M. Essaaidi, A. El Moussaoui, and A. Younes, *Grid computing security mechanisms: State-of-the-art*, International Conference on Multimedia Computing and Systems, Ouarzazate, 2009, pp. 535—540.

[22] I. Foster, N.T. Karonis, C. Kesselman, G. Koenig, and S. Tuecke, *A Secure Communications Infrastructure for High-Performance Distributed Computing*, In proceedings, 6th International Symposium on High Performance Distributed Computing (HPDC '97), Portland, Oregon, U.S.A., August 1997.

[23] I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke, *A Security Architecture for Computational Grids*, In proceedings of the 5th ACM Conference on Computer and Communications Security Conference, November 1998, pp. 82–92.

[24] V. Welch et all, *Security for Grid Services*, ANL/MCS-P1024-0203, February 2003. In proceedings, 12th IEEE International Symposium on High Performance Distributed Computing (HPDC'03), Seattle, Washington, U.S.A., June 2003.

[25] A. Bendahmane, M. Essaaidi, A. El Moussaoui, and A. Younes, *Compromised Resources Tolerance in Grid Computing Systems*, Studies in Computational Intelligence, 2010, Volume 315/2010, Intelligent Distributed Computing IV, pp. 145–154

[26] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara, *Reputation Systems*, Communications of the ACM, 43(12) , 2000, pp. 45–48.

[27] F. Azzedin and M. Maheswaran, *Integrating Trust into Grid Resource Management Systems*, In Proceedings of the International Conference of Parallel Processing, 2002.

[28] B. Alunkal, I. Veljkovic, G. von Laszewski, and K. Amin, *Reputation-Based Grid Resource Selection*, In Workshop onAdaptive Grid Middleware, 2003.

[29] S. Kamvar, M. Schlosser, and H. Garcia-Molina, *The Eigen-Trust Algorithm for Reputation Management in P2P Networks*, In Proceedings of the Twelfth International WorldWide Web Conference, 2003.

[30] E. Papalilo, and B. Freisleben, *Managing Behaviour Trust in Grid Computing Environments*, Journal of Information Assurance and Security 1 (2008) pp. 27–37

[31] C. Chen, G. Li-ze, N. Xin-xin, and Y. Yi-xian, *An Approach for Resource Selection and Allocation in Grid Based on Trust Management System*, First International Conference on Future Information Networks, ICFIN 2009, Beijing, pp.232–236.

[32] V. Vijayakumar, and R. S. D. Wahida Banu, *Trust and Reputation Aware Security for Resource Selection in Grid Computing*, IJCSNS International Journal of Computer Science and Network Security, VOL. 8 No. 11, November 2008, pp. 107–115.

Edited by: Costin Bădică and Viorel Negru
Received: December 20th, 2010
Accepted: December 31st, 2010

Scalable Computing: Practice and Experience
Volume 11, Number 4, pp. 393–399. http://www.scpe.org
ISSN 1895-1767

A GRAPH-BASED APPROACH TO CONTEXT MATCHING*

ANDREI OLARU† AND ADINA MAGDA FLOREA†

Abstract. This paper presents the work in progress towards a simple, flexible and decentralized representation of context and for the detection of appropriate context-aware action. Continuing our previous work on decentralized multi-agent systems for the context-aware exchange of information, we propose a representation for context inspired from concept maps and conceptual graphs, and also a formalism for context patterns, that allows the detection and solution of problems related to the user's context.

Key words: knowledge representation, multi-agent systems, pattern recognition

1. Introduction. Domains like Ubiquitous Computing [27] and Ambient Intelligence [8] have brought context-awareness as a central issue for research. As the electronic environment that surrounds people must assist them in more and more of their daily activities, ambient applications must consider more aspects of the user's context in order to improve their performance and their adequateness to the principles of UbiComp and AmI.

A true AmI system must be non-intrusive, but also proactive. This is a balance that may be difficult to reach. Appropriate proactive action must fit the context of the user or the user will not have an optimal experience with the system [23]. Additionally, in order to be useful, many times proactive action must result from the anticipation, based on the current context, of future situations. Context-aware action and anticipation will also make an AmI system seem "intelligent" [4, 22].

There is a large body of research dealing with the subject of context-awareness. Context has been defined as [6]: "Any information that can be used to characterize the situation of entities (i. e. whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves". Most works deal with context as location, location and time or other physical conditions, like temperature [1, 21, 10]. Another aspect of context that is considered in some works is activity [12, 14].

However, most times the situation of the user is defined only by means of aspects that have been predefined. Situations may be described by means of certain types of associations [12] or by means of ontologies and / or rules [21]. However, these methods are not very flexible and the range of contexts that can be treated becomes limited with respect to what Ambient Intelligence should ideally be.

In our approach towards an implementation of Ambient Intelligence, we are trying to build mechanisms and representations that facilitate a more flexible approach to AmI and context-awareness, while in the same time are easy to implement and can work on resource-constrained devices. In previous work a decentralized multi-agent system for the distribution of information was built, that shared relevant information between agents, based on some simple measures of context-awareness [19]. However, context-awareness requires a more complex and powerful representation of context, while less capable devices require that this representation be flexible in size and also easy to process.

This paper deals with describing a simple formalism that allows agents in a multi-agent system, that have only local knowledge, to share and process context-related information and to solve problems by using *context matching*. We consider that there is one agent assigned to each user of the system (we call the system AmIciTy and the agents AmIciTy agents). Each agent has a representation of the context of its user, including models on other users. The focus of this paper is more on defining a manner of representation, and less on the algorithms used for context matching, the agent's behaviour or the protocol used in the communication between agents.

Context matching is based on representing information about the context of a user as a conceptual graph [24] and on the existence of context patterns, or, in short, patterns. Patterns can describe (in more or less detail) situations that the user has been in, they can describe common sense knowledge, or just associations between different pieces of information that are observed to appear frequently in the user's history. They resemble the

*This work has been supported by CNCSIS–UEFISCSU, project number PNII–IDEI 1315/2008 and by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/6/1.5/S/16.

†Computer Science Department, University Politehnica of Bucharest, 313 Splaiul Independentei, 060042 Bucharest, Romania. {cs, adina}@andreiolaru.ro.

notion of pattern in knowledge discovery in databases. In fact, patterns may be extracted by using data mining techniques, but this is not the focus of this paper.

Two matching context graphs (for two different users) mean a shared context, which is an occasion for further sharing of information between the users' agents. A pattern that matches the user's context means the user has been in the situation before (or it is a well-known, commonsense, situation) and this can allow the agent to anticipate possible problems, as well as to find solutions to problems already detected.

The next section presents related work in the field of context-awareness. The proposed context representation, as well as a scenario and examples of context-aware behaviour form the content of Section 3. The last two sections give an insight on future work and draw the conclusions.

2. Related Work. In previous work in the field of context-awareness there are usually two points of focus: one is the architecture for capturing context information; the other is the modeling of context information and how to reason about it.

Ever since the first works on context-awareness for pervasive computing [7], certain infrastructures for the processing of context information have been proposed [13, 11, 15, 12, 1, 9]. There are several layers that are usually proposed, going from sensors to the application: sensors capture information from the environment, there is a layer for the preprocessing of that information, the layer for its storage and management, and the layer of the application that uses the context information [1]. This type of infrastructures is useful when the context information comes from the environment and refers to environmental conditions like location, temperature, light or weather. However, physical context is only one aspect of context [5]. Moreover, these infrastructures are usually centralized, using context servers that are queried to obtain relevant or useful context information [7, 15]. In our approach [18], we attempt to build an agent-based infrastructure that is decentralized, in which each agent has knowledge about the context of its user, and the main aspect of context-awareness is based on associations between different pieces of context information.

Modeling of context information uses representations that range from tuples to logical, case-based and ontological representations [21, 25]. These are used to determine the situation that the user is in. Henricksen et al use several types of associations as well as rule-based reasoning to take context-aware decisions [12, 3]. However, these approaches are not flexible throughout the evolution of the system – the ontologies and rules are hard to modify on the go and in a dynamical manner. While ontologies make an excellent tool of representing concepts, context is many times just a set of associations that changes incessantly, so it is very hard to dynamically maintain an ontology that describes the user's context by means of a concept. In this paper we propose a more simple, but flexible and easy-to-adapt dynamical representation of context information, based on concept maps and conceptual graphs. While our representations lacks the expressive power of ontologies in terms of restrictions, a graph-based representations is very flexible and extensible, so support for restriction may be added as future work.

Our approach to context representation is rooted in existing knowledge representation methods like semantic networks, concept maps [17] and conceptual graphs [24]. These structures can be used to describe situations (and context) in a more flexible manner and using less memory than ontological representations. While graph matching has been previously used, for instance for image processing [2], we attempt to use it for the matching of context graphs, also improving the graphs by means of special notation elements that allow the definition of patterns.

There has been a significant body of work in the domain of ontology alignment, which is vital for a viable implementation of Ambient Intelligence systems [26]. However, this is not the subject of this paper. We assume that all agents in the system work with terms from the same ontology (where it is the case), or that ontologies have already been aligned.

Software agents and multi-agent systems have been used many times in the implementation of AmI environments in the past [15, 4, 16], however context-awareness in these systems is limited to location-related associations and simple forms of representation.

3. Context Matching. The main idea of this paper is to propose a graph representation for contexts, designed in order to facilitate the detection of compatible contexts.

Scenario. Alice will go to a rock concert in the evening of the current day. The concert is located at a stadium outside the city, therefore she should find some means of transportation to get there, but she hasn't yet given thought about that. Bob, her roommate, will go to the same concert but he has not talked to Alice about that yet. However, he has already booked a taxi to get to the concert. This is a typical situation for our

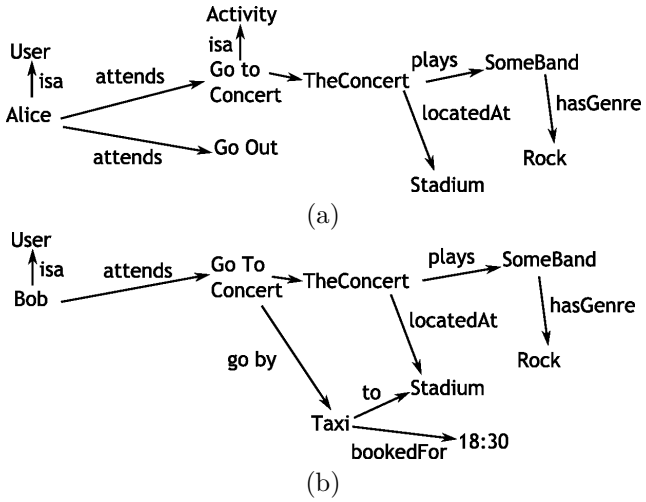


FIG. 3.1. *The knowledge of Alice's and Bob's agents, respectively: (a) Alice will go to a concert where a rock band is playing and which is located at the stadium, and then to go out with friends; (b) Bob will be going at the same concert, and has also booked a taxi to get there. Part of the relations and concepts that appear in the graphs may come from ontologies.*

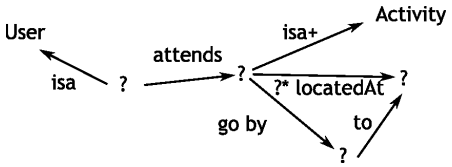


FIG. 3.2. *A pattern that says that if the user attends an activity that has a location, then the location of the activity should be reached in some way.*

approach [18]: insufficient communication between people leads to a lack of otherwise relevant information that could be easily obtained by means of an AmI system.

Alice and Bob are both users of the AmIciTy Ambient Intelligence system. What we want is that the system (1) detects the need for a means of transportation for Alice, (2) based on information on Bob's agenda, suggest that a taxi may be an appropriate solution for Alice as well, and (3) based on the existing shared context, propose to Alice that she uses the same taxi that Bob has already booked.

Each user of AmIciTy has an associated agent. Figure 3.1 shows the concept graph for the knowledge of the two agents (agent A for Alice and agent B or Bob) that is relevant to the scenario. Formally, the knowledge of each agent can be represented as a graph:

$G = (V, E)$

$V = \{v_i\}, E = \{e_k\}, e_k = (v_i, v_j, value)$

where $v_i, v_j \in V, i, j = \overline{1, n}, k = \overline{1, m}$

The values of vertices and edges can be either strings or, better, URI identifiers that designate concepts, relations, people, etc. The value of an edge may be null.

The graph that an agent has contains the knowledge that the agent has about the user and about the user's context. The graph represents the context of the user, in the measure in which the agent has perceived it (or been informed of by the user or by another agent).

First, we want the system to detect the fact that it is necessary to know how Alice will be getting at the concert. This can be done by means of the following pattern: if the user intends to attend something that is an activity, and that has a location (it's not, for instance, making a phone call), then there should also be a means for the user to reach that location. The pattern is represented in Figure 3.2.

A pattern is also a graph, but there are several additional features that makes it match a wider range of situations. The graph for a pattern s is defined as:

$G_s^P = (V_s^P, E_s^P)$

$V_s^P = \{v_i\}, v_i = string \mid URI \mid ?, i = \overline{1, n}$

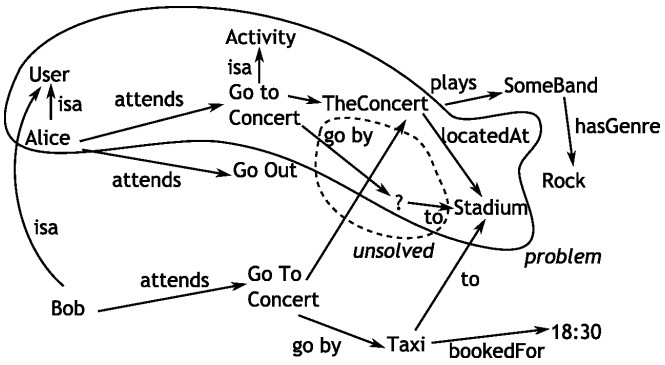


FIG. 3.3. *The knowledge base of agent A, completed with the information on Bob's agenda. Also the problem and its unsolved part are circled with a continuous and a dashed line respectively. Although the unsolved part is displayed together with the rest of the context graph, it is not a concrete or known fact so it would not be used in pattern-matching.*

$E_s^P = \{e_k\}, e_k = (v_i, v_j, E\_RegExp), v_i, v_j \in V_s^P, k = \overline{1,m}$
where E_RegExp is a regular expression formed of strings or URIs.

A *pattern* represents a set of associations that has been observed to occur many times and that is likely to occur again. Patterns may come from past perceptions of the agent on the user's context or be extracted by means of data mining techniques from the user's history of contexts. Commonsense patterns may come from public databases, and patterns may also be exchanged between agents. However, the creation or extraction of patterns is not the subject of this paper.

The agent has a set of patterns that it matches against the current context (graph G). We will mark with the ^{P} superscript the graphs or vertex / edge sets that contain special pattern features (like ? nodes, for instance).

A pattern G_s^P *matches* a subgraph G' of G, with $G' = (V', E')$ and $G_s^P = (V_s^P, E_s^P)$, iff an injective function $f : V_s^P \to V'$ exists, so that

(1) $\forall v_i^P \in V_s^P, v_i^P =?$ or $v_i^P = f(v_i^P)$ (*same value*)

and

(2) $\forall e^P \in E_s^P, e^P = (v_i^P, v_j^P, value)$ we have:

if *value* is a string or an URI, then the edge $(f(v_i^P), f(v_j^P), value) \in E'$

if *value* is a regular expression, then it matches the values $value_0, value_1, \ldots, value_p$ of a series of edges $e_0, e_1, \ldots, e_p \in E'$, where

$e_0 = (f(v_i^P), v_{a_0}, value_0)$,

$e_k = (v_{a_{k-1}}, v_{a_k}, value_1), k = \overline{1, p-1}$

$e_p = (v_{a_{p-1}}, f(v_j^P), value_p)$,

$v_{a_l} \in V'$.

In other words, every non-? vertex from the pattern must match a different vertex from G'; every non-RegExp edge from the pattern must match an edge from G'; and every regular expression edge from the pattern must match a series (that can be void, if the expression allows it) of edges from G'. Subgraph G' should be *minimal*. A graph (or subgraph) G' is minimal with respect to a matching pattern G_s^P iff there is no edge in G' that is not the match (or part of the match) of an edge in G_s^P.

A pattern G_s^P *k-matches* a subgraph G' of G, if condition (2) above is fulfilled for $m - k$ edges in E_s^P, $k \in [1, m-1]$, $m = ||E_s^P||$ and G' remains connected and minimal. The relationship of *k-matching* should be interpreted as *matching except for k edges*. Non-matching vertexes imply non-matching edges. We consider the number of edges as relevant (as opposed to number of vertices, for instance), because context is a set of associations, so it is the edges that matter.

For the example in Figure 3.1 (a), the pattern in Figure 3.2 2-matches the knowledge about Alice that her agent has, G' containing the information that the user (Alice) will attend a concert (which is an activity) which is located at the stadium. The missing edges are the *go by* edge and the *to* edge, and there is also a vertex that is missing – the means of transportation. Because the pattern fits in a percentage of 66%, it means that Alice is in the situation described by the pattern, but something is missing, so the agent should ask Alice about that piece of information or to try to find it itself (see also Figure 4.1). This is defined as a *problem*.

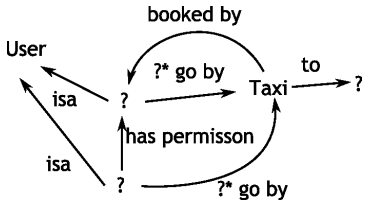


FIG. 3.4. *A second pattern, specifying that two people can use the same taxi to get to the same location if the person who has not booked the taxi has permission from the other to ride the same taxi.*

```
pattern G_s^P k − matches G →
    if k > 0
        put problem in the list
    if there is a problem and this can be a solution
        is the solution is certain / complete
            let user know
            user confirms solution
                increase confidence in pattern
            otherwise
                decrease confidence in pattern
        otherwise
            link possible solution to problem
```

FIG. 4.1. *Pseudocode of the agent's behaviour related to context matching.*

4. Problem Solving. A *Problem* is a graph G^P that contains features that are specific for patterns (like ? nodes for instance) and that is a partial instantiation of a pattern G_s^P, according to the current context. A problem G^P is the union between the subgraph G' (of the context graph G) that k-matches pattern G_s^P and the part of G_s^P that is not matched by G'. The latter is the *unsolved* part of the problem. A problem also remains associated with the pattern that generated it. Therefore, formally, if a pattern $G_s^P = (V_s^P, E_s^P)$ k-matches the subgraph $G' = (V', E')$ of G, we can define a problem p as a tuple (G_s^P, G_p^P), where G_p^P is the problem's graph:

$G_p^P = G' \cup G_x^P$
$G_x^P = (V_x^P, E_x^P)$
$V_x^P = \{v \in V_s^P, v \notin dom(f)\}$
$E_x^P = \{e \in E_s^P for\ which\ condition\ (2)\ is\ not\ fulfilled\}$

Note that G_x^P (the unsolved part of the problem) is a subgraph of G_s^P. Also note that the unsolved part may contain edges whose vertices are not both in the unsolved part. The *problem* from our example is circled in Figure 3.3 with a continuous line, and its unsolved part is circled with a dashed line.

The agents in AmIciTy are not single agents. They are part of a multi-agent system. Agents A and B communicate frequently due to the fact that Alice and Bob live in the same place and exchange a lot of data. At some point in this communication, they exchange data about Alice's and Bob's agendas, which is normal for two people that share an apartment. Agent B will send the subgraph $agenda \rightarrow Concert$ and A will send $agenda \rightarrow Concert/ \rightarrow Go\ Out$ (agent A will only send the *GoOut* activity if Alice has not designated it as private).

Agent A receives the subgraph $agenda \rightarrow Concert$ and matches it against Alice's context, detecting the compatibility (a full match). So it responds by building upon this common context: it sends a larger subgraph, containing the band playing at the concert, as well as the location of the concert. Agent B does the same operations as A (they share the same context regarding the concert, so each one's context matches the other one's), just that it also sends to A the associations $Concert - go\ by \rightarrow Taxi - to \rightarrow Stadium$.

The communication between agents as described above is done based on shared context. Starting from sharing their agendas, at each step agents detect matches between the two contexts and respond with a subgraph that is larger with one level (breadth-first).

All the data that agent A has about other agents (here, agent B) is stored in the agent's knowledge base as its model of the other users. The model for the other users is not necessarily separate though: if the same concept appears in both models (provided the concept ha the same URI, or the agent is able to detect by means of common sense knowledge that it is the same concept), both subgraphs will contain the corresponding node. Figure 3.3 shows the knowledge of agent A regarding users Alice and Bob. The model for Bob's agenda contains the same *Concert* node that is contained in the graph for Alice. When matching patterns from its pattern set, A detects that the pattern mentioned above fully matches the model for Bob. Agent A also has a *problem* that is linked to this pattern. Since Bob's context fully matches the pattern, it means it may be a *solution* to Alice's problem: Alice may also use a taxi to reach the concert. But that would mean booking a different taxi (use a different instance of the concept).

Another pattern may be used in this context: agent A may know that two people may share the same taxi to get to the same destination, if one has permission from the person that booked the taxi (we have somewhat simplified the problem and we do not mention that the two people must leave from the same location and need to reach the destination at the same time). This pattern is shown in Figure 3.4. Matching this pattern against the knowledge of agent A about Alice's context in Figure 3.3 (remember that unsolved parts are not matched), a 2-match is obtained (missing relations are *has permission* and Alice's *go by*). Not only that, but adding those relation would solve the problem that Alice has. Therefore, the agent can suggest to Alice to ask permission from Bob to use the same taxi.

In this particular case, with the given knowledge and patterns, there is only one solution to the problem that arose. But in a more realistic case, where context is more complex and there are more patterns, more solutions to the same problem may be found. In case they fit equally well in the current context, then the agent must prompt the user with all of them and the user must be given the choice.

It can be argued that context-matching is a very difficult problem in the case of large graphs and complex situations. However, resource-constrained devices will work only with smaller pieces of context information (i. e. smaller graphs), that are relevant to their function. Second, algorithms inspired from data-mining allow for incremental matching, starting from common nodes and growing the matching sub-graph (similar to the algorithm for matching Rule Schemas [20]).

Another problem that may appear in realistic situations (as opposed to our simple example) is the abundance of simultaneous matching context patterns, possibly describing contradictory situations. This is where more refined measures must be found that will allow calculating the relevance of each match. This too will be part of our future work.

5. Future Work. The work presented in this paper is in progress. We are in the process of identifying an efficient matching algorithm, as well as deploying the described formalism into a previously implemented multi-agent system.

Besides detecting compatible contexts, there is a very interesting potential in detecting incompatible contexts, or contexts that the user should not be in. Also, uncertainty has yet to be included in our work. Both these problems have been researched in the domain of conceptual graphs and graph matching.

One last question that must be further researched is if a subgraph k-matching a pattern should really be connected. Moreover, should a pattern be necessarily connected, and how could unconnected patterns and matches be interpreted.

6. Conclusion. So far, work in context-awareness for pervasive environments has been based predominantly on location-awareness and physical conditions. The use of ontologies or rules does not bring much dynamical flexibility and they are not easy to modify automatically, at runtime.

This paper presents the work in progress towards the development of a more simple – suitable for resource-constrained devices – and more flexible manner of representing context and context patterns, that allows the agents to take decisions without the need for a centralized structure, by means of their knowledge, their history and local communication alone.

REFERENCES

[1] M. Baldauf, S. Dustdar, and F. Rosenberg, *A survey on context-aware systems*, International Journal of Ad Hoc and Ubiquitous Computing, 2 (2007), pp. 263–277.

[2] E. Bengoetxea, P. Larrañaga, I. Bloch, A. Perchant, and C. Boeres, *Inexact graph matching by means of estimation of distribution algorithms*, Pattern Recognition, 35 (2002), pp. 2867–2880.

[3] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni, *A survey of context modelling and reasoning techniques*, Pervasive and Mobile Computing, 6 (2010), pp. 161–180.
[4] G. Cabri, L. Ferrari, L. Leonardi, and F. Zambonelli, *The LAICA project: Supporting ambient intelligence via agents and ad-hoc middleware*, Proceedings of WETICE 2005, 14th IEEE International Workshops on Enabling Technologies, 13-15 June 2005, Linköping, Sweden, (2005), pp. 39–46.
[5] G. Chen and D. Kotz, *A survey of context-aware mobile computing research*, Technical Report TR2000-381, Dartmouth College, November 2000.
[6] A. Dey, *Understanding and using context*, Personal and ubiquitous computing, 5 (2001), pp. 4–7.
[7] A. Dey, G. Abowd, and D. Salber, *A context-based infrastructure for smart environments*, Proceedings of the 1st International Workshop on Managing Interactions in Smart Environments (MANSE'99), (1999), pp. 114–128.
[8] K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J. Burgelman, *Scenarios for ambient intelligence in 2010*, tech. report, Office for Official Publications of the European Communities, February 2001.
[9] L. Feng, P. M. G. Apers, and W. Jonker, *Towards context-aware data management for ambient intelligence*, in Proceedings of DEXA 2004, 15th International Conference on Database and Expert Systems Applications, Zaragoza, Spain, August 30 - September 3, F. Galindo, M. Takizawa, and R. Traunmüller, eds., vol. 3180 of Lecture Notes in Computer Science, Springer, 2004, pp. 422–431.
[10] H. Hagras, V. Callaghan, M. Colley, G. Clarke, A. Pounds-Cornish, and H. Duman, *Creating an ambient-intelligence environment using embedded agents*, IEEE Intelligent Systems, (2004), pp. 12–20.
[11] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, *The anatomy of a context-aware application*, Wireless Networks, 8 (2002), pp. 187–197.
[12] K. Henricksen and J. Indulska, *Developing context-aware pervasive computing applications: Models and approach*, Pervasive and Mobile Computing, 2 (2006), pp. 37–64.
[13] J. Hong and J. Landay, *An infrastructure approach to context-aware computing*, Human-Computer Interaction, 16 (2001), pp. 287–303.
[14] M. Kaenampornpan and E. O'Neill, *An integrated context model: Bringing activity to context*, in Proceedings of the Workshop on Advanced Context Modelling, Reasoning and Management, 2004, pp. 7–10.
[15] T. C. Lech and L. W. M. Wienhofen, *AmbieAgents: a scalable infrastructure for mobile and context-aware information services*, Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), July 25-29, 2005, Utrecht, The Netherlands, (2005), pp. 625–631.
[16] C. Muldoon, G. M. P. O'Hare, R. W. Collier, and M. J. O'Grady, *Agent factory micro edition: A framework for ambient applications*, in Proceedings of ICCS 2006, 6th International Conference on Computational Science, Reading, UK, May 28-31, V. N. Alexandrov, G. D. van Albada, P. M. A. Sloot, and J. Dongarra, eds., vol. 3993 of Lecture Notes in Computer Science, Springer, 2006, pp. 727–734.
[17] J. D. Novak and A. J. Cañas, *The origins of the concept mapping tool and the continuing evolution of the tool*, Information Visualization, 5 (2006), pp. 175–184.
[18] A. Olaru, A. El Fallah Seghrouchni, and A. M. Florea, *Ambient intelligence: From scenario analysis towards a bottom-up design*, in Proceedings of IDC'2010, the 4th International Symposium on Intelligent Distributed Computing, M. Essaaidi, M. Malgeri, and C. Badica, eds., vol. 315 of Studies in Computational Intelligence, Springer, 2010, pp. 165–170.
[19] A. Olaru, C. Gratie, and A. M. Florea, *Context-aware emergent behaviour in a MAS for information exchange*, Scalable Computing: Practice and Experience - Scientific International Journal for Parallel and Distributed Computing, 11 (2010), pp. 33–42. ISSN 1895-1767.
[20] A. Olaru, C. Marinica, and F. Guillet, *Local mining of association rules with rule schemas*, in Proceedings of CIDM 2009, the IEEE Symposium on Computational Intelligence and Data Mining, March 30 - April 2, Nashville, TN, USA, IEEE Symposium Series on Computational Intelligence, 2009, pp. 118–124.
[21] M. Perttunen, J. Riekki, and O. Lassila, *Context representation and reasoning in pervasive computing: a review*, International Journal of Multimedia and Ubiquitous Engineering, 4 (2009), pp. 1–28.
[22] C. Ramos, J. Augusto, and D. Shapiro, *Ambient intelligence - the next step for artificial intelligence*, IEEE Intelligent Systems, 23 (2008), pp. 15–18.
[23] G. Riva, F. Vatalaro, F. Davide, and M. Alcañiz, eds., *Ambient Intelligence*, IOS Press Amsterdam, 2005.
[24] J. Sowa, *Knowledge representation: logical, philosophical, and computational foundations*, MIT Press, 2000.
[25] T. Strang and C. Linnhoff-Popien, *A context modeling survey*, Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp, (2004), pp. 1–8.
[26] J. Viterbo, L. Mazuel, Y. Charif, M. Endler, N. Sabouret, K. Breitman, A. El Fallah Seghrouchni, and J. Briot, *Ambient intelligence: Management of distributed and heterogeneous context knowledge*, CRC Studies in Informatics Series. Chapman & Hall, (2008), pp. 1–44.
[27] M. Weiser, *The computer for the 21st century*, Scientific American, 272 (1995), pp. 78–89.

Edited by: Costin Bădică and Viorel Negru
Received: December 20th, 2010
Accepted: December 31st, 2010

Scalable Computing: Practice and Experience
Volume 11, Number 4, pp. 401–401. http://www.scpe.org
ISSN 1895-1767

BOOK REVIEWS

EDITED BY JIE CHENG

CUDA by Example: An Introduction to General-Purpose GPU Programming
Jason Sanders and Edward Kandrot
ISBN-13: 978-0131387683
Addison-Wesley Professional; 1 edition (July 29, 2010)

1. Introduction. This book is designed for readers who are interested in studying how to develop general parallel applications on graphics processing unit (GPU) by using CUDA C. CUDA C is a programming language, which combines industry standard programming C language and some more features which can exploit CUDA architecture. With proper introduction to NVIDA's CUDA architecture and in depth explanation for setting up development environment, this book is an easy to read, easy to understand, and hands on book. Readers of this book are assumed to have at least C language as background. Through this book, readers will not only gain experience in CUDA C development languages, but also will understand a lot of important underlying hardware knowledge, which in return can help software developers develop more efficient and effective applications.

2. Outline of the Book. This book is very well organized. Each chapter consists of general introduction, chapter objectives and Chapter Review. Both Sanders and Kandrot are senior software engineers in the CUDA Platform group and CUDA Algorithm team in NVIDIA Company, respectively.

First chapter provide users background about history of GPU and CUDA architecture. Special features in CUDA architecture enable GPU to perform general purpose computation in addition to carry out traditional graphic computation. Readers can easily understand the benefit of CUDA architecture by reading though three different applications varying from medical field to environmental filed. In Chapter 2, Sanders and Kandrot equip users with complete lists of hardware and software support for running CUDA C applications. All software can be downloaded for free from websites suggested from authors. Then by a familiar Hello world program in Chapter 3, authors demystified that the CUDA C fundamentally is a standard C language with additional features which can allow application developer to specify which code can be run on device (GPU and its memory) or host (CPU and system memory). After setting all of proper background, the use of CUDA C to run parallel programs on GPU are discussed from Chapter 4 to Chapter 7. In Chapter 8, authors try to illustrate how to incorporate rendering and general purpose computation by using CUDA C. Readers without background in OpenGL or DirectX, can skip this chapter and go to the next. However, this chapter is a great addition to the book since it gives readers complete view of CUDA C. Even though CUDA C turns complicated application with single thread execution into easier case by parallel processing, there are some situation that special care should be taken when simple single thread application are tried to implement on massively parallel architecture; Chapter 9 discusses this topic. Compared to parallelism discussed in above chapters, which refers to parallel execution of a function on different sets of data, in Chapter 10, readers are exposed to a different class of parallelism on GPU, which refers to two or more completely independent tasks to be performed in parallel. Chapter 11 covers how to develop CUDA C application on Multiple GPUS. For further study, Chapter 12 shows more tools to aid CUDA C development and more resources to enhance reader's CUBA C development skills to another level.

3. Summary. Jason Sanders and Edward Kandrot wrote this book in such a way that is very easy to read and follow. Also, Sanders and Kandrot never forget the great sense of humor throughout the book. Reading this book is not only a discovery about CUDA C but also a joyful journal. It is highly recommended for students who are interested to learn CUDA C application development as Computer Science major. This book is recommended to be adopted as textbook for undergraduate students studying parallel programming.

Jie Cheng,
University of Hawaii Hilo

AIMS AND SCOPE

The area of scalable computing has matured and reached a point where new issues and trends require a professional forum. SCPE will provide this avenue by publishing original refereed papers that address the present as well as the future of parallel and distributed computing. The journal will focus on algorithm development, implementation and execution on real-world parallel architectures, and application of parallel and distributed computing to the solution of real-life problems. Of particular interest are:

Expressiveness:

- high level languages,
- object oriented techniques,
- compiler technology for parallel computing,
- implementation techniques and their efficiency.

System engineering:

- programming environments,
- debugging tools,
- software libraries.

Performance:

- performance measurement: metrics, evaluation, visualization,
- performance improvement: resource allocation and scheduling, I/O, network throughput.

Applications:

- database,
- control systems,
- embedded systems,
- fault tolerance,
- industrial and business,
- real-time,
- scientific computing,
- visualization.

Future:

- limitations of current approaches,
- engineering trends and their consequences,
- novel parallel architectures.

Taking into account the extremely rapid pace of changes in the field SCPE is committed to fast turnaround of papers and a short publication time of accepted papers.

INSTRUCTIONS FOR CONTRIBUTORS

Proposals of Special Issues should be submitted to the editor-in-chief.

The language of the journal is English. SCPE publishes three categories of papers: overview papers, research papers and short communications. Electronic submissions are preferred. Overview papers and short communications should be submitted to the editor-in-chief. Research papers should be submitted to the editor whose research interests match the subject of the paper most closely. The list of editors' research interests can be found at the journal WWW site (`http://www.scpe.org`). Each paper appropriate to the journal will be refereed by a minimum of two referees.

There is no a priori limit on the length of overview papers. Research papers should be limited to approximately 20 pages, while short communications should not exceed 5 pages. A 50–100 word abstract should be included.

Upon acceptance the authors will be asked to transfer copyright of the article to the publisher. The authors will be required to prepare the text in LaTeX 2$_\varepsilon$ using the journal document class file (based on the SIAM's `siamltex.clo` document class, available at the journal WWW site). Figures must be prepared in encapsulated PostScript and appropriately incorporated into the text. The bibliography should be formatted using the SIAM convention. Detailed instructions for the Authors are available on the SCPE WWW site at `http://www.scpe.org`.

Contributions are accepted for review on the understanding that the same work has not been published and that it is not being considered for publication elsewhere. Technical reports can be submitted. Substantially revised versions of papers published in not easily accessible conference proceedings can also be submitted. The editor-in-chief should be notified at the time of submission and the author is responsible for obtaining the necessary copyright releases for all copyrighted material.