Mass Configuration with Confirmation in Tactical Networks

Prateek K. Singh*, **Koushik Kar***, **James H. Nguyen**† and **Daniel Ku**†

*Rensselaer Polytechnic Institute, Troy, NY 12180, USA

†US Army CERDEC, Aberdeen Proving Ground, MD 21005, USA

Abstract—**In this paper we present the design and evaluation of efficient and reliable confirmation-based mass configuration protocols (cMCONF) for wireless Mobile Ad-hoc Networks (MANETs). Confirmation based MCONF - proposed as MCONF-Mode B in our earlier work - enables mass configuration changes in a MANET while obtaining feedback information from nodes using a classic flooding approach. In this paper, we propose and evaluate three versions of cMCONF that collect feedback information from the network (confirmation of successful configuration change) more efficiently than classic flooding: (a) multi-parent Destination Oriented Directed Acyclic Graph (DODAG) based cMCONF (m-DODAG-cMCONF), (b) single-parent DODAG based cMCONF (s-DODAG-cMCONF) and (c) Unicast based cMCONF. The proposed protocols improve the efficiency and reliability performance of the confirmation process of MCONF-Mode B operation. We evaluate the performance of the proposed protocols in terms of reliability and message redundancy by emulating MANETs on Common Open Research Emulator (CORE). From these emulations, we observe that m-DODAG-cMCONF provides an excellent tradeoff between reliability and message complexity across different network sizes and mobility scenarios.**

I. Introduction

Reliable implementation of configuration changes at all nodes in a MANET is often necessary to adapt and operate the MANET safely and efficiently in a changing network environment. Configuration change examples in MANET communication includes changing the communication frequency (channel), retransmission limits, timeouts, security parameters etc. It is desirable that the configuration changes be executed reliably by all nodes in the network, along with additional capacity to track execution. For critical changes, there is potential for node isolation if the configuration change process is performed incorrectly or is not executed in all nodes of the network. The configuration protocol must be able to 1) reliably and efficiently execute the configuration change throughout the MANET, 2) quickly identify nodes that have not been able to execute the configuration change (possibly due to network disconnectedness, node isolation etc). A critical configuration change benefits from the ability to gather feedback on whether the configuration change was successfully executed by all nodes, so that rollback or other corrective steps can be taken if it is not the case. Configuration changes that can permanently isolate nodes such as a changing the frequency or authentication key, can benefit from these additional verification (confirmation) step.

Confirmation based Mass Configuration (MCONF) protocol, also known as MCONF-Mode B, developed in [1] provides means to successfully perform a mass configuration change in MANET for different operating situations and provide confirmation of configuration execution. MCONF-Mode B protocol uses classic flooding approach for configuration rollout as well as confirmation gathering which results in inefficiency and delay in executing the overall configuration change process. A MCONF protocol must result in a high execution reliability for mobile network topologies using low number of messages and must be able to track configuration execution.

Most of the research work reported on configuration changes in networks has been focused on developing messaging conventions and high level languages, such as NETCONF protocol [2], and related performance enhancements [3]. Since the question of mass configuration is related to reliable broadcasting in MANETs, we briefly review that literature here. Banerjee *et al.* [4] present reliable broadcasting solution in MANETs through the creation of tree like construction to avoid duplication of flooding, while Liu *et al.* [5] and Mohsin *et al's* [6] solution depend on immediate neighbor information instead of constructing a tree. Other methods depend on the knowledge of node location [7], utilizes a clustered structure of the network [8], [9] or uses a hybrid approach [10]. As mentioned earlier, [1] proposes a mass configuration protocol MCONF which contains three different operation modes to meet conditional and unconditional configuration change demand in a MANET, but is based on an inefficient flooding based approach. In our previous work [11], we extend this work and propose two high performance efficient/reliable MCONF protocols, E-MCONF and S-MCONF, for the Mode A operation that is meant for non-critical changes, and does not require confirmation on successful configuration changes.

The specific novel contributions of this work are as follows. We propose, implement and evaluate three different confirmation based mass configuration protocol known as cMCONF protocols, specifically focused on reducing the redundancy of the confirmation step in MCONF. cMCONF is designed to provide efficient mass configuration roll-out while maintaining high reliability in the MANET. The first cMCONF protocol, named as multi-parent Destination Oriented Directed Acyclic Graph (DODAG) based Mass Configuration protocol (m-DODAG-cMCONF), creates a multi-parent DODAG for the forwarding of confirmation messages. The second cMCONF variant, s-DODAG-cMCONF, is a single-parent version of m-DODAG-cMCONF that attempts to further reduce the redundancy of feedback messages. The third protocol, Unicast-cMCONF employs a simple unicast transmission from the MANET nodes to the root during confirmation message transmission, by utilizing an external routing protocol (OLSR in our case) for its implementation.

Our evaluation is achieved through implementation on a realistic emulation platform that takes into account the full complexity of the MANETs, the routing and MAC layers in measuring performance. We utilize Naval Research Laboratory's OLSR reference prototype NRLOLSR [12], specifically for Unicast-cMCONF protocol implementation. We performed extensive emulations using CORE [13], [14] and BonnMotion [15] to establish the performance of our model; CORE allows real-time network emulation for static and dynamic network scenarios with a high-level emulation platform and graphical interface.

The rest of this paper is organized as follows. Background material on confirmation based classic flooding MCONF, and the DODAG concepts from the RPL protocol, are discussed in Section II. The proposed cMCONF protocols are presented in Section III. We present and analyze the emulation results in Section IV, and finally conclude in Section V.

II. Confirmation based Classic Flooding MCONF Model and RPL-DODAG

In this section we first discuss the original MCONF protocol, outline its two modes of operation, and discuss its limitations. We also provide a brief overview of DODAG based messaging ideas from the RPL protocol [16], some elements of which have been employed in the first two protocols of this paper.

A. Original MCONF Protocol

To effectively propagate mass configuration changes initiated by the root node of the MANET, possibly carrying out the commands of a remotely located Network Operation Center (NOC), MCONF protocol provides two operational modes. The first mode (Mode A) disseminates configuration change without any confirmation feedback. The second mode (Mode B), contrary to Mode A, requires confirmation from the network nodes after configuration roll-out. Our proposed cMCONF protocols are based on Mode B operation and are described in Section III. The MCONF protocol operation is divided into multiple states; **Ready**, **Wait_Commit**, **Execute** and **Wait_Done**. Mode A uses the first three states while Mode B utilizes all four states. In Mode A, each node is initially in a default **Ready** state and waits to receive configuration change (COMMIT) messages. The root node initiates the operation by sending a COMMIT message to its first hop neighbors which is forwarded further downstream in the network through a limited flooding based approach, as follows. Each node receiving the COMMIT message enters the **Wait_Commit**, in which it rebroadcasts the message (after regular intervals of time) until it has heard the COMMIT message being transmitted by each of its neighbors. In other words, while there are no explicit acknowledgments, hearing a COMMIT message transmission by a neighbor serves as an implicit acknowledgment that the neighbor has received the message. In Mode A, it only uses one type of message - the COMMIT message. Note that the **Execute** step can be either before or after the **Wait_Commit** phase, depending on whether the protocol implements early or lazy execution. Due to its reliance on flooding, MCONF does not rely on any routing protocol, nor does it need to maintain or construct a topology for message propagation.

Mode B supplements Mode A with an additional step (confirmation process). In Mode B, after the completion of the **Wait_Commit** and **Execute** phases, each node changes its state to **Wait_Done**, and sends (broadcasts) DONE messages that acts as confirmation of execution of configuration change by itself, as well as all other nodes that it knows of (based on the DONE messages it has heard). Thus, each DONE message include a list of nodes that it (the sender of the message) knows as having executed the change, and neighboring nodes hearing that broadcast update their database of which nodes have executed the change, which they include in their DONE messages. A node keeps broadcasting these DONE messages until it has no new information (confirmation) to report within a timeout period. At the end of this flooding based confirmation process, all nodes in the network (including the root) would know which nodes successfully executed the configuration change.

The main limitation of these two operation modes is that they utilizes classic flooding for their operations and is unsuitable for large MANETs due to high message redundancy. This results in lower throughput and wastage of network resources. In our previous work which is based on Mode A operation, we have attempted to reduce the inherent redundancy of the classical flooding technique while maintaining mass configuration performance [11]. However as discussed earlier, execution of critical changes must be associated with a feedback mechanism for tracking of configuration change in the network. Note that the information in the DONE messages sent out by different nodes can initially all be different, and gradually converges over time due to aggregation. This convergence time (or time for propagation of information throughout the network, including the root) can be quite significant, implying a high message complexity of the confirmation process. The overhead of this flooding based confirmation process can be considerably more than that of the change commit process, particularly in large MANETs. The biggest challenge is to enable this feedback (confirmation) mechanism without flooding the network with configuration change messages, and avoiding feedback implosion. Therefore, in this work we design and implement two feedback based cMCONF protocols, which employ multi-parent and single-parent destination oriented directed acyclic graph (DODAG) structures in the confirmation process. We also evaluate another cMCONF protocol called Unicast based cMCONF, and compare the performance of these three proposed cMCONF protocols with classic flooding based MCONF-Mode B in Section IV.

B. *RPL and Destination Oriented Directed Acyclic Graph*

Routing Protocol for Low Power and Lossy Networks (RPL) is designed by the IETF as a routing protocol for use in constrained networks [16]. It specifies how to build a Destination Oriented Directed Acyclic Graph (DODAG) using an objective function and a set of metrics/constraints. RPL incorporates a DODAG versioning system which ensures that the topology avoid loops and does not become stale.

DODAGs, in general, are loop free topologies which arrange nodes into a directed acyclic graph with a single root node. RPL defines objective functions to optimize the topology according to predefined goals such as link quality, energy usage or hop-count. An execution of RPL with a specific objective function is termed as instance of RPL and multiple instances of RPL can be run within a network, each with its own DODAGs. A node computes its rank in the DODAG, a metric which determines its level in graph structure, from the objective code point specified in a received DIO (DODAG Information Object) message. The neighbor which provides the best rank is chosen as the parent when multiple DIO messages from several neighbors are received. This mechanism generates upwards routes towards the sink (root node) and messages are only forwarded upstream (to parent nodes) to avoid loops. The ranks, and therefore the DODAG itself, is adapted dynamically based on the DIO message broadcasts.

III. System Model and Protocol Description

A. *Multi-Parent DODAG based cMCONF (m-DODAG-cMCONF)*

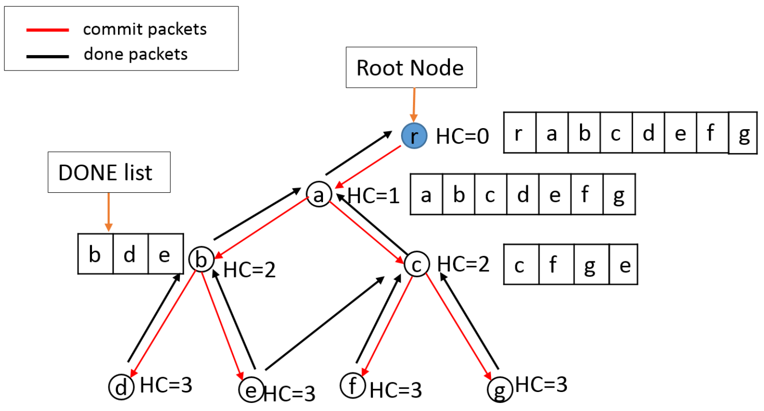

Fig. 1. m-DODAG-cMCONF: Red arrows indicate forward COMMIT transmission with Hop Count (HC) updation, followed by backward DONE message confirmation process towards root node (black arrows). Node ‘e’ has multiple parents.

In this section we present the architectural design of DODAG based cMCONF that admits multiple parents per node. We describe the mechanism for the hop-count metric, although it can be generalized to other link quality metrics as well. Each node maintains an attribute called ‘Hop Count (HC)’ which is similar to the rank metric of RPL-DODAG, and indicates the minimum number of hops to reach root node.

The HC for each node is initialized to zero. The root node initiates the protocol by broadcasting to its neighbors a COMMIT message appended with its hop count information. This message is further relayed downstream, but after updating the hop-count information. During this forward (downstream) messaging, each node looks at the HC information present in COMMIT message; then it chooses minimum value from all the received HC values and increments it by one to determine its own HC value. It considers the neighbor node(s) sending the minimum HC value as its parent node(s). (Note that the parents of node with hop-count $h + 1$ consists of all its neighbors that is at hop-count h; thus a node can have multiple parents.) Further note that no message needs to be sent by a node to the parent node(s) establishing this relationship. As we will see shortly, this relationship can simply be implied by the hop-count values tagged along with all messages that are sent/received.

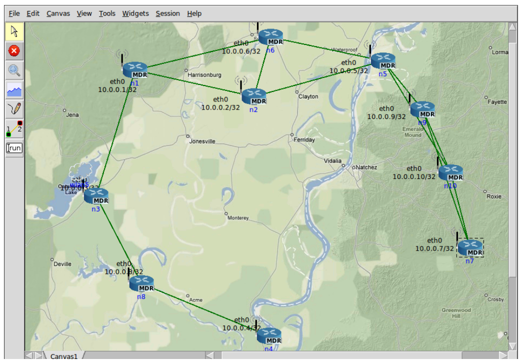

Fig. 2. A snapshot of 10 node mobile random walk topology emulation in CORE.

The forwarding of the configuration change (COMMIT) messages from the root to downstream nodes is very similar to the original MCONF, except that it includes the additional HC information. This step can be viewed as a reliable broadcast of the COMMIT message sent from the root to all other nodes in the MANET. Note that no ACK message is needed for reliability; the COMMIT message from the neighbor (other than carrying information about the neighbor’s hop-count) also serves as an implicit acknowledgment that the neighbor has received the COMMIT message. As in the original MCONF, this greatly simplifies and reduces the amount of message exchanges in the network during the forwarding of the COMMIT message (when the nodes are in the **Wait_Commit** state). Note that a node keeps tracks of all HC values of all its neighbors, as indicated in these COMMIT messages. As in the original MCONF, the **Execute** step at a node is before or after the **Wait_Commit** phase (but must be after the first COMMIT message is received by the node, of course), depending on whether the execution is *early* or *lazy*.

After the completion of the **Wait_Commit** (early execution) or **Execute** (lazy execution) phase, a node moves to the **Wait_Done** state, in which a node periodically resends broadcasts DONE messages, letting its neighbors know that it

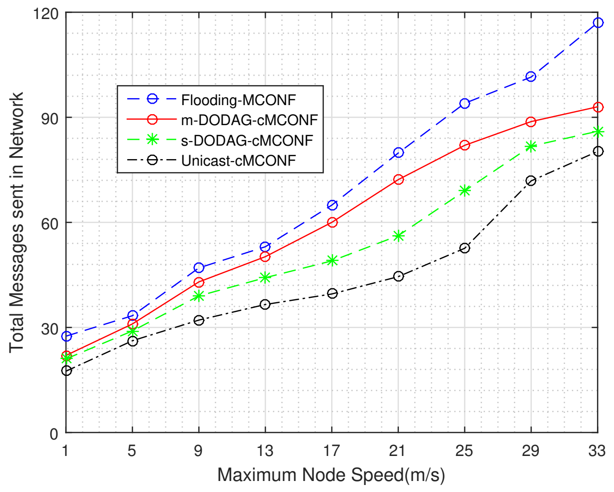

Fig. 3. Redundancy Performance vs Mobility for 10 node network

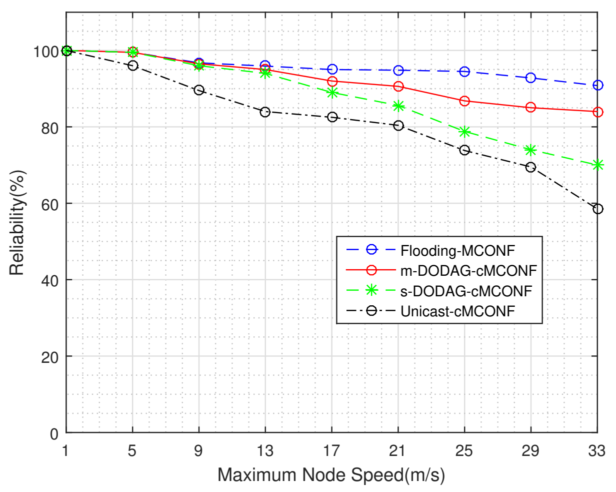

Fig. 4. Reliability Performance vs Mobility for 10 node network.

is done executing the change. However, unlike MCONF-Mode B, in m-DODAG-cMCONF a node rebroadcasts a DONE message only if it was sent by a child node. Since the parent-child relationship was not formally established (through message exchange), if a node has a hop-count of h, any node with hop-count $h + 1$ is assumed to be a child. (For convenience, the hop-count information could be included in the DONE message as well, although it is not necessary if each node maintains a list of the hop-counts of its neighbors from the **Wait_Commit** step, as described earlier.) Note that a node can aggregate the information from the DONE message from several children, and send its DONE messages periodically with the latest aggregated information. Also note that there is no formal acknowledgment of the DONE message. If a node hears the information in its latest DONE message included in a parent's DONE message, it assumes that the parent must have received its DONE message; it will rebroadcast the DONE message after regular intervals (up to retransmission limit) until that happens. Thus, in the **Wait_Done** phase the information on which nodes have executed propagates efficiently back to the root node, in an aggregated manner through the destination (root) oriented DAG structure. This structure is *implicitly* constructed/maintained through the HC information exchange during the **Wait_Commit** phase.

B. Single Parent DODAG based cMCONF (s-DODAG-cMCONF)

The Single Parent DODAG based cMCONF's implementation is similar to the m-DODAG-cMCONF, with one important difference. For a node with hop-count $h + 1$, if there are multiple neighboring nodes with hop-count h, only one of them is chosen as the parent, and only that node will be rebroadcasting the information in the child node's **Wait_Done** message. Therefore, the child node would have to choose one of those neighbors (with hop-count h) as a parent node, and notify it. This notification can be done without any additional message - simply by indicating the parent node's id (like the routing address) in the DONE message. In that way, only a node that sees its own id (as a parent node) in the DONE message will aggregate and rebroadcast that information. Note that child node can potentially choose another parent node (at the same hop-count h) when retransmitting the **Wait_Done** message - for load balancing, for example. However, the key difference here is that unlike m-DODAG-cMCONF, a single message in s-DODAG-cMCONF propagates to the root in only one path.

C. Unicast based cMCONF

In this protocol, each node simply unicasts the DONE message to the root node during the confirmation process. Unlike the DODAG based approaches, such unicast at the transport layer does not implement hop-by-hop reliability. While its reliability can be low in high mobility scenarios (as we will see in the next section), it avoids any redundant transmission of messages. Moreover, it relies on the existence of a unicast routing protocol (such as AODV or OLSR) in the MANET. In this work, we primarily use this Unicast-cMCONF to performance benchmark the two DODAG-cMCONF approaches.

IV. Emulation Results

With Common Open Research Emulator (CORE), we emulate different mobile scenarios to test varying mobility conditions using pre-generated movement scripts. CORE provides a high-level emulation platform with a graphical interface and provides the ability to connect to live networks. Figure 2 shows a random walk scenario emulation using CORE with 10 nodes including 9 mobile nodes and 1 stationary node serving as the gateway or root. We utilized BonnMotion to create Random Walk mobility condition to simulate movement of nodes across a rectangular terrain. To represent general ground vehicular speed range, mobility of the nodes is varied from

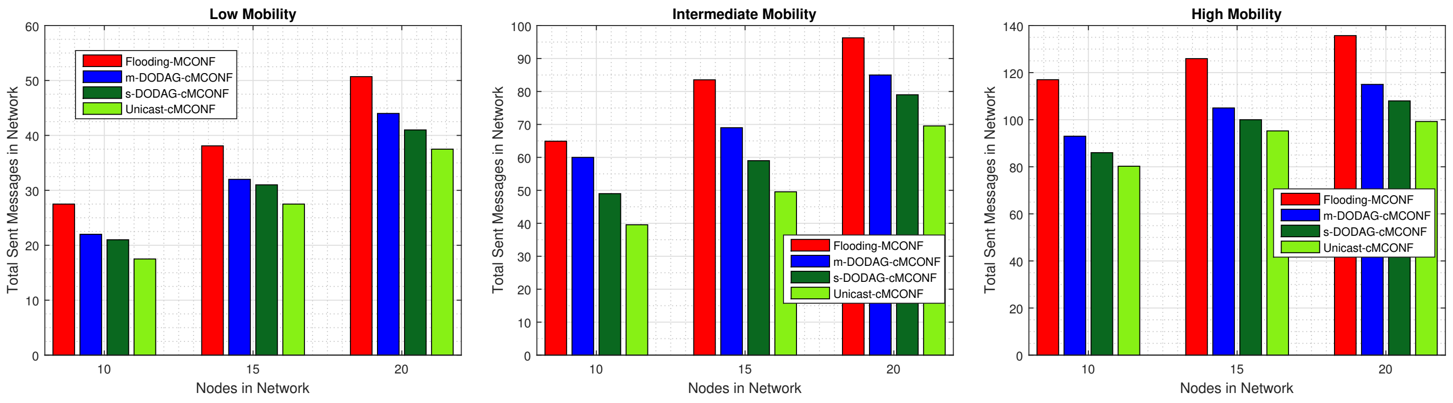

Fig. 5. Redundancy Performance using Low, Intermediate and High Mobility for 10, 15 and 20 node networks of the proposed protocols.

1 m/s (2.24 miles/hour) to 33 m/s (73.8 miles/hour). In this implementation, we utilize lazy execution operation for mass configuration. We evaluate the performance of our proposed protocols through two different metrics: (a) reliability i.e. the percentage of nodes being executed and (b) redundancy i.e. the average number of total sent messages (COMMIT, DONE etc.) from the nodes in the network.

General parameters and their values used in different cM-CONF protocol evaluation are presented in Table I. We emulate a 10, 15 and 20 node mobile CORE scenarios in this evaluation. For all mobility cases, the gateway (or root) node initiates the protocol. Movement of nodes results in the reduction of average nodal connectivity of the network, which creates disconnection in topology for a single or groups of nodes, with higher mobility resulting in further disconnections.

TABLE I
cMCONF Protocol's General Parameters & Settings

Parameters	**Description**	**Value**
m	Message size	1KB
r_{done}	Retransmissions count	20
t_{exec}	Execution timeout	15 s
t_{dodag}	DONE timeout- m-DODAG, s-DODAG	200 ms
t_{uni}	DONE timeout - Unicast	200 ms

A. *Performance Analysis*

In the first set of results presented here, we emulate a 10-node mobile scenario as shown in Figure 2. In Figures 3 and 4, we analyze the redundancy and reliability attained through our proposed protocols and compare them with those attained by the classic flooding based MCONF. Note that for any given protocol, the redundancy increases with increase in mobility. We observe that all three proposed protocols outperform the classic flooding MCONF as far as message efficiency is concerned. The reduction in message redundancy is even lower comparably at high node mobility rates for our proposed protocols. The effect of this reduction in redundancy on the protocol's reliability is shown in Figure 4. We observe that even though the flooding based MCONF achieves a comparatively higher reliability than our proposed protocols, it does so at the cost of consistently higher redundancies. The reliability performance of s-DODAG-cMCONF and Unicast-cMCONF protocols is slightly worse than flooding based MCONF. However, m-DODAG-cMCONF protocol maintains good reliability and redundancy performance, especially at high mobility where it is able to reduce redundancy by 27% with a slight decrease in reliability. Moreover, all three proposed protocols outperform flooding based MCONF at low mobility. The network suffers more disruptions with increase in mobility which reduces the average number of executed nodes. This results in more redundancy (larger number of messages) at high mobility as shown in Figures 3 and 5.

In the next set of experiments, we increase the network size to 15 and 20 nodes respectively and compare the redundancy and reliability performance with classic flooding based MCONF, at different mobility levels. Figures 5 and 6 shows redundancy and reliability performance for 10, 15 and 20 node scenario collectively at low (1 m/s or 2.2 miles/hr), intermediate (17 m/s or 38.02 miles/hr) and high mobility (33 m/s or 73.8 miles/hr). Note that the increase in nodal density in a network results in larger number of messages, as expected. Similar to the 10 node scenario, all three proposed protocols outperform the flooding based MCONF at low mobility. At high mobility, m-DODAG-cMCONF protocol is a standout among all the proposed protocols from a redundancy-reliability tradeoff perspective.

The option of having multiple parent nodes broadcasting a child node's messages enhances m-DODAG-cMCONF protocol's performance when compared with the flooding based MCONF. Its lower message redundancy does not affect reliability performance much at all mobility rate as is evident from Figure 6. Note that in Unicast-cMCONF protocol, the probability of root node receiving confirmation messages through unicast from all other nodes decreases considerably at high mobility, which lowers the configuration execution rate (reliability performance) of the protocol. Low mobility ensures higher average nodal connectivity and hence relatively higher

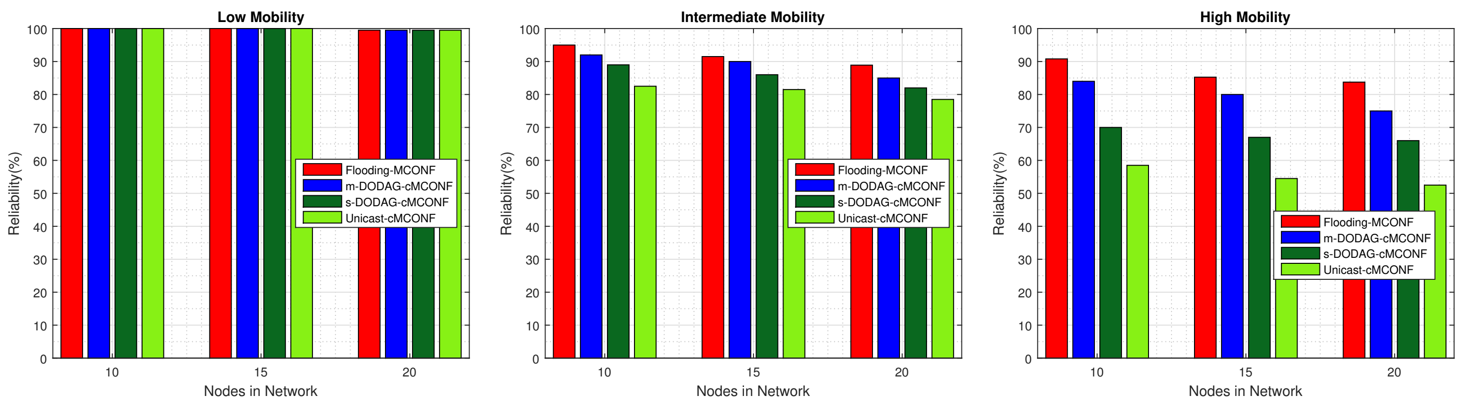

Fig. 6. Reliability Performance using Low, Intermediate and High Mobility for 10, 15 and 20 node networks of the proposed protocols.

configuration execution. Therefore, Unicast-cMCONF and s-DODAG-cMCONF are best suited for low mobility scenarios. However, it should be noted that unicast-cMCONF can only be used when an existing unicast routing protocol (like OLSR) is already operational in the MANET. The DODAG based cMCONF protocols do not have this limitation. Furthermore, m-DODAG-cMCONF protocol performs well at both low and high mobility scenarios, and therefore can be considered for use over a wide range of mobility.

V. Conclusion

In this paper we propose and evaluate three variants of a confirmation based mass configuration protocol (cMCONF) for implementing efficient and reliable configuration changes in MANETs. The first cMCONF variant, m-DODAG-cMCONF, focuses on reducing message redundancy in the network by employing DODAG during confirmation process and uses multiple parents to forward a node's message for better reliability with a slight increase in message redundancy. s-DODAG-cMCONF applies similar implementation but only allows single parent of a node. The third cMCONF variant, Unicast-cMCONF, focuses on efficiency by enabling unicast communication between child node and root node by utilizing an existing unicast routing protocol. Performance of all three cMCONF protocols are evaluated against the Classic Flooding based MCONF-Mode B for different mobility scenarios and three different MANETs with increasing nodal density. The performance of all three protocols are quite good in the low to intermediate mobility range. At high mobility, the performance of s-DODAG-cMCONF and Unicast-cMCONF degrades significantly in terms of reliability. However, the m-DODAG-cMCONF can still be used in high mobility conditions with high reliability and significantly decreased redundancy as compared to a flooding approach. Our results show that m-DODAG-cMCONF is efficient, versatile and well suited for a wide range of applications.

References

[1] P. Katlic, K. Kar, J. Morris, J. H. Nguyen, and R. G. Cole, "Design and evaluation of a mass configuration protocol (MCONF) for tactical manets," in *Military Communications Conference, MILCOM 2015-2015 IEEE*. IEEE, 2015, pp. 1361–1366.

[2] J. Yu and I. Al Ajarmeh, "An empirical study of the NETCONF protocol," in *Networking and Services (ICNS), 2010 Sixth International Conference on*. IEEE, 2010, pp. 253–258.

[3] P. S. Giri Kuthethoor, J. Strohm, J.-y. Lu, C. Hansupichon, G. Hadynski, D. Climek, J. DelMedico, N. Rome, D. Kiwior, D. Dunbrack *et al.*, "Performance improvements to NETCONF for airborne tactical networks."

[4] S. Banerjee, A. Misra, J. Yeo, and A. Agrawala, "Energy-efficient broadcast and multicast trees for reliable wireless communication," in *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, vol. 1. IEEE, 2003, pp. 660–667.

[5] S. Alagar, S. Venkatesan, and J. Cleveland, "Reliable broadcast in mobile wireless networks," in *Military Communications Conference, 1995. MILCOM'95, Conference Record, IEEE*, vol. 1. IEEE, 1995, pp. 236–240.

[6] Y. Liu, F. Y. Li, and H.-P. Schwefel, "Reliable broadcast in error-prone multi-hop wireless networks: Algorithms and evaluation," in *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*. IEEE, 2007, pp. 5329–5334.

[7] M. Mohsin, D. Cavin, Y. Sasson, R. Prakash, and A. Schiper, "Reliable broadcast in wireless mobile ad hoc networks," in *System Sciences, 2006. HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on*, vol. 9. IEEE, 2006, pp. 233a–233a.

[8] E. Pagani and G. P. Rossi, "Reliable broadcast in mobile multihop packet networks," in *3rd annual ACM/IEEE international conference on Mobile computing and networking*. ACM, 1997, pp. 34–42.

[9] W. Wu, J. Cao, and M. Raynal, "Eventual clusterer: A modular approach to designing hierarchical consensus protocols in MANETs," *IEEE transactions on parallel and distributed systems*, vol. 20, no. 6, pp. 753–765, 2009.

[10] W. Lou and J. Wu, "Double-covered broadcast (DCB): A simple reliable broadcast algorithm in MANETs," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 3. IEEE, 2004, pp. 2084–2095.

[11] P. K. Singh, J. H. Nguyen, S. K. Gupta, K. Kar, and D. Ku, "High performance mass configuration protocols for manets using efficient broadcasting," in *Performance Computing and Communications Conference (IPCCC), 2016 IEEE 35th International*. IEEE, 2016, pp. 1–10.

[12] J. Macker and R. Lee, "NRL OLSR routing protocol implementation," 2009.

[13] J. Ahrenholz, "Comparison of CORE network emulation platforms," in *Military Communication Conference 2010*, 2010.

[14] "Common open research emulator CORE." [Online]. Available: http://www.nrl.navy.mil/itd/ncs/products/core,

[15] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn, "Bonnmotion: a mobility scenario generation and analysis tool," in *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, 2010.

[16] T. Winter, "RPL: IPv6 routing protocol for low-power and lossy networks," *RFC 6550*, 2012.