A Sense of Danger: Dendritic Cells Inspired Artificial Immune System for MANET Security

Nauman Mazhar
Deptt of Electrical and Computer Engg
Michigan State University
MI 48823, USA
naumaz@msu.edu

Muddassar Farooq
Next Generation Intelligent Networks
Research Center (nexGIN RC)
NUCES, Islamabad, 44000, Pakistan
muddassar.farooq@nu.edu.pk

ABSTRACT

AIS based intrusion detection systems have traditionally performed self non-self discrimination and suffer from issues such as scalability, false positives, problems with detector generation/holes, need for an initial learning phase, etc. A relatively newer immunological discovery, the Danger Theory, now paves the way for designing more efficient, 2nd generation artificial immune systems. In this paper, we develop a dendritic cell based distributed misbehavior detection system, *BeeAIS-DC*, for a Bio/Nature inspired MANET routing protocol, *BeeAdHoc*. In MANETs, the frequent node movements cause the system self to change, thus increasing the rate of false positives. Our proposed system inspires from the danger theory and models the behavior of the dendritic cells to detect the presence or absence of danger to provide a tolerogenic or immunogenic effect. We have implemented our proposed framework, *BeeAIS-DC*, in network simulator, ns-2, and evaluated its security and network performance. Our results indicate that modelling the dendritic cells allows the *BeeAIS-DC* to dynamically update its detector set to cater for a changing self due to node mobility, and at the same time provides protection against the routing attacks. The network performance evaluation shows that the AIS overhead of *BeeAIS-DC* does not cause significant degradation of its performance, which is vital for a battery/bandwidth constrained mobile node.

Categories and Subject Descriptors

C.2.0 [**General**]: [Security and protection]; C.2.1 [**Network Architecture and Design**]: [Distributed networks, Wireless communication]; C.2.2 [**Network Protocols**]: [Protocol architecture, Routing protocols]

General Terms

Algorithms, Design, Security

Keywords

Artificial Immune Systems, Dendritic Cells, Mobile Ad Hoc Networks, Self-Organization, Misbehavior Detection

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.

1. INTRODUCTION

Artificial Immune Systems (AIS) are inspirations from the Biological Immune System (BIS) [5]. Since the BIS is meant to protect the human body against invasion and damage by pathogens, the most obvious application of its artificial counterpart is to protect computer systems against intrusions by attackers. AIS, over the years, have therefore been studied extensively for network anomaly detection; the authors of [3] provide a comprehensive review.

Mobile adhoc networks (MANETs) is an active area of research. While standardization yet remains, a number of proposals for MANET routing protocols exist in the classical networking domain. These include the *DSR* (Dynamic Source Routing) [9] and *AODV* (Ad-Hoc On-demand Distance Vector Routing) [13], which are well known, reactive routing protocols for MANETs. Research in the Bio/Nature inspired domain of MANET routing has also resulted in the development of state of the art nature inspired protocols, such as *AntHocNet* [4], *BeeAdHoc* [17] and *Termite* [15].

Security in MANETs is an open issue, with MANETs offering a challenging environment to be secured. The wireless medium is inherently insecure; all nodes in close vicinity (within the wireless range) can hear all transmissions and initiate spurious transmissions of their own. MANETs thus provide an ideal environment for a malicious node to fabricate and launch different types of routing attacks. As a result, an attacker can either disrupt the normal routing behavior of a protocol or significantly degrade the performance of the network. A malicious node can join a MANET and easily launch the fabrication, tampering and dropping attacks [12].

A number of security solutions have been proposed for MANET routing protocols, based on standard cryptography and AIS. In classical MANET routing protocols, major security solutions using the cryptographic approach are *ARIADNE* [8], which uses symmetric cryptography to secure the *DSR* protocol, and Secure Ad-Hoc On-demand Distance Vector *(SAODV)* [19], which uses asymmetric cryptography for security of *AODV*. In nature inspired MANET routing protocols, the security vulnerabilities of *BeeAdHoc* have been studied in [12] and a security framework *(BeeSec)* based on digital signature authentication has been proposed. The authors have demonstrated that it successfully counters a number of attacks on *BeeAdHoc*. However, cryptography includes compute intensive mathematical operations and imposes heavy computational and communication load on mobile nodes already constrained in energy and bandwidth.

Artificial Immune Systems offer a relatively novel and promising paradigm to solve the problem of security in MANETs. In classical MANET routing protocols, misbehavior detection of *DSR* using AIS was proposed in [16]. The system is able to detect dropping attacks launched by multiple nodes simultaneously. Similarly, in the nature inspired domain, an AIS based solution, *BeeAIS*

[11], using self non-self discrimination, has been proposed for the *BeeAdHoc* protocol. Launching a number of fabrication and tampering attacks, the authors demonstrated that the *BeeAIS* protocol can counter the routing attacks that were successful against the base protocol, *BeeAdHoc*. Also, the performance comparison showed that the cryptographic security system, *BeeSec*, degraded the *BeeAdHoc* performance to a greater degree than the AIS based solution, *BeeAIS*. Similar results have been reported in [18], when comparing the cryptographic and AIS based security for the *BeeHive* nature inspired routing protocol for fixed networks.
Most of the AIS based intrusion detection systems proposed so far, use self non-self discrimination from classical immunology. Occasional systems add clonal selection to generate memory detectors for improving the secondary response. These systems are based upon negative selection alone and suffer from issues, such as scalability, false positives, problems with detector generation/holes, need for an initial learning phase, etc. In the case of MANETs, there is a serious problem posed by mobility, where the frequent node movements change the system self, causing an increase in the false positive rate. This requires the system to redefine "self" and carry out dynamic updation of detector sets. The relatively newer immunological discovery, the Danger Theory, provides a better explanation of the behavioral aspects of the BIS governing autoimmunity. This theory thus allows building more robust, 2nd generation AISs to cater for problems of the nature that afflict MANETs.
Therefore, in this paper, we propose a Danger Theory based misbehavior detection system for the *BeeAdHoc* routing protocol. Our proposed system, *BeeAIS-DC*, takes inspiration from the danger theory and models the behavior and functions of the dendritic cells (DCs) to provide dynamic updation of detector sets. Having detected the presence or absence of danger signals, the *BeeAIS-DC* algorithm follows the dendritic cell differentiation pathways to incorporate the new/changed self, and detect the non-self, thus providing protection to the mobile wireless network against attacks from malicious nodes. To the best of our knowledge, this is the first attempt to provide DC based security in the nature inspired domain of MANET routing.

Organization of Paper. The rest of the paper is organized as follows. In Section-2 we introduce the Danger Theory, detailing the function and behaviour of the dendritic cells and discuss some relevant applications. In Section-3 we briefly describe the *BeeAIS* security framework already implemented for the *BeeAdHoc* protocol and describe the simulations that we performed under mobility. We point out that the *BeeAIS* does not provide an efficient security solution under mobility conditions. We then introduce our proposed security framework, *BeeAIS-DC* in Section-4. In Section-5, we describe the attacker framework, embedded in ns-2, that launches routing attacks on *BeeAIS-DC* and demonstrates the ability of our proposed protocol to counter these attacks. In order to verify that our security enhancements do not degrade the performance of the original *BeeAdHoc* algorithm, we extensively compare *BeeAIS-DC* with *BeeAdHoc*, *AODV* and *DSR* protocols in Section-6. Our ns-2 simulation results clearly indicate that the network performance of *BeeAIS-DC* is quite close to that of *BeeAdHoc*; in fact, the *BeeAIS-DC* achieves better performance compared to *AODV* and *DSR*. Finally, we conclude the paper with an outlook to our future research.

2. DANGER THEORY

The *self nonself discrimination* is a widely accepted viewpoint in immunology, according to which the activation of adaptive immune system depends upon the recognition of foreign entities in the body. Danger Theory, proposed in [10], challenges this claim and postulates that in addition to pathogen recognition, the adaptive immune response requires detecting the presence of "danger" in tissues, indicating some damage to the body cells due to pathogenic infection. This recognition of "danger" is performed by certain cells of the innate immune system, the Dendritic cells (DCs). In effect, the Danger Theory puts the innate immune system in control of the adaptive immune system, with the ability, in the absence of "danger" in tissues, to suppress the adaptive immune response.

Dendritic Cells (DCs). The DCs are Antigen Presenting Cells (APCs) responsible for sampling the antigens from the tissues, including self and non-self antigens, and then presenting these antigens in the *thymus* for *T-cells* maturity. Immune system cells, inclusive of DCs, communicate with each other through secretion of specific molecules, termed as "signals". When a body cell undergoes apoptosis (planned cell death), the signals generated are different from that of a necrotic cell dying of pathogenic infection. The DCs are sensitive to relative concentrations of these signals in the fluid surrounding the cells in tissues. DCs express receptors on their surface that allow them to receive signals from the environment. DCs, therefore, act as information fusion agents where they receive information from different sources, process that information and then produce the appropriate *immunogenic* or *tolerogenic* response.
Depending upon the types of signals present in the residing tissue (*safe signals* or *danger signals*), the DCs may exist in one of the following three states:

Immature DCs. A DC initially arriving in the tissue is in *immature* state. In this form, it acts as a *phagocyte* to clear the tissue of cell debris, and also collects antigens, presenting them on the cell surface. An *immature* DC, when exposed to the different types of signals present in the tissue and depending upon the relative concentrations of these signals, transforms into the *semi-mature* or the *mature* state. A higher concentration of PAMPS (Pathogen Associated Molecular Patterns) and danger signals from the dying cells cause an *immature* DC to become *mature*. While signals resulting from apoptotic cells transform an *immature* DC into a *semi-mature* DC. In both states, the DC is able to migrate and present the collected antigens in the *thymus* for *T-cell* activation.

Semi-Mature DCs. *Thymus* is the immune system organ where *T-cells* undergo maturation. The *semi-mature* DCs present their collected antigens to *T-cells* in *thymus* in a tolerogenic context. Exposure to safe signals during the antigen collection period causes the *semi-mature* DCs to secrete cytokines that leads to *T-cell* suppression. The *T-cells* whose receptors bind to the antigens presented by the *semi-mature* DCs are de-activated, thus preventing the immune system to respond to these antigens.

Mature DCs. When an *immature* DC has had sufficient exposure to PAMPS and danger signals, it ceases to collect antigens and migrates to *thymus* as a *mature* DC. The cytokines secreted by *Mature* DCs have an immunogenic effect on the *T-cells* present in the *thymus*. If the *T-cell* receptors match any of the antigens presented by the *mature* DCs, the *T-cell* is activated. Activation of *T-cell* is needed to initiate the effector function for co-stimulation of B-cells during an adaptive immune response.

2.1 Applications of Danger Theory in AIS

The Danger Theory and dendritic cell behavior has found useful application in the design and development of artificial immune systems, in general, and anomaly detection in particular. The authors in [1] and [2] discuss how the latest immunological concepts proposed by the danger theory may be mapped to solve the intrusion

detection problem in computer security. They focus on identifying the various (danger) signals and carry out their functional analysis to drive the adaptive immune response. The Danger Project [14], with an aim to find the missing link between the AIS and intrusion detection, resulted in the development of the Dendritic Cell Algorithm (DCA), which was introduced in [6] as an abstract model for the dendritic cells interactions and behaviour. The algorithm features all the major behavioral aspects of the dendritic cells including the ability to sample multiple antigens, process signals, express costimulatory molecules and output cytokines, adopt differentiation pathways and present the antigens in an appropriate context. The preliminary results obtained indicated the suitability of the algorithm for anomaly detection. Further experiments were conducted on a machine learning dataset and detection of outgoing portscans, reported in [7]. On the basis of these experiments, the authors conclude that the DCA has potential as a classifier for static machine learning dataset and as an anomaly detector under real time conditions.

3. BEEAIS: ARTIFICIAL IMMUNE SYSTEM SECURITY

BeeAIS [11] is an AIS based security framework for the *BeeAdHoc* protocol. It is based on self non-self discrimination and performs anomaly detection using the *negative selection*. *BeeAIS* first learns the normal behavior of the system during an initial *learning* phase of 50 seconds, and then monitors the system for occurrences of abnormal patterns. The system, therefore, has the ability to detect previously unknown attacks.

Static Node Simulations. The authors in [11] compared the security characteristics of *BeeAIS* with its base protocol *BeeAdHoc* and the cryptographic security framework, *BeeSec* [12]. Simulations were performed in ns-2 using a static grid of 49 nodes. Static scenario was selected to make it easier to show the effect of attacks; with node mobility it becomes hard to demonstrate the attacks effects. The authors demonstrated that *BeeAIS* was able to detect a number of routing attacks.
Using the same simulation scenario, the authors also evaluated the protocol performance to determine how much the network performance of the base protocol, *BeeAdHoc*, gets degraded due to the additional processing and communication cost of the security solution. The ns-2 simulation results showed that the performance parameters for *BeeAIS* were close to that of *BeeAdHoc* and better than that for *BeeSec*, especially in the case of control overhead and energy efficiency. It was, therefore, concluded that *the AIS based security does not appreciably degrade the system performance compared to the cryptographic system*.

3.1 BeeAIS simulations under mobility

In this paper we evaluate the performance of the *BeeAIS* protocol under mobility and show that the *BeeAIS* self non-self model cannot adapt to a changing self caused by node mobility in MANETs. We perform simulations in ns-2 to compare the average throughput of *BeeAIS* with that of its base protocol *BeeAdHoc* and also with the classical MANET routing protocols, *DSR* and *AODV*. A rectangular area of operation, $2400 \times 480\ m^2$, is selected and the number of nodes are varied from 10 to 60. Each run is of 1000 seconds duration. Node movement is according to the "random waypoint" model. Each node in the network sends and receives data, comprising constant bit rate (CBR) peer-to-peer traffic at the rate of 30 packets/second. The results are averaged over five independent runs to factor out stochastic elements.

We define the protocol average throughput as, *"the total number of data bits delivered to destination nodes during the simulation, divided by the total simulation time"*. We computed the average throughput of *BeeAIS*, *BeeAdHoc*, *DSR* and *AODV* protocols. Figure-1 shows that the *BeeAIS* suffers from the lowest average throughput. To determine the cause for the *BeeAIS* low average throughput, we computed the average number of data bytes handed down to the *BeeAIS*, *BeeAdHoc*, *DSR* and *AODV* protocols by their transport layers for routing, along with the average number of data packets dropped by the protocols during the course of the simulation. Figure-2 shows the data bytes received by the protocols from their transport layers, normalized by the highest value. We see that the *BeeAIS* receives the minimum amount of data amongst all the protocols; 39.8% to 65.6% less than the *BeeAdHoc* protocol. Also, the *BeeAIS* dropped a higher number of data packets, Table-1. Compared to the *BeeAdHoc* protocol, *BeeAIS* dropped from 28.1% to 145.5% more packets. Consequently, *BeeAIS* has the lowest average throughput among the compared protocols.
Investigating the high data packet drop by *BeeAIS*, we measured the *BeeAIS* ability to detect the *self antigens (Ags)* as *self* when the frequent node movements cause the system self to change. Our results are shown in Table-2 for four different network scenarios. For each scenario, we measured the average number of antigens (scout Ags, forager Ags Type-I and forager Ags Type-II, terminology explained in [11]), which are received by the nodes over five independent simulation runs. Since these simulations do not involve generating routing attacks, all the antigens are self Ags. We determine the average number of self Ags detected as non-self Ags (false positive, FP) and as self Ags (true negative, TN). We then compute the false alrm rate (FAR), i.e the percentage of self Ags that are detected as non-self Ags. Our results in Table-2 indicate that the

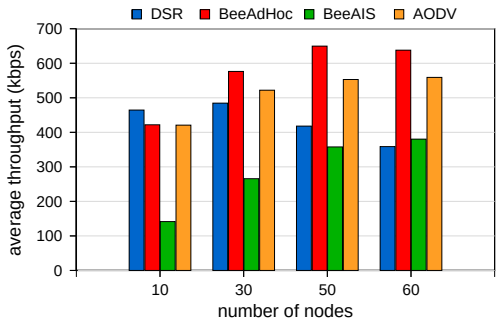


Figure 1: Comparison of protocol average throughputs

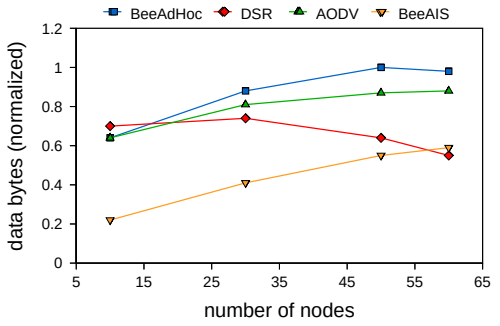


Figure 2: Comparison of application data handed down by TCP layer for transporting to destination nodes

Table 1: Comparison of data packets transported and dropped by protocols

Protocol	*Average number of packets*	*Number of nodes*			
		10	30	50	60
Beeadhoc	*generated by application*	100864.20	138241.80	156178.00	153409.00
	dropped - route not available	228.20	469.20	632.80	595.80
	dropped per 1000 generated	**2.26**	**3.39**	**4.05**	**3.88**
AODV	*generated by application*	100722.00	126313.60	134829.00	136482.20
	dropped - route not available	189.60	186.00	123.60	88.00
	dropped per 1000 generated	**1.88**	**1.47**	**0.92**	**0.64**
Beeais	*generated by application*	33930.60	63935.00	86204.40	91599.60
	dropped - route not available	188.40	405.60	477.60	454.80
	dropped per 1000 generated	**5.55**	**6.34**	**5.54**	**4.97**

Table 2: BeeAIS: Detection of self Ags as non-self Ags due to mobility

Number of nodes	*Ag type*	*Avg Ags rcvd*	*Avg Ags Detected*		FAR
			FP	TN	(% age)
10 nodes	***scout Ags***	586.40	395.00	191.40	**67.360**
	forager Ags Type-I	30654.80	4.80	30650.00	0.015
	forager Ags Type-II	38312.50	398.25	37914.25	1.039
30 nodes	***scout Ags***	8297.00	2542.20	5754.80	**30.639**
	forager Ags Type-I	58911.20	38.00	58873.20	0.064
	forager Ags Type-II	58873.20	1099.80	57773.40	1.868
50 nodes	***scout Ags***	14106.00	2247.60	11858.40	**15.933**
	forager Ags Type-I	81798.80	51.20	81747.60	0.062
	forager Ags Type-II	81747.60	3034.00	78713.60	3.711
60 nodes	***scout Ags***	16804.60	2157.40	14647.20	**12.838**
	forager Ags Type-I	86718.00	51.00	86667.00	0.058
	forager Ags Type-II	86667.00	6873.00	79794.00	7.930

scout Ags has a high FAR, causing as many as 67.36% scouts to be dropped in small MANETs; the figure drops to 12.84% with an increase in node density and higher node connectivity.

When scouts are dropped, new routes are not discovered and the nodes drop the foragers due to *route not available*. The dropping of foragers fools the Transmission Control Protocol (TCP) layer to initiate congestion control when its retransmission timer expires. The sending TCP thus reduces its congestion window and enters the slow start phase. This causes a reduction in the amount of data handed down by the TCP layer to the network layer for routing, which decreases the average throughput of the network.

3.2 BeeAIS Mobility Limitation

The low average throughput of the *BeeAIS* protocol in mobility scenarios can be attributed to the *BeeAIS* initial *learning* phase of 50 seconds. The detector sets generated are based upon the normal behavior (system self) learned during the *learning* phase. However, when node mobility causes the system self to change, the detector sets do not adapt to the changing self or to the changing non-self environment. Resultantly, the new and changed self is classified as non-self, and the relevant antigens, scouts or foragers, are dropped. Therefore, in order to allow mobility in *BeeAIS*, the system needs to incorporate a dynamic detector set, which keeps evolving with time keeping pace with the most recent system *self*, to allow for changes in the system *self* and the *non-self* space.

4. BEEAIS-DC: A DENDRITIC CELLS INSPIRED AIS SECURITY FRAMEWORK

BeeAIS-DC is a danger theory based AIS security framework, which represents a third approach towards securing the *BeeAdHoc* protocol, after the *BeeSec* [12] and *BeeAIS* [11]. The proposed framework models the dendritic cells to provide the capability of dynamically learning the system *self* and the *non-self* that keeps changing with the node mobility in MANET environment. Therefore, using the latest immunological discovery of the dendritic cell behavior, *BeeAIS-DC* overcomes the limitations in the basic *BeeAIS*, discussed in Section 3.2.

BeeAIS-DC employs a dynamic detector set, mediated through dendritic cells. We model the dendritic cells to sample the Ags (scouts) from the body tissues (the *node*); sampling includes both the *self* and the *non-self* Ags. Then depending upon the presence or absence of *danger signal*, the dendritic cells follow the differentiating pathways towards their terminal states, either *mature* or *semi-mature*, before presenting the sampled Ags for *T-cell* (detector) maturity in the *thymus*. The use of the *danger signal* in *BeeAIS-DC* precludes the need for an initial *learning phase* at system start up time. Moreover, the absence of *danger signal* allows the changed normal behavior of the system to be presented as new self, instead of being interpreted as non-self. The key aspect of the system, therefore, is its ability to start differentiating between the *self* and the *non-self* from quite early in the system operational life span, and also to adapt to a changing self and non-self environment. Details of the *BeeAIS-DC* algorithm are explained in the coming sections.

4.1 Antigens

In *BeeAIS-DC* we adopt the same Ag format as in [11]. An Ag is formed whenever a node receives a *forward* scout or a *backward* scout. The relevant header fields are extracted from the *scouts*, comprising the quadruple $\langle S_{sct}, D_{sct}, RtLen, node_{i-1} \rangle$. Antigens are represented in binary hamming shape space and have a string length of 52 bits each. An Ag has four genes, having lengths *16, 16, 4 and 16* bits and each gene represents a header field value. All the four collected genes are then concatenated to form an Ag.

4.2 Dendritic Cell (DC) Formation

When a scout is seen by a node for the first time, a dendritic cell is born. At birth, several attributes of the dendritic cells need to be

Table 3: List of BeeAIS-DC symbols and parameters

Symbol/Parameter	Description
$S_{sct}, D_{sct}, RtLen, node_{i-1}$	scout source, scout destination, source route length and the previous node address
Ag, T_{curr}, DC_{sct}	antigen, current time during simulation and scout dendritic cells
$Count_{FS}, Count_{BS}$	number of forward and backward scouts recvd
UDINT	fixed small interval of time defined for the system such that after each UDINT period the system checks for occurrences of danger signals and updates the dynamic detector set
THRESH-RCVD-FS, THRESH-RCVD-BS	upper limit for average forward or backward scouts to be received by a node before the context can be declared as dangerous
CO-STIMUL-SCT	co-stimulatory threshold for transition of dendritic cell state to MATURE, to allow presentation of the sampled non-self Ag in thymus for detector generation
NUM-DETS-SCT	number of detectors (antibodies) maintained by the system at any given time for matching the incoming scout Ags

initialized, which support their later functionality. These include:

DC Ag. The scout Ag is added to the dendritic cell. This represents the Ag sampled from the tissue that is later presented to the *T-cells* during their maturation in *thymus*.

DC Life. Dendritic cells live in the system for a short duration and then die. This ensures presentation of only the most recent system state in *thymus* at all times, and facilitates correct interpretation of the current system *self* and *non-self*.

DC State. State of a DC may be *immature*, *semi-mature* or *mature*. At birth, a dendritic cell is *immature*. When it samples the Ag and is exposed to *safe signals* it transitions to the *semi-mature* state; and if exposed to *danger signals* it differentiates to the *mature* state. It can then migrate to the *thymus* to present the sampled Ag.

On receiving a scout, the node needs to determine whether the same bee agent was processed earlier. The node therefore matches the S_{sct}, D_{sct} and the complete source route with respective values in the collected scout dendritic cells.
The life of a new born scout dendritic cell is determined by Equation-1. In case a similar scout reappears (matches an existing scout DC), the scout DC life is reset using the same relation. Also the count for receiving a *forward* or a *backward* scout ($Count_{FS}$ or $Count_{BS}$) for the matching DC is incremented. This information is used later by *BeeAIS-DC* to determine the occurrence of scout *danger signal* as explained in Section 4.3.

$$DC_{sct}\, life \;=\; T_{curr} \;+\; UDINT \tag{1}$$

4.3 Danger Signal Computation

Computation of *danger signal* by a node is central to the concept of DC based dynamic updation of detector set. A *danger signal* in Human Immune System occurs when there is evidence of necrotic cell death in the *tissue*, indicating damage to body due to pathogenic infection. In a mobile network scenario, damage to network may be irregular and inefficient routing behavior. Therefore, if there is evidence of routing problems in the network, the relevant *danger signal* might be raised.
DCs need activation by *danger signal* to change their state to *mature* before migration to *thymus* and presentation of the sampled Ags as *non-self*. The absence of *danger signal* results in a *semi-mature* state for the DCs with the sampled Ags regarded as self. Identification of a suitable *danger signal* in the network should thus allow incorporation of the changed and most recent *self* and *non-self* states in the system, for continuous updation of the detector set. In *BeeAIS-DC*, the computation of *danger signal* and updation of detector set is carried out at fixed, periodic intervals of time that are *update interval* (UDINT) seconds apart. While the system is in operation, the scout DCs at each node keep count of the forward and backward scouts received by the node during the last UDINT period of time. At the end of every UDINT period, the DCs determine the average forward and backward scouts received on each of the stored paths. Each time the computed averages exceed their thresholds (THRESH-RCVD-FS, THRESH-RCVD-BS) on a path, the co-stimulation level for that path is raised. Finally, when the co-stimulation level exceeds the co-stimulation threshold for scouts (CO-STIMUL-SCT), the *danger signal* for that scout is turned "HIGH". Now, the context for this DC becomes "dangerous", the DC turns its state to *mature* and migrates to *thymus* to present its sampled Ag as a positively identified *non-self* scout Ag. Equation-1 indicates that a scout DC survives a minimum of two consecutive UDINT time periods before completing its life and dying a natural death. Within this period, however, if a similar scout arrives again, the DC life is increased for another UDINT period. This implies that the *danger signal* would turn high only if within the life span of a DC the *non-self* Ag keeps arriving, is detected as suspected and the number of UDINT intervals within which the Ag is detected as suspected exceeds the threshold. This provides sufficient co-stimulation before raising the *danger signal* and helps to reduce the rate of false positives.
Once raised, the *danger signal* remains high for the subsequent (CO-STIMUL-SCT + 1) number of UDINT periods. This is to lower the false negatives in case the attack continues but skips detection in contiguous UDINT periods.

4.4 Detector Set Updation

In *thymus*, the process of *T-cell* maturation takes place in the presence of DCs. If the sampled Ag is presented by the DC in *semi-mature* state, the *T-cells* that match the Ag die. In other words, a population of *T-cells* is generated that is tolerant to *self* through elimination of *T-cells* whose antibodies match the self Ags. If the sampled Ag is presented in the *mature* state, the matching *T-cells* get activated and are then ready to help initiate the adaptive immune response.
On the same principle, in *BeeAIS-DC*, the purpose of the Ag sampling by DCs and determination of their states as *semi-mature* or *mature* is to affect the creation of antibodies or detector set for scouts that are *self* tolerant. The resulting detectors would then recognize or match only the *non-self* Ags and the activated ones would assist in initiating the required response against non-self Ags.
At system startup, to produce the required scout detctor set, random detectors are generated and subjected to negative selection with respect to the *self* Ags presented by the DCs in *semi-mature* state. At other times the existing detector set is used. If any detector matches a *self* Ag, it is removed from the system. This may result in the number of scout detectors to fall below the level (NUM-DETS-SCTS) specified for the system. Therefore, to make up for the lost detectors, the node again generates random detectors and adds only

those to the scout detector set that do not match the *self* Ags. This ensures that the resulting detectors are capable only of binding with the *non-self* Ags.
The Ags sampled by DCs, for whom the context was "dangerous", are presented by the DCs in *mature* state. The *mature* DCs cause activation of those *T-cells* that match the sampled *non-self* Ag. We model this activation of *T-cells* by matching all detectors with the Ags presented by DCs in *mature* state and changing the state of matching detectors from *naive* to *activated*. Now if any of the incoming Ags match these *activated* detectors, the Ags are detected as *non-self*.

4.5 Eliminating or Refreshing DCs

After every UDINT period of time, when the scout detector set has been updated, we need to update the scout DCs also. The DCs that die a natural death, i.e completed their lives during the last UDINT period, need to be eliminated from the system. The surviving DCs are then refreshed to restart the process of Ag sampling in tissues and determining the occurrence of *danger signal*. The state of the surviving *semi-mature* and *mature* dendritic cells is changed to *immature*. Moreover, the data gathering fields of the DCs ($Count_{FS}$, $Count_{BS}$) are also reset for collection of new data in the following UDINT period.

4.6 Matching Antigens and Detectors

During system operation, as a node receives the bee agents, it matches the scout Ags with the scout detector set to classify them as *self* or *non-self*. In case a scout Ag matches an activated scout detector, a *non-self* Ag is assumed to have been identified and the matching scout is dropped.

5. DEMONSTRATING ATTACK EFFECTS

To verify the security functionality of *BeeAIS-DC*, we implemented the system in network simulator, ns-2, and performed simulations to demonstrate its security capabilities. We use the same simulation scenario as in [12], comprising nine nodes in a simple topology, and perform the attack simulations on *BeeAIS-DC* to demonstrate the behavior of the protocol under the following conditions:

- **Normal Routing Behavior.** Fully functional *BeeAIS-DC* protocol in the absence of any routing attacks.
- **Partially Functional Under Attack.** *BeeAIS-DC* with all the protocol scout processing but without the bee agent drop action upon detection of a *non-self* Ag. Attacks in this case should be successful.
- **Fully Functional Under Attack.** Fully functional *BeeAIS-DC* including packet drop upon detection of *non-self* Ags. Launching of attacks in this case should fail.

5.1 Node Topology for Attacks

The network topology selected for demonstrating the effect of attacks is shown in Figure-3. It is a rectangular area of dimensions $1000 \times 500\ m^2$, where *node-0* is the source and *node-8* the destination. The source *node-0* has a TCP traffic source that generates constant bit rate (CBR) data traffic for the sink connected to the destination *node-8*.
In Figure-3, we can see three distinct paths between the source and the destination nodes; path *0-7-8*, *0-5-6-8* and *0-1-2-3-4-8*. The path *0-1-2-3-4-8* being the least optimum should virtually have no packets routed over it under normal routing conditions.

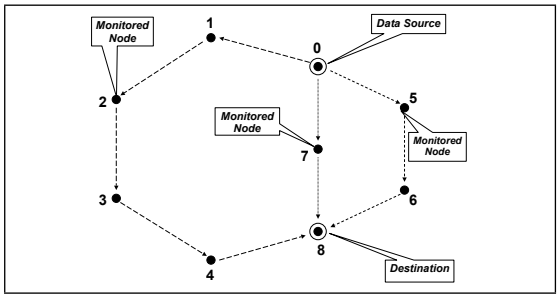


Figure 3: Node topology selected for attacks

5.2 Attacks on BeeAIS-DC

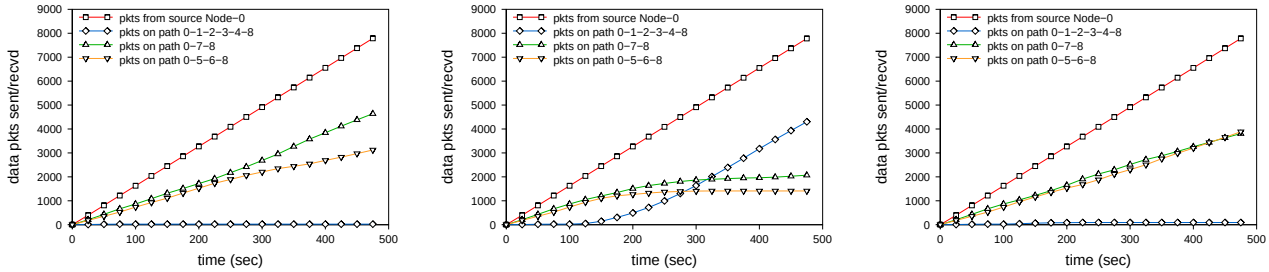
To launch routing attacks on *BeeAIS-DC* protocol we developed an *attacker framework* in ns-2 and launched two different types of routing attacks. During attacks we monitor the routed traffic at three points in the network, *node-2*, *node-5* and *node-7*, and then generate traffic maps to indicate the success or failure of these attacks.

Attack-1: Forging Forward Scout. This attack is launched at time t=100 seconds after the start of simulation. The attacker *Node-4* launches forged forward scouts into the network trying to install a forged route *0-1-2-3-4-8*. These fake packets have *Node-0* as the S_{sct} and *Node-8* as the D_{sct}. Figure-4(b) shows the result of the attack when *BeeAIS-DC* is running with partial functionality. In this case, as the forward scouts are received at D_{sct} *Node-8*, they are returned to the S_{sct} *Node-0*, causing the forged route to be established. Subsequently, all foragers start to follow the forged route and the attack is successful.
However, in the case of *BeeAIS-DC* running with full functionality, the attack is not successful, Figure-4(c). At D_{sct} *Node-8*, when the arrival rate of forged forward scouts exceeds the threshold in more than the CO-STIMUL-SCT number of contiguous update intervals, the scout *danger signal* due to forward scouts is turned high. The relevant DCs then transition to *mature* state and present the sampled forged scout Ags as *non-self* Ags for scout detector set updation. Resultantly, the scout detectors are able to match the forged scouts and drop them to make the attack unsuccessful. Therefore, the *BeeAIS-DC* routing behavior remains the same as it was without attacks, Figure-4(a).

Attack-2: Forging Backward Scout. The attack involving spoofed backward scouts is launched by *Node 2* at time t=100 seconds. The attack is successful in the case of *BeeAIS-DC* running with partial functionality due to the forged path *0-1-2-3-4-8* getting established at S_{sct} *Node 0*. As shown in Figure 5(b), the malicious *Node 2* is able to divert the data packets onto the attack path. However, when attack is launched with *BeeAIS-DC* running with full functionality, the D_{sct} *Node 8* is able to detect the forged backward scouts as *non-self* Ags and drop them. Consequently, as seen in Figure 5(c), the forged path *0-1-2-3-4-8* is not established and the protocol routing behavior remains the same as in the case of *BeeAIS-DC* without attacks, Figure-5(a).

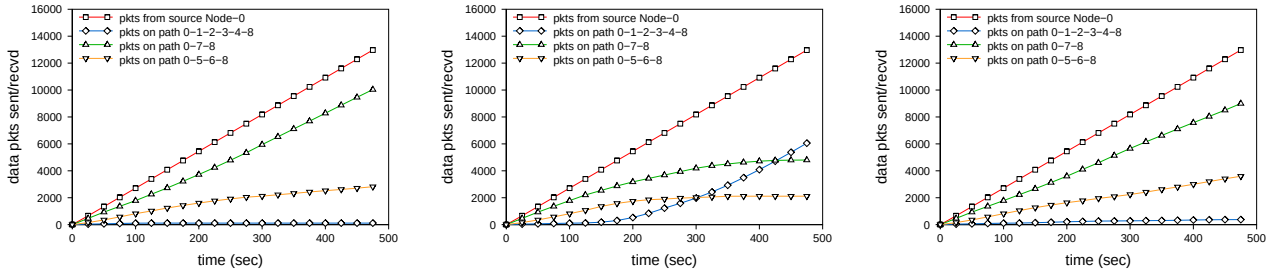
6. NETWORK PERFORMANCE

We studied the network performance of our proposed security framework, *BeeAIS-DC*, through extensive simulations in the network simulator ns-2 and compared *BeeAIS-DC* with the base protocol *BeeAdHoc*. We use the same simulation scenario as in Section-3.1. The performance metrics used are:



(a) Normal routing without attacks (b) Under attack without AIS pkt dropping (c) Under attack with AIS pkt dropping

Figure 4: BeeAIS-DC Attack-1: Forging Forward Scout



(a) Normal routing without attacks (b) Under attack without AIS pkt dropping (c) Under attack with AIS pkt dropping

Figure 5: BeeAIS-DC Attack-2: Forging Backward Scout

Throughput. *The number of data bits delivered to the application layer at the destination node in a unit interval of time.*
Packet Delivery Ratio. *The ratio of data packets successfully delivered to destination nodes and total number of packets generated for those destinations.*
Latency. *The average difference in time when a packet is generated at the source and when it got delivered to the destination.*
Average Hops. *The average number of hops for all the paths traversed by data packets.*
Transmission efficiency. *The number of data bytes delivered to the application at destination nodes at the cost of a unit control byte.*
Average Control Overhead. *Total number of control bytes transmitted by all nodes in the network.*

Figure-6 gives the results for our extensive performance evaluation of the *BeeAIS-DC* algorithm. We have compared the performance of the security framework with its base protocol *BeeAdHoc* as well as with the *AODV* and *DSR*. We find that the AIS overhead of *BeeAIS-DC* does not significantly degrade the performance of the *BeeADHoc* protocol; all performance parameters for both *BeeAdHoc* and *BeeAIS-DC* are nearly the same. At the same time, we can see that in most of the cases, the *BeeAIS-DC* outperforms the state of the art classical MANET routing protocols, *AODV* and *DSR*, which are without the overhead of the security frameworks. Our proposed security framework is thus able to provide protection against the routing attacks while giving the same or slightly better network performance compared to the other MANET routing protocols.

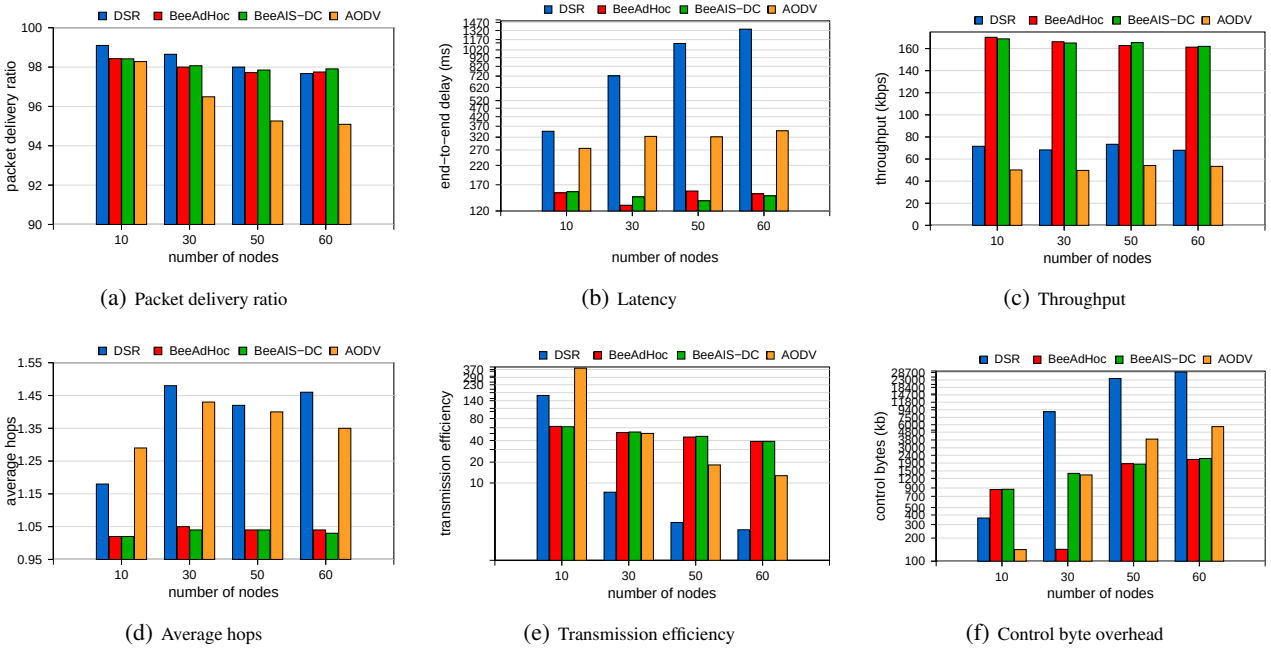
We also compared the network average throughputs for all protocols, Table-4. The average throughput for *BeeAIS-DC* is quite close to that of *BeeAdHoc*. This indicates that the *BeeAIS-DC* does not suffer from the mobility limitation as *BeeAIS*. The DCs are correctly able to identify the newly changed system *self* and do not cause dropping of scouts with related reduction in the network average throughput.

Protocol	*Number of Nodes*			
	10	30	50	50
BeeAdHoc	421.86	576.36	649.67	637.91
DSR	464.50	484.54	418.01	358.72
AODV	420.81	522.00	553.02	559.09
BeeAIS-DC	419.68	570.85	656.62	660.09

Table 4: Average network throughputs for protocols (kbps)

7. CONCLUSION AND FUTURE WORK

In this paper we have introduced a secure routing framework, *BeeAIS-DC*, for the Bio/Nature inspired MANET routing protocol, *BeeAdHoc*. Our proposed framework is based upon the Danger Theory, modelling the function and behavior of the dendritic cells as in the Biological Immune System. *BeeAIS-DC* constitutes a third approach towards securing the *BeeAdHoc* protocol against malicious routing attacks; the earlier approaches being the cryptographic system, *BeeSec*, and the self non-self discrimination based AIS security system, *BeeAIS*.
We first showed that the self non-self based system, *BeeAIS*, suffers from a poor average network throughput. This is because the frequent node movements in the mobile adhoc network cause the system *self* to change and the static detector set of *BeeAIS* starts to detect the newly changed *self* as *non-self*. To overcome this limitation, we implemented a danger signal based AIS, the *BeeAIS-DC*. Our proposed system models the dendritic cells to provide the capability of dynamically updating the detector set to cater for a changing system *self* and the *non-self*. By sensing the presence/absence of danger in tissues, *BeeAIS-DC* is able to differentiate between the



(a) Packet delivery ratio (b) Latency (c) Throughput

(d) Average hops (e) Transmission efficiency (f) Control byte overhead

Figure 6: Performance results comparing BeeAIS-DC with BeeAdHoc, DSR and AODV

newly changed *self* and the malicious *non-self* behavior.
We also extensively evaluated the network performance of *BeeAIS-DC*, comparing it with *BeeAdHoc*, *DSR* and *AODV* protocols. Our results show that the *BeeAIS-DC* provides protection to a mobile adhoc network against routing attacks with minimal security overhead on network performance. In future, we intend to extend the *BeeAIS-DC* to cater for attacks involving tampering/forging of forwagers and also to detect the dropping attacks in MANETs.

8. REFERENCES

[1] U. Aickelin, P. Bentley, S. Cayzer, J. Kim, and J. McLeod. Danger theory: The link between ais and ids. In *Proceedings of the ICARIS-2003, LNCS 2728*, pages 147–155, 2003.

[2] U. Aickelin and S. Cayzer. The danger theory and its applications to artificial immune systems. In *Proceedings of the ICARIS-2002*, pages 141–148, 2002.

[3] U. Aickelin, J. Greensmith, and J. Twycross. Immune system approaches to intrusion detection - a review. In *Proceedings of the ICARIS-2004, LNCS 3239*, pages 316–329, 2004.

[4] G. Di Caro, F. Ducatelle, and L.M. Gambardella. Anthocnet: An adaptive nature inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications*, 16(2):443–455, 2005.

[5] L. N. de Castro and J. Timmis. *Artificial immune systems: a new computational intelligence approach*. Springer, 2002.

[6] J. Greensmith, U. Aickelin, and S. Cayzer. Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection. In *Proceedings of the ICARIS-2005, LNCS 3627*, pages 153–167, 2005.

[7] J. Greensmith, J. Twycross, and U. Aickelin. Dendritic cells for anomaly detection. In *Proceedings of the CEC*, pages 664–671, 2006.

[8] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. *Wireless Networks*, 11(1-2):21–38, 2005.

[9] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, pages 153–181. 1996.

[10] P. Matzinger. Tolerance, danger, and the extended family. *Annual Review of Immunology*, 12:991–1045, 1994.

[11] N. Mazhar and M. Farooq. Beeais: Artificial immune system security for nature inspired, manet routing protocol, beeadhoc. In *Proceedings of ICARIS-2007, LNCS 4628*, pages 370–381, Aug, 2007.

[12] N. Mazhar and M. Farooq. Vulnerability analysis and security framework (beesec) for nature inspired manet routing protocols. In *Proceedings of GECCO-2007*, pages 102–109, July, 2007.

[13] C. Perkins and E. Royer. Ad-hoc on-demand distance vector routing. In *Proceedings of 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, Feb 1999.

[14] Danger Project. http://www.dangertheory.com.

[15] M. Roth and S.Wicker. Termite: Ad-hoc networking with stigmergy. In *Proceedings of IEEE GLOBE-COM*, Dec 2003.

[16] S. Sarafijanovic and J.Y. Le Boudec. An artificial immune system approach with secondary response for misbehavior detection in mobile ad-hoc networks. *IEEE Transactions on Neural Networks*, 16(5), Sep 2005.

[17] H.F. Wedde, M. Farooq, T. Pannenbaecker, B. Vogel, C. Mueller, J. Meth, and R. Jeruschkat. Beeadhoc: an energy efficient routing algorithm for mobile ad hoc networks inspired by bee behavior. In *GECCO*, pages 153–160, 2005.

[18] H.F. Wedde, C. Timm, and M. Farooq. Beehiveais: A simple, efficient, scalable and secure routing framework inspired by artificial immune systems. In *PPSN*, pages 623–632, 2006.

[19] Manel Guerrero Zapata. Secure ad hoc on-demand distance vector (saodv) routing. Internet-Draft, draft-guerrero-manet-saodv-05.txt, February, 2005.