2013 IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing

An Efficient Routing Scheme for Overlay Network of SOAP Proxies in Constrained Networks

Muhammad Ali, Hailong Sun, Wei Yuan
School of Computer Science and Engineering
Beihang University
Beijing, China
ali_phy@hotmail.com, {sunhl, yuanwei}@act.buaa.edu.cn

***Abstract*—An overlay network of SOAP proxies over constrained and ad hoc networks of military, law enforcement and emergency response is usually established for reliable access of standard Web services. SOAP messages are stored and forwarded reliably from node to node using a reliable message oriented transport. For dynamic access of Web services across participating networks, a SOAP proxy node needs to find the next SOAP proxy node towards destination. In this paper, first we propose an efficient routing scheme considering structure and constraints of underlying networks of higher delay, frequent packet loss and limited bandwidth. Second, we propose the use of logical hierarchical names for overlay nodes which helps minimizing routing overheads and simplifies management of addresses. Through a limited scope proactive routing our routing scheme finds routes to nodes on adjacent networks. Whereas it uses hierarchical node names to derive routes to farther destinations on hierarchically deployed operational networks. Proposed scheme has been evaluated by analyzing routing overhead with variation of total number of nodes on participating networks. Simulation results demonstrate a good performance of our routing scheme for overlay network of SOAP proxies.**

Keywords—Web Services; SOAP Proxy; Overlay Routing; Constrained Networks; Overlay Networks;

I. INTRODUCTION

Service Oriented Architecture (SOA) offers features to design and build software components in the form of services which are reused for dynamic composition of software systems. SOA is generally implemented through standard Web services by exposing business functions through standardized interfaces. Loose coupling, between services and service clients, along with standard interfaces are prominent features of Web services. Such features are useful for dynamic integration and interoperability of software systems [1]. SOA is commonly implemented in enterprises with deployment of Web services usually over Internet or stable enterprise networks [2].

A. Background

Departments like Defence, Law Enforcement or Public Security also have requirement of net centric software applications with the features of dynamic integration and interoperability with other software [3]-[5]. Implementing SOA through standard Web services in their operational environments is not similar to that of ordinary environments. Unlike stable Internet or enterprise networks, constrained networks characterized by higher delays, frequent disconnections and packet losses are not supportive for expected performance of Web services to realize SOA [2], [6]. SOAP communication which is the backbone of Web services is required to be reinforced to withstand such conditions for stable access of services. Researchers have already put forward their ideas and schemes to handle such difficulties as in [6], [7], [8], and [9]. These ideas are generally about to replace or augment the underlying TCP based HTTP communication with a reliable message based transport for SOAP messages. The message based communication is considered to be suitable for constrained networks when carried out with reliable delivery [6]. Usually SOAP proxies are used at service client as well as service nodes to bridge standard HTTP/TCP based SOAP requests and responses of legacy Web services and clients, across constrained network as shown by a model in Fig.1. For a typical Web service request, the service client at node A, uses HTTP/TCP based communication with locally running SOAP proxy which transfers SOAP request message, over unreliable network, directly to the SOAP proxy at Web service node B using a reliable message based communication. The request message is subsequently delivered to the Web service by its SOAP proxy through standard HTTP/TCP communication. A service response is delivered back to service client using the similar communication method in reverse order. This type of SOAP transport using end-to-end reliability control is useful when both client and Web service nodes are on same network. However when multiple constrained networks are involved between Web service and client, an end-to-end reliable transport of messages gets degraded due to increased packet loss, delays and disconnections along the route through multiple unreliable networks and routers. To deal with this problem SOAP messages can be stored at intermediate nodes along the route and forwarded reliably from one node to next node. An overlay network of SOAP proxy nodes is therefore established to use message based SOAP communication of Web services over multiple interconnected constrained networks as shown by a model in Fig. 2.

978-0-7695-5088-6/13 $31.00 © 2013 IEEE
DOI 10.1109/HPCC.and.EUC.2013.73

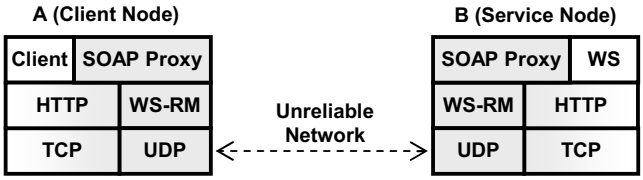

Figure 1. Web Service access over unreliable network using end-to-end SOAP Proxies.

Source node 'A' reliably delivers a SOAP message for destination node 'B', to its next intermediate SOAP proxy node 'R' along the route towards destination node. Intermediate node 'R' then reliably delivers this message to next node along the route. After reliable transport of given message through intermediate node(s), the original SOAP message is finally delivered to destination SOAP proxy node B. In [6] and [7], authors have presented the use of such overlay SOAP proxies for reliable Web services communication over constrained networks. They have also enabled the intermediate nodes along the route to store and forward SOAP messages for reliable delivery through multiple interconnected networks.

B. Research Motivation and Methodology

When an overlay network of SOAP proxy nodes is established over constrained networks for message oriented reliable delivery of SOAP messages, it raises requirement of routing decisions at such nodes. Any SOAP proxy while forwarding a message will need to select the next overlay SOAP proxy along the route to required destination SOAP proxy corresponding to required Web service. A dynamic routing scheme suitable for constrained networks will be required to carry out routing decisions under dynamic operational conditions. Along with routing scheme, SOAP proxy nodes will need to have an addressing scheme at overlay level for unique identification across all the participating networks. This is because all such nodes might not be having unique network addresses (excluding the case of adjacent networks) or nodes might be operating in heterogeneous networks of different types of network addresses.

Earlier work related to robust Web services is generally focused on reliable delivery of SOAP messages across the network(s) [7], [8], [9]. Although some of the earlier work [6], [7] has utilized a basic dynamic routing but its major contribution is towards the use SOAP proxies for reliable Web services. Application layer routing has not been well considered and routing scheme has not been applied considering the limitations of underlying networks. A basic distance vector like proactive routing has been used, considering a single flat network of connected overlay SOAP proxies and with the assumption of unique service names across all the participating networks. Using this kind of scheme, nodes on networks need to store routes to all other services available on networks which incurs a large overhead of routing updates using considerable share of meager bandwidth. This scheme will also require the explicit management of unique services names across all participating nodes on various networks. Although several algorithms related to network layer routing for ad hoc networks are available as surveyed in [10], but their use for overlay routing should well suit to nature, type and resources of the underlying networks for lower overheads from routing.

In our work we have considered above mentioned issues for dynamic SOAP transport through SOAP proxy nodes of overlay network over constrained networks. We have focused on the overly routing scheme among SOAP nodes with respect to dynamic aspect of network connectivity, nodes availability, resource constraints and lack of centralized management of participating networks especially regarding the nodes addressing. We present a routing scheme suitable for such networks with lower overheads and an addressing scheme for node names to simplify addresses management. Instead of using proactive or reactive routing only we have utilized a hybrid approach to minimize the routing overheads of table sizes and network updates. Lower routing overheads are required for the networks which do not support excessive data transmission and are constrained due to limited bandwidth, frequent losses and higher delays. It lowers the routing overhead by minimizing the scope of proactive routing by sharing the link state information only among nodes on networks which are directly connected according to operational hierarchy. Whereas, to fulfill the need for routing to networks which are not adjacent in hierarchy, it utilizes the hierarchical routing scheme based upon hierarchical node names.

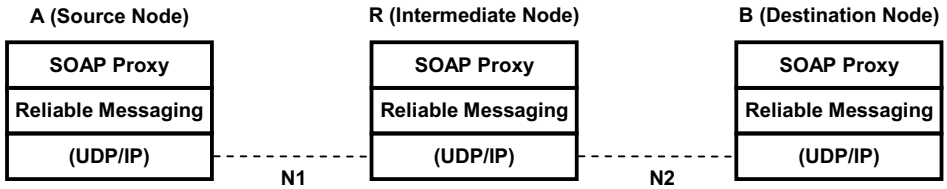

Figure 2. Overlay network of SOAP Proxies with store-and-forward capability at intermediate SOAP node.

Considering the hierarchical structure of networks used in operational command and control, we propose that hierarchical names can be used for identification of SOAP proxy nodes and these names can be taken from logical and standard role names of command hierarchy. It will simplify the process to standardize and manage node names or addresses across participating networks. Practically, hierarchical node names will need not to be resolved to actual network address (e.g IP) for communication.

Major contributions of this research work are as follows:

- A dynamic routing scheme for overlay network of SOAP proxies, suitable for unreliable and constrained networks. This scheme is proposed for a lower routing overhead in terms of routing table sizes and network updates, considering the structure and limitations of underlying networks.
- We propose the use logical hierarchical naming scheme to identify the SOAP nodes across overlay network. This scheme is based upon logical role names and hierarchical structure of operational command and control. Use of standard logical hierarchical names for SOAP nodes simplifies management of addresses across networks and is the basis of our routing scheme.

Routing scheme has been implemented in JAVA through an algorithm and its performance has been analysed for overheads using network simulation. It has also been compared with behaviour of ordinary proactive routing used to route packets among overlay nodes in earlier research work [7].

This paper is outlined as: Section II discusses earlier research related to use of SOA, Web services and dynamic routing with respect to unreliable networks. In sections III, target network structure along with the proposed addressing and routing schemes have been described. Section IV presents the routing algorithm, we implemented for performance evaluation. Finally in sections V and VI we describe our experiments along with results and conclude the work, respectively.

II. RELATED WORK

Research work [2], [3] - [5], and [11] is related to our work as it highlights the characteristics of constrained networks in use of military, law enforcement and emergency response agencies. This work encourages the use of SOA and Web services in such operational environments. This category of work, however, does not deal with the problems of Web services communication and routing issue.

Authors in [8] and [9] have enabled Web services to work in constrained networks by replacing underlying TCP transport with reliable message based transport for communication of SOAP messages. Message based communication is end-to-end between client and service. This type of work is not for Web services access across multiple interconnected unreliable networks. It does not consider messages store-and-forward through intermediate networks using overly network.

In [7], E. Skjervold et al. have created an overlay network of SOAP proxies with a store and forward capability for using Web services in constrained networks. However, this work is not particularly focused on routing among overlay proxy nodes. For dynamic routing they have used proactive routing scheme based upon unique service names across all the networks. Nodes need to keep routes to all the services available on participating networks through dynamic updates among nodes. For constrained networks our hybrid routing scheme reduces the overhead of storing routes to all destinations on network and also propagation of all such routes to all other nodes. As it is node names based so services created dynamically e.g. at client side in case of asynchronous access of Web services, can also be accessible without further routing updates.

MANET routing algorithms, referenced in [10], like Destination-Sequenced Distance-Vector (DSDV), Optimized Link State Routing (OLSR), Ad hoc On Demand Distance Vector (AODV), Dynamic Source Routing (DSR) and Temporally Ordered Routing Algorithm (TORA) are available for network layer routing with a typical classification as proactive routing protocols and reactive routing protocols. Proactive protocols maintain information about every other node in routing tables through routing updates for subsequent routing requests. Reactive protocols on the other hand do not store information about all other nodes beforehand and establish a route when it is required. In [12], G. Pei et al. have presented hierarchical network layer routing for wireless ad hoc networks in which participating nodes of one network operate collaboratively forming an overall hierarchical network of individual groups. Our work also considers similar structure of network but provides routing solution for an overlay of SOAP nodes at application level.

A use of only proactive routing will overload the routing tables with routes and will generate heavy network traffic for its updates to know about all the SOAP proxy nodes present on all networks. On the other hand only use of a routing scheme which is purely reactive in which a route is determined by broadcasting packets to every node around current SOAP node and waiting for final destination SOAP proxy for its reply to get the route, will also produce a heavy traffic load for such constrained networks. Any kind of scheme may have its pros and cons [10] and should not be used indiscriminately. Our objective is to utilize any one or hybrid of such schemes for a solution, considering the underlying structure and resources of constrained networks.

III. NETWORK STRUCTURE AND ROUTING SCHEME

In order to make a selection of routing scheme for target networks, structure and type of such networks should be studied. It is observed that networks deployed for military operations, disaster response or law enforcement operations are of ad hoc type, unreliable and may not have very high data rates [2]. A major characteristic of such deployments is that the networks are deployed according to functional hierarchy of working units or teams. Such networks may be simultaneously voice and data enabled through use of same radio sets. We can model the layout of such networks like a tree corresponding to command or functional hierarchy. Each functional group or unit usually has its own private network and participating nodes collaboratively operate or move together within in certain area of operation [12]. The group can have its link to

another network or group at upper level in functional hierarchy through a node which is usually the commander or head of group. Likewise any subordinate node in group may have a link with another network of its subsequent subordinates as commander or head of that network. Information travels across networks through head nodes of group leaders. When deployed, such networks map the functional hierarchy and the networks take the form of a hierarchically connected tree of small networks as shown by a model in Fig. 3. Network nodes have been represented by filled circles whereas unfilled circles represent groups' network, showing connectivity of related nodes through solid lines. Heavy solid lines show direct connectivity of nod C1 to other nodes B1,C2,D1, and D2.

A. Overlay Addressing

Participating group networks may be managed independently by respective groups and usually have lack of standardized or unique IPs across all the networks. To handle this addressing problem, considering the hierarchical layout of interconnected networks according to the functional hierarchy, we propose the use of functional call signs, logical names or abbreviations e.g. Commander, CO or GL as names of SOAP nodes. Because such role names are already standardized and unique across groups, when fully qualified according to hierarchy (e.g. CO of unit 2 of Alpha Division), there will be no need for special arrangements for unique addressing for overlay SOAP nodes.

As SOAP proxies will be operating at application level close to user level therefore for addressing purposes, SOAP nodes can utilize logical names mapped to functional position or role name of respective users in hierarchy. Overlay routing will no more be dependent on network layer addressing for unique network addresses across all the networks with the exception of adjacent networks. Having benefits of uniqueness and no additional standardization mechanism required for SOAP nodes addressing, such hierarchical addresses will also be useful for routing algorithm as described in next sections.

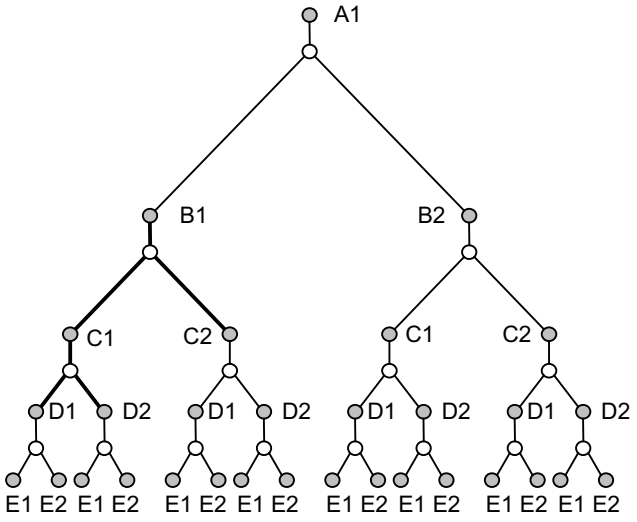

Figure 3. Hierarchically interconnected networks of individual groups with three nodes per group.

B. Basic Routing Scheme

Node names are unique within group and also across networks when fully qualified according to functional hierarchy. Participating units or groups operate in certain command hierarchy and so is their communication across the groups. The networks of individual groups are linked in the form of a tree as described earlier. Within a group, transmission is usually broadcast based and any node can communicate to others nodes on network. Across the groups information travels from one group (network) to other connected group (network) through group head node of group lower in hierarchy. It is suggested that if any node needs to communicate to any other node within or across the networks it can send message using that nodes fully qualified name as destination address. There will be only one route to destination and next hop along the route will be inferred from particular destinations address. However for delivery of message to a directly connected node, UDP datagram containing the SOAP message can be sent by using broadcast network address. Only that SOAP proxy node whose name matches the node name, set as next hop address in received SOAP message, will process the message. This mechanism eliminates the need for node name resolution to network address and therefore makes the scheme independent of network addresses.

The routing seems to be very simple as far as networks are linked in the form of a basic tree i.e. by having only one link to next upper or lower level network. No routing updates are required to be sent to neighboring SOAP nodes. However, a node may send visibility updates to inform neighbouring nodes within a group network about its presence on wireless network of group. When a packet needs to be sent to particular destination its next hop with respect to current SOAP node can be inferred easily. Next hop will be on same network (in case of last hop), on upper level network or at one of the lower level networks depending upon hierarchical destination address. Following same scheme at intermediate SOAP nodes along the route, message will ultimately reach the destination network following the hierarchical route and will be delivered to the required SOAP proxy node. If destination node would not be available the message would be considered unreachable. Therefore this type of deployment as shown in Fig. 3 needs very simple hierarchical routing even without any routing updates through neighbouring networks and complex routing tables.

C. Hybrid Routing Scheme

In practice, network deployment as shown by Fig. 3, may not be fail safe as there is dependency on a single link between any two adjacent networks through group head SOAP nodes. For example, referring to Fig. 3, if message from any subordinate node has to go to its upper level network and the group head node of its network is not operational, the message will not be delivered further up from this network.

For operational purposes, a group network usually can have a backup link or additional link to some other subordinate node (other than head node) of adjacent network lower in hierarchy. Using this kind of network layout we come up with network structure as shown in Fig. 4. In this figure, subordinate network of A1.B1 has an additional backup link to subordinate network

of A1.B1.C1 through subordinate node A1.B1.C1.D1 in addition to primary link to this network through node A1.B1.C1. Now adjacent networks may have two links between each other for redundancy against link failures. Such secondary links are show by dotted lines in Fig. 4. As a result, a node, e.g. A1.B1.C1 has got additional direct links to other networks as shown by heavy dotted lines.

Only hierarchical routing scheme using hierarchical node names as described earlier is not enough for this type of network topology. It needs to have some knowledge about the available redundant links through some proactive type of link state mechanism. At the same time it should be noted that underlying constrained networks may not support propagating information about all the nodes across all the networks due to higher delays, losses, and lower bandwidth. We have used algorithm to make a selection of next hop out of given redundant links or possible next hops along the route towards the destination. For links selection from redundant links to adjacent networks it makes use of link state sort of approach in which nodes are aware of their surrounding links to next networks. However, this picture is limited to adjacent networks only to limit the size of routing tables and updates. For overall routing across multiple networks or destinations not on adjacent networks, the algorithm makes use of previously described hierarchical routing based upon destination node name. Using a mix of both techniques we have devised the algorithm as described in following section.

IV. ROUTING ALGORITHM

For reliable delivery of SOAP messages between source and destination SOAP proxy nodes separated by multiple networks, store and forward capability will be provided at intermediate nodes on every intermediate network. For dynamic, ad hoc and larger networks instead of static routes configuration dynamic routing is required and if the networks are not supportive as well then the choice of routing technique should be such that it does not produce big overheads of traffic as well as routing table sizes.

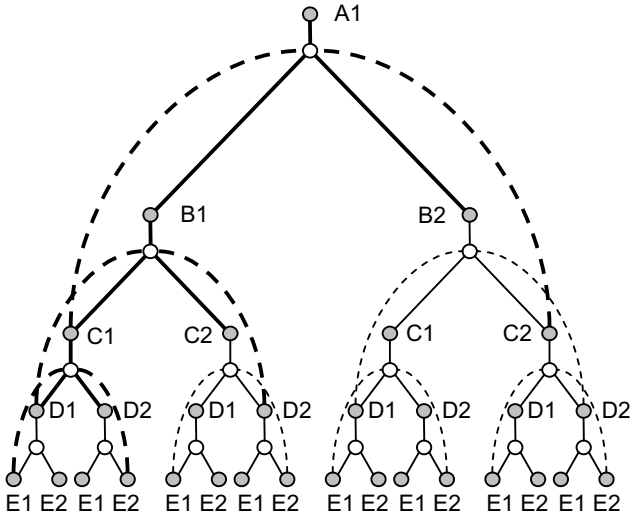

Figure 4. Hierarchically interconnected networks having additional backup links represented by dotted lines.

Considering the network layout, constraints and nature of communication for networks used for military, law enforcement and disaster recovery kind of operations we propose the use of following algorithm at a given SOAP proxy node for next hop selection towards the ultimate destination SOAP proxy node. The implementation of routing scheme consists of three parts

- Routing table update
- Updates to network
- Selection of next hop

We refer a network of one group by the logical name of its group head SOAP node, e.g. in Fig. 4, top level network is owned by node A1, being head of network. This network has subordinate nodes A1.B1 and A1.B2. Primarily, a node on a network may have two links: one to its parent or group head's network and one to its children or subordinates network which is its own network. Additionally, some of the nodes may be linked to three networks having an additional backup link to parent network of their corresponding parent or head node. For example, node A1.B1.C1 has a third direct connection to network of A1 through a backup link to parent network of its parent network, represented by dotted line. We refer third network as secondary network of node A1.B1.C1. Secondary links are shown by dotted lines in figure. Following notations are used in the algorithms

- *P(r)* and *gP(r)* represent parent and grand parent of node *r*, respectively.
- $C_i(r)$ and $gC_k(r)$ represent *ith* child and *kth* grand child of node *r*, respectively.
- $S_j(r)$ is *jth* sibling of node *r*

A. Routing Table Update

As a requirement of proactive part of our routing algorithm, link states for directly connected parent, children, and secondary networks are to be stored for link selection to next networks. Therefore each node in network has its routing table which is dynamically updated with updates from directly attached nodes. An update from a node will tell the presence of its source node at a particular network and additionally it would present a list of selected nodes, as described by section IV.B, presently connected to the source node of update. Routing information is stored in routing table for a particular interval after which it is deleted if not refreshed by subsequent periodic update from respective source node.

1) Routing Table Format: Routing table is populated with route records based upon routing updates received from connected networks. A route record consists of the following fields

- *NextHop:* Stores the name of directly connected node *n*
- *NeworkID:* Stores the ID of network through which node *n* is directly connected to *r*
- *RouteTo:* Stores the list of specific nodes which are directly reachable through node *n*

- *Timestamp:* When the route record was last updated

2) Update Algorithm: Following algorithm is used to update the routing table when a routing update is received from any of connected networks.

a) Input

- *n*: Node name of source of routing update
- *networkID*: Represents the network of received routing update
- *nodeList*: List of specific node names reachable through *n*. If routing packet is received from *secondaryNetork* then *nodeList = null*

b) Algorithm.

```
1:  For Each routing record R from routing table
2:    If R.NextHop equals n and R.NetworkID equals
        networkID then
3:      update R.RouteTo with nodeList
4:      update Timestamp
5:      quit
6:  Next For
7:  Create new routing record R using n, networkID, and
      nodeList
8:  Append R to routing table with current Timestamp
```

Nodelist for nodes connected through secondary networks is not stored in table to limit the size of routing table and links information is limited up to adjacent networks only.

B. Updates to Network

Nodes need to inform their presence on a particular network by sending periodic updates to their neighbouring nodes. Such updates are not large lists of routing records but contain the source node name and a list of some of the directly connected nodes (not all nodes to all networks) as given by following algorithm. List of connected nodes is created from the routing table information and it may be an empty list. In that case update will show only the presence of current node to other nodes on network. Following is the algorithm to send periodic updates to connected networks.

If *parentNetwork* exists **then**
Broadcast following information as routing update to *parentNetwork*
Source: *r*
Destination: *ToAll*
List of connected nodes: All $C_i(r)$ and $gP(r)$ for which node names are present in routing table as directly connected nodes. List may by empty or null
If *childrenNetwork* exists **then**
Broadcast following information as routing update to *childrenNetwork*
Source: *r*
Destination: *ToAll*
List of connected nodes: All $gC_k(r)$ and $P(r)$ for which node names are present in routing table as directly connected nodes. List may be empty or null.
If *secondaryNetwork* exists **then**
Broadcast following information as routing update to *secondaryNetwork*
Source: *r*
Destination: *ToAll*
List of connected nodes: *null*

C. Selection of Next Hop

Major part of the routing is routing decision based upon dynamically updated routing table and destination address. The following algorithm takes a destination address as input, consults the current routing table and returns the next hop or node address along the route to given destination. It returns empty if the next hop is presently not reachable as inferred from routing state from routing table.

a) Input and Output

- *d*: Destination Node
- *r*: Router Node
- *n*: Next Node *(output)*

b) Algorithm

```
1:  If d equals r then n = r
2:  Else If d is connected to r then n = d
3:  Else If d is offspring of r then
4:    If d equals Ci(r) then n = empty
5:    Else extract gCk(r) from d
6:      If gCk(r) is connected to r then n = gCk(r)
7:      Else if r has route to gCk(r) through connected
          P(gCk(r)) then n = P(gCk(r))
8:      Else n = any sibling of gCk(r) connected to r // it
          may also be empty.
9:  Else if d is Sj(r) then n = empty
10: Else if d is offspring of Sj(r) then
11:   Extract Ci(Sj(r)) from d
12:   If Ci(Sj(r)) is connected to r then n = Ci(Sj(r))
13:   Else if r has route to Ci(Sj(r)) through connected Sj(r)
        then n = Sj(r)
14:   Else if r has route to Ci(Sj(r)) or any sibling of Ci(Sj(r))
        through connected P(r) then n = P(r)
15:   Else n = empty
16: Else if d equals P(r) then n = empty
17: Else if d is offspring of gP(r) then
18:   Extract Ci(gP(r)) from d
19:   If Ci(gP(r)) is connected to r then n = Ci(gP(r))
20:   Else If Ci(gP(r)) not equals d then
21:     extract gCk(gP(r)) from d
22:     If gCk(gP(r)) is connected to r then n = gCk(gP(r))
23:     Else if any sibling of gCk(gP(r)) connected to r
24:       then n = sibling of gCk(gP(r)) connected to r //
            may not be empty.
25:       quit
26: If gP(r) is connected to r then n = gP(r)
27: Else if r has route to gP(r) through its parent P(r) then
28:   n = P(r)
29: Else n = connected Sj(r) having route to gP(r) // may also
      be empty.
```

A caller of this algorithm may send the destination unreachable message back to original source node of the SOAP message.

At any SOAP node *r* the algorithm first finds that if destination is directly connected to *r* or the message is for *r* itself, it returns the destination as next hop for delivery of message. Messages destined to group head node, subordinate nodes and siblings nodes are directly delivered on network if available in routing table as directly connected node. Therefore the algorithm returns empty as next hop (unreachable) if the destination is parent node, any child node or sibling and it is not present in routing table as directly connected node. For other destination nodes it goes for indirect delivery by finding the next available hop along the route. If destination is on sub tree rooted at *r* it will be routed through corresponding child or alternatively (if child in not available) through grand child connected to *r* through a secondary (backup) link. If destination is on sub tree rooted at any of the sibling nodes, message will be routed through corresponding sibling and in case of its unavailability (failure of node or link) alternate route will be taken through parent node of *r* which gets a secondary link with network of sibling node of *r*. Messages which are not destined for sub tree rooted at parent of *P(r)* are first tried to be delivered to directly connected related network and if direct link is not available, messages are routed through *P(r)* or *gP(r)* depending upon availability of link.

V. EXPERIMENTS AND RESULTS

For evaluation of proposed routing scheme we have implemented the described algorithms in JAVA and evaluation has been carried out on a simulated network. We developed a simulator in JAVA by implementing the corresponding data structures for network nodes, networks, and routers with routing tables. Network nodes were configured to be connected to network(s) according to required network layout represented by Fig. 4. Nodes were assigned with names based upon hierarchical naming scheme whereas individual networks were configured to have a particular set of nodes representing a group network of subordinates. On simulator we setup a network of interconnected group networks each having same number of subordinate nodes. Network nodes could send updates to networks, receive updates from network and populate respective internally stored routing tables. We could monitor routing updates and table sizes of nodes during our experiments for evaluation of routing scheme. The total number of nodes was varied either by changing the number of nodes per group or changing the number of hierarchical levels, to observe routing overhead. We have evaluated the routing scheme overhead with all the network links in place so that we could get the maximum sizes of updates and routing tables at network nodes. Routing table size was measured as the total number of node names stored by all routing records in the routing table.

During experiments, we retrieved the maximum routing table size among all the participating nodes and observed its variation with change in number of nodes. We observed that for a given number of nodes per individual network, maximum routing table size did not increase with increase in number of nodes after five levels of network hierarchy, as shown in plot of Fig. 5. This is because the proactive part of our routing scheme is limited to two levels on each side, both up and down, of a given hierarchical level of a node, i.e. to grand parent and grand children levels respectively.

We observed variation of maximum routing table size with increase in number of nodes by increasing number of nodes per individual network for a fixed number of levels of network hierarchy. We observed an increase in maximum table size with increase in number of nodes per individual network as shown in plot of Fig. 6.

For comparison with the routing scheme as used in earlier work [7], we have also plotted the behaviour of distance vector like proactive routing scheme in which all the nodes will store routes to all other overlay nodes on networks as all the nodes and all nodes might need to host services. The comparison of plots in Fig. 5 and Fig. 6 shows the lower routing table overhead of our routing scheme.

We have also found percentages of participating nodes having different routing table sizes. A plot of percentage of nodes against available sizes of routing tables on network is shown in Fig. 7. This profile for hierarchical network layout shows that most of the nodes on network, mainly from lowest level of tree having no subordinates at all, keep smaller routing tables for proactive part of routing. A significant percentage of nodes on network, mainly from second lowest level having no secondary links to lower networks have average sized routing tables as evident from small peak at average table size in Fig. 7.

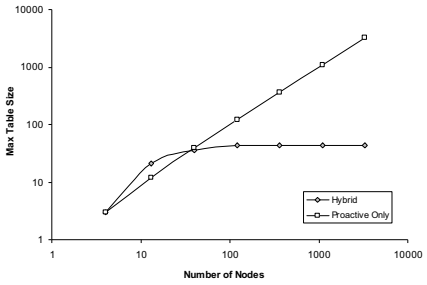

Figure 5. Variation of maximum routing table size with increase in number of nodes by changing hierarchy levels. (logrithmic scale)

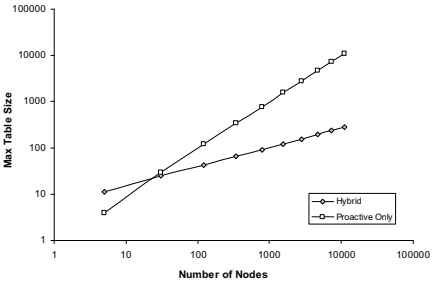

Figure 6. Variation of maximum routing table size with increase in total number of nodes by changing group size. (logrithmic scale)

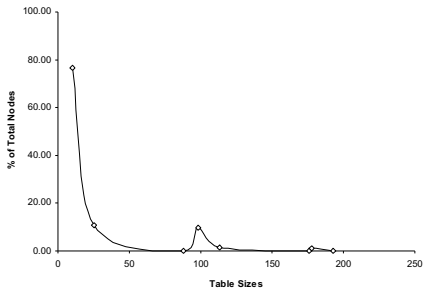

Figure 7. Percentage of nodes having different routing table sizes.

Variation of maximum routing table size with increase in number of nodes shows that our routing scheme has a lower overhead in terms of routing table sizes and thus routing updates overhead on network traffic. As routing tables are populated only through updates received from connected networks, we may take sizes of routing tables as an impact of routing overhead on network traffic as well.

Also our routing scheme, unlike other proactive schemes of having routes to all the overlay nodes, will require only a few updates. This is because it needs proactively gathered link states information for only adjacent or directly connected networks whereas to find general or courser direction towards farther destinations it utilizes hierarchical scheme based upon our hierarchical addressing of nodes.

VI. CONCLUSION

We have presented an efficient overlay network routing scheme required for store-and-forward capability of SOAP proxy nodes working at application level. SOAP proxies may use this routing to find the next hop towards a given destination node while providing reliable access of Web services across multiple interconnected constrained networks. The routing scheme has been developed for minimized routing overheads of routing tables and network traffic for unreliable networks having packet losses, delays, disconnections, and lower bandwidth, by considering operational networks of military, law enforcement and emergency response deployments.

Performance evaluation results demonstrate a lower routing overhead of the routing scheme as nodes need to get information only about adjacent networks. Nodes do not need to store routes to all other participating nodes or networks as routes to farther destinations are derived from the logical hierarchical node names of destinations. Based on the hierarchical structure of network deployments, use of logical hierarchical node names also makes the scheme independent of network layer addresses, which is a useful feature for easy management of addresses across multiple networks. A profile of routing table sizes across participating network nodes highlights the use of smaller sized routing tables used to store information about adjacent networks. Comparison of simulation results with routing in earlier work also proves the scheme suitable for use in constrained networks.

ACKNOWLEDGMENT

This work was supported by China 863 program (No.2012AA011203, No.2011AA01A202), National Natural Science Foundation of China (No.61103031, No.61370057), A Foundation for the Author of National Excellent Doctoral Dissertation of PR China, Beijing Nova Program, Specialized Research Fund for the Doctoral Program of Higher Education (No.20111102120016), the State Key Lab for Software Development Environment (No.SKLSDE-2012ZX-12).

Moreover, we are thankful to fellow researchers of Service Computing Lab in School of Computer Science and Engineering, Beihang University who supported us during this work. We would like to appreciate the moral support from our families for our contribution and achievement in research.

REFERENCES

[1] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "Service-Oriented Computing: State of the art and research challenges," *Computer*, vol. 40, no. 11, Nov, 2007.

[2] N. Suri, "Dynamic Service-Oriented Architectures for tactical edge networks," in *ACM Proceedings of the 4th Workshop on Emerging Web Services Technology*, 2009.

[3] P. Lalbakhsh, N. Sohrabi, and M. N. Fesharaki, "The role of Service Oriented Architecture in battlefield situational awareness," in *2nd IEEE Int. Conf. on Comput. Sci. and Inform. Tech*, 2009.

[4] Y. Bassil, "Service-Oriented Architecture for weaponry and battle Command and Control systems in warfighting," *International Journal of Information and Communication Technology Research*, vol. 2, no.2, Feb, 2012.

[5] D. Russell, N. Looker, L. Liu, and J. Xu, "Service-Oriented integration of systems for military capability," in *11th IEEE symposium on Object Oriented Real-Time Distributed Computing*, 2008.

[6] K. Lund, E. Skjervold, F. T. Johnsen, T. Hafsoe, and A. Eggen, "Robust web services in heterogeneous military networks," *IEEE Computer Magazine*, vol. 48, no.10, Oct, 2010.

[7] E. Skjervold, T. Hafsoe, F. T. Johnsen, and K. Lund, "Delay and Disruption tolerant web services for heterogeneous networks," in *Military Communications Conference*, IEEE, 2009.

[8] S. Simanta, D. Plakosh, and E. Morris, "Web services for handheld tactical systems," in *IEEE Systems Conference*, 2011.

[9] G. Gehlen, F. Aijaz, and B. Walke, "Mobile web service communication over UDP," in *IEEE 64th Vehicular Technology Conference*, 2006.

[10] I. Chlamtac, M. Conti, and J. J. N. Liu, "Mobile ad hoc networking: imperatives and challenges," *Ad Hoc Networks*, vol. 1, no. 1, Jul, 2003

[11] D. R. Wilcox and M. G. Ceruti, "A structured service-centric approach for the integration of Command and Control components," in *IEEE International Conference on Services Computing*, 2008.

[12] G. Pei, M. Gerla, X. Hong, and C. C. Chiang, "A wireless hierarchical routing protocol with group mobility," in *Wireless Communications and Networking Conference*, IEEE, 1999.