Volume 2, No. 9, September 2011
Journal of Global Research in Computer Science
ISSN-2229-371X
RESEARCH PAPER
Available Online at www.jgrcs.info

IMPROVED BOOTSTRAP SECURITY WITH CONSISTENT CACHE IN MOBILE AD HOC NETWORK ROUTING

P. Parameswari *[1], C. Chandrasekar [2]
[*1] Research scholar, Anna University of Technology, Coimbatore, India
param_pnr2000@yahoo.co.in
[2] Reader, Periyar University, Salem,India
ccsekar@gmail.com

Abstract -Dynamic node mobility of Mobile Ad hoc Network (MANETs) causes security concern over routing and data communication. Recent research works focused on trust worthiness, routing protocol security, denial of service attacks, etc., More focus is made on bootstrap security which for authenticated and secured neighbor relationships between mobile nodes. Our previous work presented bootstrap security scheme capitalizing neighbor node relationship and share knowledge information of other nodes in dynamic MANET. The bootstrap security handles subverted node and the subverted link attacks efficiently. However inconsistence in the cache causes improper information being shared between the neighboring nodes. The improper or faulty information sharing between nodes, leads to various security threats. In this work we present an improved bootstrap security model with efficient cache consistency scheme to share proper and correct information sharing between the mobile nodes of the ad hoc network. The improved bootstrap security minimizes attacks in query nodes of server control cache and eliminates fake requests in query nodes. Cache consistency scheme is a server control mechanism adapts the process of caching a data item and instructs the query node to updates data in the respective mobile cache nodes. NS-2 Simulations are carried out to analyze the performance of improved bootstrap security for query node from passive and active attacks with average data request response time, cache update delay, and bandwidth utilization. The simulation topology is constructed to analyze adversary correctness and anticipation of cache consistency of the cache nodes updated to query nodes. The performance of improved bootstrap security shows an improvement of 15% in terms of control packets information sharing between mobile nodes. The improved bootstrap security shows 35% of improvement in data delivery ratio compared to that of conventional bootstrap security.

Keywords **-** Security, Bootstrapping, Cache Consistency, Ad Hoc Network, Query Directory, Cache Nodes

INTRODUCTION

Deny-by-default computer systems promise enhanced information security (in comparison to their allow-by-default counterparts) by relying on policy rules to explicitly define the various actions that system components are allowed to take. For example, policy in a deny-by-default network would specify which nodes can forward data or control traffic to which other nodes datagrams for which no policy is provided are dropped. In contrast, policy in today's Internet typically specifies only what traffic should be blocked (via firewalls or ingress filters at routers) datagram outside of policy are forwarded. The promise of enhanced security at the expense of open functionality is particularly appropriate in the context of mission-critical applications with the potential of an adversarial presence e.g., military Mobile Ad-hoc Networks (MANETs).

A crucial challenge in a deny-by-default MANET, in the dynamic network topology and information exchange requirements, is the complexity of maintaining nodes up-to-date policy. The previous work addressed this challenge by defining, an axiomatic set of policies from which nodes can obtain additional policies or update outdated policies. It provided bootstrapping by which nodes form, the neighbor relationships among themselves in a manner consistent with current policy. Along with the definition of the Bootstrap as a Finite State Machine (FSM), the proposal in this paper also proved its correctness (safety and liveliness). However, this initial analysis neglected the possibility of nodes or wireless channel being subverted by a knowledgeable adversary. The proposal of this work further handles the knowledgeable adversaries to secure the nodes effectively.

In MANET, data caching is essential as it reduces contention in the network, increases the probability of nodes getting desired data, and improves system performance. The major issue that faces cache management is the maintenance of data consistency between the client cache and the server. In a MANET, all messages sent between the server and the cache are subject to network delays, thus, impeding consistency by download delays that are considerably noticeable and more severe in wireless mobile devices. All cache consistency algorithms are developed with the same goal in mind to increase the probability of serving data items from the cache that is identical to those on the server.

This work describes a server-based scheme implemented on top of the COACS caching architecture we proposed in COACS, elected query directory (QD) nodes cache submitted queries and use them as indexes to data stored in the nodes that initially requested them (CN nodes). Since COACS did not implement a consistency strategy, the proposal described present several improvements i.e., enable the server to aware of cache distribution, restore consistent cached data items, and enhance data update rate as quick as the request made by clients. With these enhancements, improved bootstrap security with cache consistency presented an efficient caching system, augments conventional bootstrap security by minimizing average query response time and there of attack resistance rate.

RELATED WORKS

A mobile ad hoc network (MANET) is an infrastructure less network consisting of a set of mobile nodes that are able to communicate with each other in a multi-hop manner without the support of any base station or access point. A node in a MANET is not only a node but also a router that is responsible of relaying packets for other nodes. Most of the existing protocols have assumed a MANET as a non-hostile, trusted environment. Unfortunately, in the presence of malicious nodes, a MANET is highly vulnerable to attacks due to its openness features, dynamically changing topology, and lack of centralized infrastructure, etc. Thus, the security issues for MANETs are very challenging.

Deny-by-default computer systems [1] promise enhanced information security (in comparison to their allow-by-default counterparts) by relying on policy rules to explicitly define the various actions that system components are allowed to take. For example, policy in a deny-by-default network [2], [8] would specify which nodes can forward data or control traffic to which other nodes, datagrams for which no policy is provided are dropped. In contrast, policy in today's Internet [5] typically specifies only what traffic should be blocked (via firewalls or ingress filters at routers), datagrams outside of policy are forwarded. The promise of enhanced security at the expense of open functionality is particularly appropriate in the context of mission-critical applications with the potential of an adversarial presence e.g., military.

A crucial challenge in a deny-by-default mission critical MANET, in which network topology and information exchange requirements can be time-varying, is the manner in which the nodes maintain up-to-date policy [9]. The previous work [10] addressed this challenge by defining an axiomatic set of policies from which nodes can obtain additional policies or update outdated policies and a protocol (referred to as the Bootstrap protocol) by which nodes form, or bootstrap, the neighbor relationships among themselves in a manner consistent with current policy. Along with the definition of the Bootstrap protocol as a Finite State Machine (FSM) [5], we also proved its correctness (safety and liveliness), however, this initial analysis neglected the possibility of nodes or wireless channel being subverted by a knowledgeable adversary.

Several cache consistency (invalidation) schemes have been proposed in the literature [3], [4] for MANETS. In general, these schemes fall into three types i.e., pull or client model (caching node (CN) asks for updates from server), push or server model, (server sends updates to CN), and cooperative model (CN and server cooperate to keep the data up-to-date). Pull-based strategies achieve smaller query delay times at the cost of higher traffic load, whereas push-based strategies achieve lower traffic load at the cost of larger query delays. Cooperative-based strategies tend to be halfway between both ends.

Server-based approaches employ invalidation reports (IRs)[7] that are periodically broadcasted by the server. An IR normally carries the IDs of the updated data items and the time stamps of the updates. Most research in this area has focused on reducing the time intervals between updates or making the process of sending update reports less static. One of the improvements over the basic IR approach was proposed by Li et al. [12]. The basic idea is for each node to cache the last K reports that are broadcasted by the server, and for the server to store the IRs for future use. Several approaches attempted to make the process of sending reports dynamic [11], [13]. Yuen et al. proposed a scheme by absolute validity interval (AVI) that is calculated dynamically by the server based on the update rate. A serious issue, however, arises with the above approaches when the frequency of updates at the server is high, which could overload the clients with update reports.

The consistency of location-dependent data was studied in [6], which proposes methods to validate and update information that change their values depending on the location of the client, like traffic reports and parking information. In this context, not only the temporal consistency but also the location-dependent consistency (due to client mobility) is taken into consideration. The proposed work in this paper presented a bootstrap security based cache consistency scheme for the security of cache nodes to be carried out to eliminates attacks generated by social neighbor nodes.

SECURE BOOTSTRAP DSR PROTOCOL FOR AD HOC NETWORKS

The Bootstrap DSR protocol is designed for a scenario where two nodes meet in deny by default MANET and both nodes can hear the other's transmissions. The protocol allows the two nodes to establish a neighbor relationship when that relationship, and the process for bootstrapping that relationship, is consistent with policy. Two nodes establishing a neighbor relationship, is that they have each agreed to send/receive messages to/from each other. The Bootstrap protocol relies on a set of axiomatic policies, which is the initial set of primitive policies that are installed a priori and allow a deny-by-default system to establish neighbor relationships. Since axiomatic policies do not define whether a particular neighbor relationship is allowed, for performance reason a node will typically initialized with non-axiomatic OkNeighbor policy that defines a set of nodes with which this node can be neighbor.

First, all messages (exchanged between two nodes) are privacy and integrity protected via encryption and cryptographic hash. Strong identity (e.g., provided by signature) is in place, so no node can impersonate any other node. Second, each message has a timestamp, which serves as a sequence number for defending against message reordering/replaying attack. Third, a node never simultaneously keeps two or more bootstrap sessions with any other node. To rate-limit requests, a node ignores repeated requests within a certain time interval from the first such request.

Subverted Node and Link Threat:

In this threat model, we suppose that exactly one of the two nodes engaged in the Bootstrap protocol is subverted, meaning that the adversary has full control over the behavior of that node. The adversary may always choose to neglect taking its steps in the protocol and, in turn, prevent the neighbor relationship from being established. The adversary's intent is to establish the neighbor relationship.

By our up-to-date policy assumption and the verified correctness of the protocol, we know this neighbor relationship will only be established if permitted by policy.

Weaken the adversary such a way that both nodes in the Bootstrap protocol are non-subverted and instead the adversary is tapped into the channel between them. Neither node is aware of the existence of the adversary, each perceiving the channel like any other unreliable channel. Any message from one node may be processed (i.e., forwarded, delayed or dropped) by the adversary before it may be received by the other. Such a scenario is possible if the adversary uses a directional antenna. Our assumed strong identity services prevent the adversary from altering/spoofing messages. Our analysis proceeds differently depending on different assumptions about the adversary's knowledge of the protocol.

Optimized Bootstrap Security:

The ad hoc protocol provides the capability of using multiple routes between a source-destination pair. It, however, does not provide a mechanism (or framework) to prioritize these routes. When more than one route between a source and destination is discovered, DSR considers the first route known for transmission, but this route may not be the best route considering various factors that characterize an ad hoc wireless network. The prioritization of routes require some state information about the relay nodes, where optimization factors are obtained from random topology changes for mobility and battery level on a node's availability to security issues.

SECURED CACHE NODES FOR CONSISTENT SERVER UPDATE MANET

The nodes in the mobile ad hoc have cooperative cache. Elected query directory (QD) nodes cache submitted queries. Cache nodes (CN nodes) store indexed data initially requested. Cooperative server cache maintains table containing id of a data item (or query) and address of the CN that caches the data. Node needing a data item sends its request to its nearest QD. If QD finds in its cache, it forwards to CN, CN turn sends the item to requesting node (RN). Otherwise, it forwards it to its nearest QD, if the request traverses all QDs without being found, a miss occurs and it gets forwarded to the server, then server sends the data item to the RN.

The cooperative cache adapts a model in which each data object is associated with a single node that can update the source data. This node is referred to as the data source node. Each data object is cached by a collection of nodes called the caching nodes. The data copies held by the caching nodes are called the cache copies. Two basic mechanisms for cache consistency maintenance are push and pull. Using push, the data source node informs the caching nodes of data updates. Using pull, the caching node sends a request to the data source node to check the update. The routing protocol employed in the network layer provides the hop count between each pair of nodes, and the hop count of data transmission is used to measure the consistency maintenance.

Consistent Server Cache updates:

The consistent server cache updates provides data Consistency based on the Pull with TTR. In Pull with TTR, each cache copy is associated with a timeout value Time to Refresh (TTR). The initial value of TTR is set to d. When TTR is valid (TTR > 0), the caching node directly serve cache queries. When TTR expires, the caching node first pulls the data source node to update the cache copy and to renew TTR to d. Then the caching node can directly serve cache queries. By associating a TTR value with each cache copy, guarantees that deviation between the source data and the cache copy will not be over d, thus ensuring data Consistency. Although the Pull with TTR algorithm guarantees data Consistency, it is not cost effective, mainly due to the round–trip consistency maintenance cost imposed by the pull mechanism.

Using push, the data source node informs the caching nodes of data updates, which only imposes one-way consistency maintenance cost (traffic overhead, query latency etc.). However, if the data source node is unaware of the cache status of each cache copy, there exist redundant data update propagations in the following two cases i.e., After the cache copy and the associated TTR are renewed via push, there comes no cache query before the TTR expires. Before the TTR expires, there are multiple data updates (only the last data update should make the data source node push the caching nodes); Thus, the following design principles is followed in designing improved bootstrap security with cache consistency use the push mechanism to save consistency maintenance, if the cache copy is expected to serve queries.

After the data source node has decided the Push Set, i.e., the caching nodes which should receive the data update, it needs to decide how to propagate the data updates among the selected caching nodes. SSUM employs a greedy but efficient strategy to disseminate the PUSH message (which contains the source data update and IDs of the push set nodes) to all caching nodes in the push set. The data source node sends the PUSH message to the nearest caching node in the push set via unicast. The caching node which receives the PUSH message deletes itself from the push set. It acknowledges the PUSH message with a PUSH_ACK message, and then goes on relaying the push message to the nearest caching node in the push set (The path length between two nodes are provided by the routing protocol, according to our assumption). This process is repeated until the push set is empty. The last caching node will send a PUSH message to the data source node, which initiates the PUSH message propagation process.

Attack Vulnerabilities in Query Nodes:

MANET with low-overhead made complex to monitor an environment, some of their attributes make them even more susceptible to security attack or damage. To achieve a better understanding of the risks faced by the network, brief the security attacks against cache nodes of the MANET. Briefing allows us to reason about attacks at a level higher than a simple list of vulnerabilities. It provides a classification system that ideally suggests ways to mitigate attacks by prevention, detection, and recovery. It can aid risk management by identifying vulnerabilities and making attacker characteristics explicit.

The attacker has an identity and a motive, and is able to do certain things in or to the MANET. An attack targets some service or layer, exploiting a vulnerability. An attack may be thwarted, or it may succeed with varying results. Each of these elements is necessary to understand the whole process of a security attack of cache nodes. The two types of security attacks are passive and active. In passive attacks, selfish nodes use the network but do not cooperate, saving battery life for their own communications: they do not intend to directly damage other nodes. In Active attacks, malicious nodes damage other nodes by causing network outage by partitioning while saving battery life is not a priority.

Firewall mechanism to thwart security attack:

In security attacks, the hacker's objective is to render target cache nodes inaccessible by legitimate users. MANET without sufficient protection from security attacks may not be deployable in many areas. Apart from special cases whereby an a priori trust exists in all nodes, the nodes of an ad hoc network cannot be trusted for the correct execution of critical network functions. Essential network operations that assure basic connectivity can be heavily jeopardized by nodes that do not properly execute their share of the network operations like routing, packet forwarding, name-to address mapping, and so on. Node misbehavior that affects these operations range from simple selfishness or lack of collaboration due to the need for power saving to active attacks aiming at security attacks and subversion of traffic are safeguarded by the firewall.

EXPERIMENTAL RESULTS ON IMPROVED BOOTSTRAP SECURITY WITH CACHE CONSISTENCY

Experimental simulations are conducted with NS-2 (V2.34) to evaluate the control packet, message delivery rate and end to end delay of the bootstrapping security mechanism applied in DSR. The simulation used a random way point model, area 1000 * 1000, maximum velocity 20 m/s, wireless range 250 m, nodes 50, and data transfer rates 4packets/s for our simulation topology scenarios. The simulation compares secure bootstrap protocol to DSR and implementation of optimized DSR. In bootstrap DSR, the security overhead of DSR has been improved by reducing route request overhead and shortening packet length which has been shown by using the bootstrap mechanism. In the simulation, it has been shown that the performance of bootstrap DSR provided security with less overhead than optimized DSR.

Simulation of Bootstrapping Security on DSR:

In order to enable direct, fair comparisons between the DSR and the bootstrap DSR, it was critical to challenge the protocols with identical loads and environmental conditions for security provisioning. Each run of the simulator accepts a scenario file as input that describes the exact motion of each node and the exact sequence of packets originated by each node, together with the exact time at which each change occurs in motion or packet origination. Since each protocol was changed in an identical fashion, the performance of these protocols can directly be compared.

Packet Overhead: Bootstrap routing supports DSR since it performs end-to-end signature authentication of control messages and verification of whether a node is authorized to send a control message. Therefore, an intermediate node cannot reply from its cache or send a route maintenance message concerning a link that it is not an end-point of, since it cannot be verified whether that node had the right to send that message; i.e., the node could be malicious. In terms of packet overhead, there is a slight difference between secure bootstrap DSR and optimized DSR, since SDSR does not require any additional message exchanges. In DSR, broadcasting of each Route request packet involves several consecutive control packets for route discovery. So, reduction of Route Request packets in Bootstrap reduces control packet overhead. Figuer 1 shows control packet overhead between bootstrap DSR and optimized-DSR for networks with 10, 25, 50, 75 and 100 nodes.

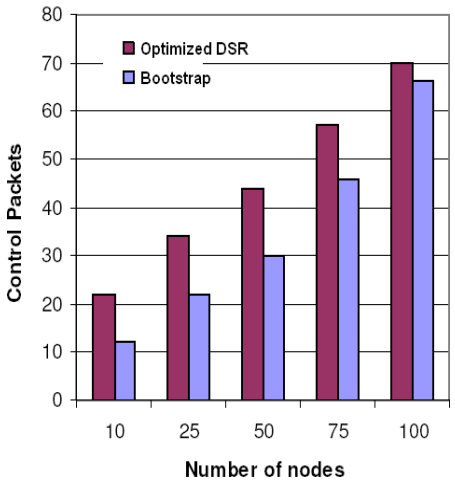

Figure 1: Control Packets for Bootstrap and Optimized DSR

Message over head: Bootstrap routing protocol exchanges the same number of messages as optimized DSR. However, these messages also contain signatures and public-keys. Every control message is signed (16 bytes) and contains the public-key of the signer (128 bytes) since it is not known whether the recipient already has a copy of the public key. Alternatively, if the symmetric key version of Secured bootstrap DSR is used and HMAC-MD5 is the message authentication code, then only an extra 16 byte field is used. Delivery ratio is a measure of efficiency. In bootstrap, since the number of control packet is reduced, number of sent packet is also reduced. So there is less traffic in the network and delivery ratio is high. So it can be said that, bootstrap is more efficient than optimized DSR, depicted in Figure 2.

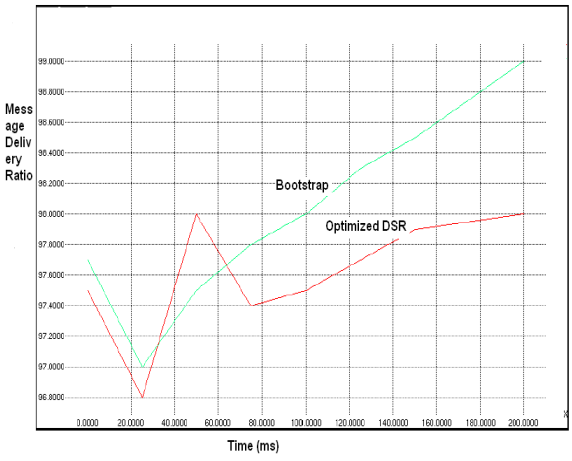

Figure 2: Message Delivery ratio on to Bootstrap DSR and optimized DSR

Average delay: Bootstrap protocol performs most similar to that of DSR which gives a lower bound on the average delay of the security protocol as depicted in Figure 3. Delay metric is measured as the time between packets being sent from the secured application layer at one node and received at the secured application layer of the destination node. This delay takes into account the route discovery and the transmission time. For high mobility, the optimized DSR performs slightly better bootstrap. This is caused by the high number of stale routes provided by the reply from cache in the optimized DSR. However, for low mobility, bootstrap has an average delay double that of optimized DSR.

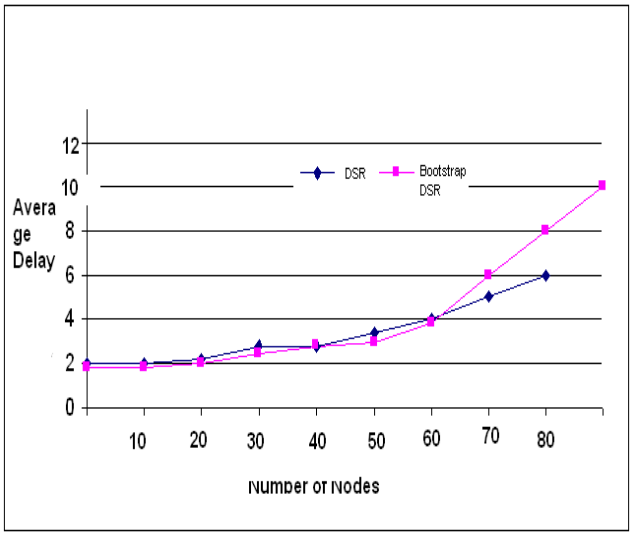

Figure 3: Average delay of Bootstrap DSR and Optimized DSR for node security

Performance of Security on Cache Consistent Scheme:

The security with cache consistency scheme is measured in terms of attack detection rate, false positive and false negative rate, bandwidth and query response time. These are the differences between the corresponding measures when no cache updating is in place and when server update mechanism is employed. Requests for data in the ad hoc network and data updates at the server are assumed to be random processes and may be represented by exponential random variables. The rate of requests processing depends on the fastness of rate of updates. Both the bandwidth metric and response time metric are influenced by the number of data requests issued by requesting nodes relative to the number of data updates that occur at the server (Figure 4).

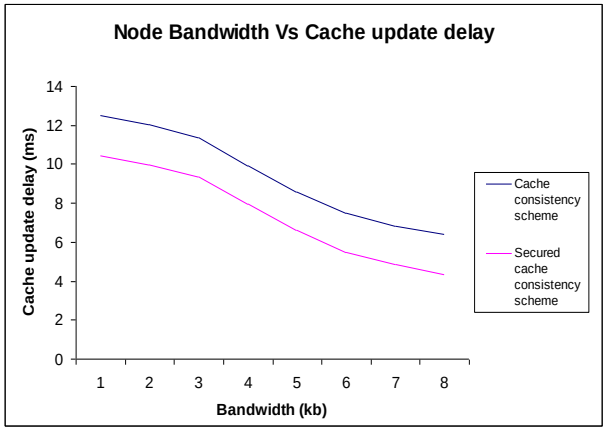

Figure 4: Bandwidth consumption rate of secured cache consistency Scheme

The ratio of attack resistant query request rate to bandwidth utilized to respond to the requests attack resistance expressions indicate that the wireless bandwidth gain may be negative, but the response time gain is always positive. The security resistance performance generates a reasonable trade-off, especially when bandwidth is sacrificed in order to speed up the query response time. It illustrates the average bandwidth and response time gains under two situations, when server cache updates is applied as described and with a modified version that lets the server send updates to the receiving nodes at all times. The performance of secured cache consistency scheme based on the node density is depicted in Figure 5 against cache update delay.

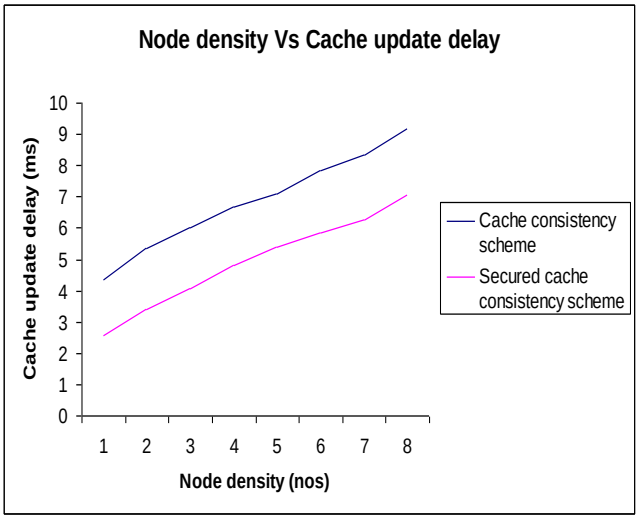

Figure 5: Performance measure of secured cache consistency scheme on node density

Before analyzing the graphs, note that several factors play a role in determining the gain values, most importantly the number of QDs, and the data packet size (Sd) relative to the request packet size (Sr). The effect of the number of QDs is intuitive because with more QDs, more data will be cached in the network, and therefore, both the bandwidth and response time gains will increase. The bandwidth utilization per node increases with the inconsistent cache mechanism, but in secured consistent cache mechanism minimum bandwidth utilization is required which shows better performance of our proposed work.

CONCLUSION

The improved Bootstrap security with consistent cache eliminates the need to assume pre-established secure associations among the nodes of the network. It is highly secure against multiple uncoordinated attackers and simulations demonstrated shows that subverted node attack cannot deadlock a normal node, and subverted link attack cannot undermine the correctness of the protocol. The security mechanism enable query node to maintain cache consistency which in turn minimize inappropriate information attack in Ad hoc Network. The robustness of the architecture and its ability to cope with dynamic environments is high.

The cache consistent mechanism maintains functionality of frequent updates of policy, strong identity, hop-by-hop encryption, and sequence numbers. Node cache consistency prevents link setup with subverted node attacks for detecting and policy updating. Cache consistency is maintained by cooperative cache between nodes for caching data items instructed by server control nodes.

The NS-2 simulation shows that performance of improved bootstrap security in terms of data delivery ratio, delay and control packet overheads are better than conventional bootstrap security model. The performance of the improved bootstrap security is evaluated through gain loss model and compared with Updated Invalidation Report mechanism. The simulation shows that proposed work scale to moderately large network even on frequent node requests. Varying node mobility and link disrupt affect cache update delay to a minimum extent even in dense traffic of ad hoc network.

REFERENCES

[1]. K. Argyraki and D. Cheriton, "Network capabilities: The good, the bad and the ugly," in HotNets-IV, November 2005.

[2]. H. Ballani, Y. Chawathe, S. Ratnasamy, T. Roscoe, and S. Shenker. "Off by default!" in HotNets-IV, November 2005.

[3]. H. Artail, H. Safa, K. Mershad, Z. Abou-Atme, and N. Sulieman, "COACS: A Cooperative and Adaptive Caching System for MANETS," IEEE Trans. Mobile Computing, vol. 7, no. 8, pp. 961- 977, Aug. 2008.

[4]. H. Artail and K. Mershad, "MDPF: Minimum Distance Packet Forwarding for Search Applications in Mobile Ad Hoc Networks," IEEE Trans. Mobile Computing, vol. 8, no. 10, pp. 1412- 1426, Oct. 2009.

[5]. S. Bratus, A. Ferguson, D. McIlroy, and S. Smith, "Pastures: Towards usable security policy engineering," in 2nd International Conference on Availability, Reliability and Security, 2007.

[6]. J. Cao, Y. Zhang, L. Xie, and G. Cao, "Consistency of Cooperative Caching in Mobile Peer-to-Peer Systems over MANETs," Proc. Third Int'l Workshop Mobile Distributed Computing, vol. 6, pp. 573- 579, 2005.

[7]. G. Cao, "A Scalable Low-Latency Cache Invalidation Strategy for Mobile Environments," IEEE Trans. Knowledge and Data Eng., vol. 15, no. 5, pp. 1251- 1265, Sept. 2003.

[8]. T. Wolf, "Design of a network architecture with inherent data path security," in 3rd ACM/IEEE Symposium on Architecture for networking and communications systems, 2007.

[9]. X. Yang, D. Wetherall, and T. Anderson, "A DOS-limiting network architecture," in ACM SIGCOMM, 2005.

[10]. M. Srivatsa, D. Agrawal and S. Balfe, "Bootstrapping Coalition MANETs" in ITA Technical Report, February 2008.

[11]. X. Kai and Y. Lu, "Maintain Cache Consistency in Mobile Database Using Dynamical Periodical Broadcasting Strategy," Proc. Second Int'l Conf. Machine Learning and Cybernetics, pp. 2389- 2393, 2003.

[12]. W. Li, E. Chan, Y. Wang, and D. Chen, "Cache Invalidation Strategies for Mobile Ad Hoc Networks," Proc. Int'l Conf. Parallel Processing, Sept. 2007.

[13]. J. Yuen, E. Chan, K. Lain, and H. Leung, "Cache Invalidation Scheme for Mobile Computing Systems with Real-Time Data," SIGMOD Record, vol. 29, no. 4, pp. 34-39, Dec. 2000.

SHORT BIODATA OF THE AUTHOR'S

P.Parameswari doing Ph.D [Part-Time] research under Anna University of Technology, Coimbatore. Published papers in reputed National and International journals and Participated in various National and International Seminars and Conferences & workshops under the guidance of research supervisor.

Dr.C.Chandrasekar obtained his Ph.D in the year of 2005 . He is in teaching profession for the past 18 years in reputed engineering colleges and universities.. He is guiding more than 30 Ph.D Scholars . He is presently working as Reader at Periyar University Salem. He has presented many papers in National Conferences International Conferences and published many papers in International Journals. His area of interest includes data mining, Image processing, Mobile AdHoc Networks, Network Security etc.