Journal of Information Security, 2014, 5, 207-217
Published Online October 2014 in SciRes. http://www.scirp.org/journal/jis
http://dx.doi.org/10.4236/jis.2014.54019

Scientific Research

Security Enhanced Adaptive TCP for Wireless Ad Hoc Networks

Waleed S. Alnumay

Computer Science Department, King Saud University, Riyadh, KSA
Email: wnumay@ksu.edu.sa

Received 22 August 2014; revised 20 September 2014; accepted 13 October 2014

Abstract

TCP is a reliable transport protocol designed to perform well in wired networks where packet losses are due congestion. However in wireless ad hoc networks, where packet losses are due to channel errors or link failures between mobile nodes, TCP degrades its performance. Further, it lacks certain protection mechanisms from internal and external malicious nodes. In this paper, a security enhanced and adaptive TCP, namely SA-TCP, has been proposed for wireless ad hoc networks. SA-TCP uses network layer information to detect various types of packet losses and adjusts the value of congestion window dynamically according to the conditions of the dynamic network. It works normally to collect the samples of congestion window and calculates the mean from these samples to set the value of future congestion window. SA-TCP also adjusts the value of congestion window limit according to network conditions. In order to make SA-TCP suitable in highly vulnerable wireless ad hoc networks, a less complex identity-based public key cryptography has been integrated with the proposed protocol. The three-way handshaking process of SA-TCP is made secure by generating a secret session key on-fly between source-destination. Simulation results show that SA-TCP gives higher throughput compared to the popular *New Reno* and ATCP in different wireless ad hoc network scenarios.

Keywords

WANET, TCP, Congestion Window, Security

1. Introduction

Performance of transmission control protocol (TCP) for wireless ad hoc networks (WANETs) is an active area of research and it has received a lot of attention for its applicability in military operations, disaster rescue missions and other civilian applications. TCP performs well in wired environment where packet losses are

mainly due to congestion in the network, but in case of WANETs, a marked degradation in the performance of TCP has been shown, where packet losses are due to channel errors or link failures between mobile nodes [1]. The main reason is the inability of TCP to distinguish the various causes of packet losses. The newer versions of TCP, such as Tahoe, Reno and New Reno [2] also exhibit similar properties [3].

A number of transport layer protocols have been proposed in the literature [1] [4]-[9] to enhance TCP's performance in mobile ad hoc environments. However, it may be noted that when a packet drop occurs, most of the protocols are unable to detect the actual reason of packet loss (congestion, link failure, lossy channels or high bit error rates). In WANET, due to dynamic nature of the network and variable number hops between source and destination, the fairness and efficiency get degraded. It is also seen that the existing protocols are not good enough in setting its parameters like congestion window, congestion window limit, round trip time and the retransmission timeout timers.

The existing transport layer protocols lack certain protection from dynamic internal and external malicious activities and hence relies on separate security mechanisms. Existing secure routing protocols though ensure the correctness of the route discovery, may not provide secure end-to-end data delivery at transport layer. The major security threats [10] associated with transport layer protocols in WANET are such as SYN *flooding*, *session hijacking* and ACK *storm*. Therefore, following are some important goals required to be met while designing a transport layer protocol for wireless ad hoc networks: a) Maximizing the throughput per connection; b) Providing both reliable and unreliable connections as per the requirements of application layer; c) Providing adaptability to the dynamics of the network such as rapid change in topology and changes in the nature of wireless links; d) Utilizing available bandwidth efficiently; e) Handling resources such as battery power and buffer sizes efficiently in resource-constraint environment; f) Proper utilization of information from the lower layers of the protocol stack to improve network throughput; g) Maintaining end-to-end semantics; and h) Considering the security aspects as regards WANETs to make itself complete and self-sufficient.

In this paper, we have modified New Reno protocol for wireless ad hoc networks so that it can take care of the above issues. We call this modified protocol as *Secure and Adaptive TCP* (SA-TCP) and present the proposed protocol in Section 4. The proposed protocol while in operation utilizes the feedback from the network layer so that it can detect different types of packet losses and act accordingly. It also dynamically adapts the parameters (e.g., congestion window, congestion window limit) according to the conditions of the dynamic network, which in turn reduces the chances of congestion and MAC layer collisions in the network. This increases the network throughput and the performance is enhanced significantly. Further, SA-TCP uses an identity based public key cryptography that does not require a trusted third party, which is absent in wireless ad hoc networks. It generates a session secret key on-fly between source-destination nodes to make the three-way handshaking process secure. This proposed security mechanism incurs less overhead as it does not need a certificate for the purpose of authentication.

Paper Overview: The rest of the paper is organized as follows: the related work in enhancing and securing the transport layer protocols is presented in Section 2. In Section 3 we present the system model that is followed in this paper, which is followed by Section 4, wherein the proposed secure and adaptive transmission control protocol is presented. Section 5 shows the simulation results, and finally, conclusions of the paper are given in Section 6.

2. Related Work

A number of protocols have been proposed in the literature [1] [4]-[9] [11] to enhance the performance of transport layer in wireless ad hoc networks. These existing transport layer protocols can be classified into *layered* and *cross layer* approaches [12]. Under layered approach, a protocol works independently at a particular layer of OSI model. Under cross layer approach, the protocol works by interacting between two layers. The proposed SA-TCP presented in this paper utilizes transport layer and network layer, and therefore it falls under cross layer approach. TCP DOOR [13] and fixed *RTO* [14] are under layered approach, whereas, ELFN [1], TCP-F [4], TCP-BuS [9], ATCP [8] and P-TCP [11] are under cross layer approach. In the following paragraphs, we present the cross layer approaches used to design/enhance transport layer for wireless ad hoc networks.

Explicit Link Failure Notification (ELFN) [1] sends a route error message of DSR protocol to notify link failure to the source node. On receiving the route error message, the source node stops the congestion control mechanism and goes into a standby mode. Once the route is re-established, the source node restarts the data

transmission with the same congestion window size and RTO timer as was at the time of link failure. This may either lead to more congestion in the new route giving low throughput or inefficient utilization of bandwidth. TCB-Bus [9] is another protocol that works similar way as ELFN. Additionally, it buffers of packets at the intermediate node, named Pivoting Node (PN), which detects the link failure on an active route. The downside of this protocol is that PN has to drop all buffered packets whenever the PN fails to find a new partial route to the destination.

Chandran *et al.* proposed TCP-F [4] that utilizes network layer feedback from intermediate nodes for detecting link failures in wireless ad hoc networks. On detecting a link failure by an intermediate node, a *route failure notification* (RFN) message is sent back to the source node. On receiving the RFN, the source node puts the TCP connection into a snooze state, resets all the timers and stops further transmission. If an intermediate node finds a new route to the destination, it sends a *route re-establishment notification* (RRN) message to the source node. On receiving the RRN, the source node again activates the TCP connection. The main drawback of TCP-F is that upon reactivation of TCP connection the size of the congestion window remains same as before which may lead to either congestion or inefficient bandwidth utilization.

Liu and Singh proposed Ad hoc TCP (ATCP) [8] that provides a robust solution as it considers most of the issues related to TCP in wireless ad hoc networks. However, ATCP performs poorly due to inefficient congestion window adaptation whenever a link failure occurs [15]. Upon detection of a link failure, it enters into slow start phase and increments the congestion window in a slow pace. P-TCP [11] takes feedback from network layer to detect different types of packet losses and adjusts its parameters according to the varying wireless ad hoc network environment.

All of the above transport protocols, except P-TCP, do not consider the security aspects. In a multi-hop wireless ad hoc network, an intelligent attacker may place itself in an active route and later it can spoof an address of a node in order to start its malicious activities by dropping, forging, misrouting or injecting data packets. Therefore, it is necessary to design a secure transport layer protocol for secure data transmission between source and destination nodes. In the following paragraphs, we present some exiting work on securing TCP for wireless ad hoc networks.

The scheme proposed in [16] establishes different routes through different intermediate nodes and sends multiple shares of a message through them at different time slots. However, this scheme is suitable for delay tolerant static networks. Papadimitratos and Haas proposed SMT [17] that transmits all the shares of a message simultaneously through the multiple independent routes rather than in different time slots. However, both of the schemes assume that multiple routes exist in a dynamic wireless ad hoc network which may not be realistic in some cases.

In [18], an ID based secure TCP scheme is proposed. This scheme can securely transmit data using a Diffie-Hellman [19] session key. It assumes that the public-private key pairs along with IDs of nodes are distributed before the network deployment. In this protocol, a node may not be able to change its ID throughout the lifetime of the network and therefore the protocol is secured. However, the protocol has considerably high overhead due to key generation in each session. P-TCP [11], uses an identity-based public key cryptography to generate a session key for secure data transmission between source and destination nodes. However, this protocol may fail when a malicious node exists on the path.

IPSEC [20] is a popular security mechanism mainly used in wired networks to mitigate most of the attacks discussed in Section 1. However, it does not provide an intermediate node to directly access an IP header of a transmitted packet. The transport layer protocols proposed for WANETs have to rely on information fed back from the intermediate nodes (e.g., *Explicit Congestion Notification* (ECN) [21]), and hence IPSEC cannot be incorporated with these protocols [22]. Similar is the case with PCT, SSL and TLS [23] proposed for the wired networks.

3. System Model and Key Distribution

Here, we assume SDRP [24] as the ad hoc routing protocol used in the network layer. We also assume that *Explicit Congestion Notification* (ECN) [21] is enabled and the public/private key pairs are distributed to each node prior to the network starting its operation. The public (PK_A)/private (SK_A) key pair of a node A are distributed using the following technique [25] [26]: Let us assume that Q_1 and Q_2 be the two groups of a prime order O and P_1 and P_2 be the generators of Q_1. A bilinear pairing is a map $e: Q_1 \times Q_1 \to Q_2$ which

has the following properties:

- Bilinear: $e(xP_1, yP_2) = e(P_1, P_2)^{xy}$, $\forall x, y \in Z_Q$;
- Non-degenerate: There exist $P_1, P_2 \in Q_1$ such that $e(P_1, P_2) \neq 1$;
- Computable: There exist an efficient algorithm to compute $e(P_1, P_2)$ $\forall P_1, P_2 \in Q_1$.

The security parameters Q_1, Q_2, e, H_1, H_2, P_1, $P_{\text{pub}} = mP_1$ are publicly known by all the nodes in the network. Here, the network master key m is shared by a set of n nodes. Threshold cryptography technique is used where t nodes in a network is required to construct the master key m. H_1 and H_2 are the two cryptographic hash functions such that $H_1 : \{0,1\}^* \to Q_1$ and $H_2 : \{0,1\}^* \times Q_1 \to Z_Q$. Node A with identifier ID_A (*i.e.*, hardware address of A), will have a public key $PK_A = H_1(\text{ID}_A)$ and a private key $SK_A = mPK_A$. In addition, shared key (K) between nodes A and B is generated in a non-interactive fashion [27]: $K_{AB} = e(SK_A, PK_B) = e(PK_A, SK_B)$. The above key distribution is necessary for authentication of nodes in the ad hoc network to nullify the attacks.

4. SA-TCP: Secure and Adaptive Transmission Control Protocol

In this section, we present the proposed modification to TCP New Reno, named SA-TCP, for wireless ad hoc networks. This work is an extension of our earlier work [28]. SA-TCP handles network congestion in three phases: *slow start*, *congestion avoidance* and *congestion detection*. In *slow start* and *congestion avoidance* phases, SA-TCP enhances the performance of TCP New Reno by adjusting the values of congestion window (CWND) and congestion window limit (CWL) according to dynamic network conditions. Therefore, SA-TCP minimizes the chance of congestion and MAC layer collisions in the network.

In congestion avoidance phase of SA-TCP, whenever there is a change in size of CWND, it is taken as a sample and this sample value of $(t-1)$-th congestion window size has been denoted here as $X(t-1)$. SA-TCP operates in normal mode (*i.e.*, similar to original TCP New Reno) for collecting samples of earlier CWND sizes $X(t-1)$ (where $i = 1, 2, 3, \cdots, L$). It calculates the value of current congestion window $X(t)$ from these L samples using the following Equation (1):

$$X(t) = \frac{1}{L}\sum_{i=1}^{L} X(t-i) \tag{1}$$

SA-TCP also sets the sizes of future CWND sizes in case the network condition remains unchanged. The proposed algorithm collects such L samples while working in the normal mode and creates the future congestion window sizes. Here, the value of L is an important factor to predict the current congestion window size. Using a smaller value of L may give inaccurate prediction of the network condition, whereas a larger value of L may make SA-TCP behave like TCP New Reno. We use a least-square method [11] [29] to compute the value of L as follows:

$$S_R = \sum_{i=1}^{L} (\text{CWL} - \text{CWND}_i)^2 \tag{2}$$

Here $\text{CWND}_i <= \text{CWL}$, and CWND_i is the i-th sample of the congestion window that is predicted. We select the value of L corresponding to the minimum sum of residuals S_R. These ensure that the value of congestion window CWND will not exceed the value of congestion window limit CWL [11].

SA-TCP also reduces the chance of MAC layer collisions by restricting the value of the congestion window according to the equation [30] $\text{BDP}_{\text{Max/Min}} = N/k$, wherein $\text{BDP}_{\text{Max/Min}}$ is the upper/lower bound of bandwidth delay product, $(1/8) \le k \le (1/4)$ and RH is the number of round-trip hop-count from source to destination nodes on the path. SA-TCP source node obtains the value of RH through RREP message of SDRP [24] routing protocol.

In *congestion detection* phase, SA-TCP detects the congestion in the network using the *explicit congestion notification* (ECN) [21] bit. When a node on an active route detects the congestion, it sends an ACK segment to the source node with ECN bit enabled. On receiving the ACK segment, the source node reduces both sizes of present CWL and CWND to half its size and sets this reduced CWND size as the slow start threshold (ssthreshold) value. Thereafter it sets the value of CWND size to 1 and enters into the slow start phase [11]. SA-TCP handles congestion using the pseudo code presented in Algorithm 1.

Apart from segment loss due to congestion, segment may be lost due to either lossy channel or a link failure

between the source and destination nodes. In order to detect a link failure on an active route, SA-TCP uses the *route error* (RERR) message of SDRP [24] routing protocol. Here, it uses the concept of P-TCP [11] while setting up its parameters after the route is re-established. The source node stops transmitting segments to the destination once it detects a link failure and waits till the route is re-established. When the route is re-established, the source node obtains the value of round-trip hop-count (RH) from *route reply* (RREP) message of [24] routing protocol and calculates the value of congestion window limit CWL for the new route. Further, if the size of congestion window CWND is higher than the size of CWL, SA-TCP source node reduces the size of CWND to half. Otherwise the source node keeps the value of CWND same as was at the time of link failure. At this point, SA-TCP enters back to its original phase of *congestion avoidance* or *slow start* as it was at the time of link failure. The above condition is necessary to ensure that the size of CWND never exceeds the size of CWL even when the route is re-established. Hence SA-TCP minimizes the chance of congestion and collisions in the network. Algorithm 2 presents the pseudo code for SA-TCP link failure handler.

Algorithm 1: *congestion-handler for SA-TCP*

```
if ACK is received then
    if ECN bit is enabled then
        ssthreshold = CWND/2;
        CWND = 1;
        CWL = CWL/2;
        exit and enter into slow start phase;
    end
end
```

Algorithm 2: *link-failure-handler for SA-TCP*

```
if source node detects a link failure then
    if route is reestablished by receiving RREP then
        Obtain the value of RH from RREP;
        if RH <= 4 then
            CWL = 2;
        else
            CWL = RH/5;
        end
        if CWL < CWND then
            CWND = CWND/2;
        end
        back to original phase (slow start or congestion avoidance);
    end
end
```

Security

This section describes a mechanism to secure the three-way handshaking connection establishment and connection termination processes of SA-TCP. Here, source (S) and destination (D) nodes of the end-to-end connection to be established first exchange their ID_S (ID_S and ID_D) and random numbers (r_S by source node and r_D by destination node) to each other in signed (σ_S) RREQ_S and signed (σ_D) RREP_D messages respectively. Each intermediate node I on the path also generates a random number r_I and sends it to source and destination nodes in signed (r_I) RREQ_I and RREP_D messages respectively. The signature scheme proposed in [25] is used here for the purpose of authentication. Thereafter node S generates a session secret key $K_{SD} = e(r_S SK_S, R_S PK_D)$, wherein SK_S is the private key of node S, PK_D is the public key of node D and $R_S = \sum_{i=1}^{J} r_i$ (here J is the number of intermediate nodes between source and destination nodes). Subsequently S starts three-way handshaking connection establishment process with D similar way to the scheme given in [18]. It first generates an initial sequence number (ISN_S) from a monotonically increased random number (M_S) and a hash value of source port address (PORT_S), destination port address (PORT_D), ID_S, ID_D and K_{SD}. S also computes an authentication tag (δ_S) on $\mathrm{SYN}(\mathrm{ISN}_S)$ segment using a HMAC function and a session secret key K_{SD}. This $\mathrm{SYN}(\mathrm{ISN}_S)$ segment is sent along with the tag δ_S by S to D.

On receiving the $\mathrm{SYN}(\mathrm{ISN}_S)+\delta_S$ from S, D computes the session secret key $K_{DS} = e(r_D SK_D, R_D PK_S)$, wherein SK_D is the private key of D, PK_S is the public key of S and

$R_D = \sum_{i=1}^{J} r_i$. It also generates the authentication tag (δ_G) on received $\text{SYN}(\text{ISN}_S)$ segment using the hash function and K_{DS}. If the generated tag (δ_G) and the received tag (δ_S) are same, then D concludes that S is authenticated. On successful authentication, D generates ISN_D and an authentication tag (δ_D) on $\text{SYN}(\text{ISN}_D)+\text{ACK}(\text{ISN}_S+1)$ segment in the similar way to S. This $\text{SYN}(\text{ISN}_D)+\text{ACK}(\text{ISN}_S+1)$ segment is sent along with δ_D by D to S. On receiving $\text{SYN}(\text{ISN}_D)+\text{ACK}(\text{ISN}_S+1)+\delta_D$, S computes the authentication tag (δ_G) on received $\text{SYN}(\text{ISN}_D)+\text{ACK}(\text{ISN}_S+1)$ segment using the session secret key K_{SD}. If both of the tags (generated δ_G and received tag δ_D) are same then S concludes that D is authenticated. S also sends $\text{ACK}(\text{ISN}_D+1)+\delta_S$ segment to D. D generates the tag (δ_G) on received $\text{ACK}(\text{ISN}_D+1)$ segment and verifies it with the received tag (δ_S). If both tags are same, D finishes the three-way handshaking connection establishment process and also allocates resource for S and starts sending of data messages along with authentication tags. The pseudo code of three-way handshake connection establishment processes for source (S) and destination (D) are given in Algorithm 3 and Algorithm 4 respectively. SA-TCP secures the three-way handshake connection termination process in the same way as connection establishment process. It may be noted here that an intermediate node sends ECN in the ACK segments and authentication tags whenever the node detects congestion.

Analysis: The proposed SA-TCP sends all the segments along with authentication tag and ID of the sender. Here, an ID of a network node can be spoofed by an attacker but it may not be able to generate the valid authentication tags. This is due to the fact that it is difficult for the attacker to know the private keys and the master key of a network. The master key is shared between n nodes in the network where it may only get disclosed if $t+1$ nodes gets compromised. Hence the chances of SYN flooding and segment forging attacks are very low in SA-TCP. The proposed protocol generates the initial sequence number (ISN) from a monotonically increasing random number (R) and a hash value of source port (PORT_S), destination port (PORT_D), source ID ID_S, destination ID ID_D and a session secret key. Therefore, it is also difficult for the attacker to guess the ISN and hijack a session or creating ACK storm in the network.

It can also be seen that SA-TCP does not require any certificate or large sized public key for authentication. It uses identity-based cryptography where the public key of a node is generated from its ID (*i.e.*, 48-bit hardware address). Therefore, SA-TCP only requires to send the ID with the segments thereby eliminating the need of sending the public key. Thus, we find that the proposed protocol has low overhead. However, the session secret key has to generate for each session, which increases the protocol overhead slightly. The following equation

Algorithm 3: *SA-TCP Source*

```
1  Generate random number r_S;
2  Generate σ_S = H(SK_S, RREQ(ID_S, r_S));
3  Send RREQ(ID_S, r_S) + σ_S to destination;
4  if RREP(ID_D, r_1, r_2, ..r_D) + σ_D is received from destination then
5      Compute PK_D = H_1(ID_D);
6      Compute σ_G = H(PK_D, RREP(ID_D, r_1, r_2, ..r_D));
7      if σ_G == σ_D then
8          route is secured;
9          destination is authenticated;
10         call three-way-connection-establishment-src();
11     end
12 end
13 three-way-connection-establishment-src() {
14 Generate R_S = Σ_{i=1}^{J} r_i;
15 Compute session key K_SD = e(r_S SK_S, R_S PK_D);
16 Compute ISN_S = M_S + H(PORT_S, PORT_D, ID_S, ID_D, K_SD);
17 Generate SYN(ISN_S);
18 Compute δ_S = H(SYN(ISN_S), K_SD);
19 Send SYN(ISN_S) + δ_S;
20 if SYN(ISN_D) + ACK(ISN_S + 1) + δ_D is rcvd then
21     Generate δ_G = H((SYN(ISN_D) + ACK(ISN_S + 1), K_SD);
22     if δ_G == δ_D then
23         Connection established;
24         Generate ACK(ISN_D + 1);
25         Compute δ_S = H(ACK(ISN_D + 1), K_SD);
26         Send ACK(ISN_D + 1) + δ_S;
27     end
28     else
29         Drop SYN(ISN_D) + ACK(ISN_S + 1) segment;
30     end
31 end
32 }
```

Algorithm 4: *SA-TCP Destination*

```
if RREQ(ID_S, r_1, .., r_S) + σ_S is received from source then
    Compute PK_S = H_1(ID_S);
    Compute σ_G = H(PK_S, RREQ(ID_S, r_1, .., r_S));
    if σ_G == σ_S then
        route is secured;
        source is authenticted;
        Generate random number r_D;
        Compute σ_D = H(SK_D, RREP(ID_D, r_D));
        Send RREP(ID_D, r_2) + σ_D towards source;
        call three-way-connection-establishment-dst();
    end
end
three-way-connection-establishment-dst {
if SYN(ISN_S) + δ_S is received then
    Compute PK_S = H_1(ID_S);
    Compute R_D = Σ_{i=1}^{J} r_i;
    Generate session key K_DS = e(r_D SK_D, R_D PK_S);
    Compute δ_G = H(SYN(ISN_S), K_DS);
    if δ_G == δ_S then
        Compute ISN_D = R + H(PORT_S, PORT_D, ID_S, ID_D, K_DS);
        SYN(ISN_D) + ACK(ISN_S + 1);
        δ_D = H(SYN(ISN_D) + ACK(ISN_S + 1), K_DS);
        Send SYN(ISN_D) + ACK(ISN_S + 1) + δ_D;
    end
    else
        Drop SYN(ISN_S) segment;
    end
end
if ACK(ISN_D + 1) + δ_S is rcvd then
    Compute δ_G = H(ACK(ISN_D + 1), K_DS);
    if δ_G == δ_S then
        Connection established;
        Allocate resources for S;
    end
    else
        Drop ACK(ISN_D + 1) segment;
    end
end
}
```

shows that both source-destination pairs generate the same session secret key:

$$K_{SD} = e\left(r_S SK_S, R_S PK_D\right) = e\left(SK_S, PK_D\right)^{r_S R_S} = e(sPK_S, PK_D)^{r_S R_S}$$
$$= e\left(PK_S, PK_D\right)^{sr_S R_S} = e\left(R_D PK_S, r_D sPK_D\right) = e\left(R_D PK_S, r_D SK_D\right) = K_{DS}.$$

where $R_S = \sum_{i=1}^{J} r_i = R_D$.

5. Simulation

In order to compare the performances, we have simulated the proposed SA-TCP along with TCP New Reno [2] and *Ad hoc* TCP (ATCP) [8] using NS-2 (version-2.34) simulator on a computer running Linux Cent OS 5. *Throughput* is chosen as the performance comparison metric. We have considered three different simulation scenarios as follows:

Scenario 1—*Dynamic Random Topology*: In this scenario, we adopt *random way-point* mobility model where we vary the speed of the nodes between 0 m/s to 25 m/s. The pause time is set to 0s for acquiring continuous motion of the nodes in the simulations. This scenario is simulated for 50 number of mobile nodes in a network area of size $670\,\text{m} \times 670\,\text{m}$.

Scenario 2—*String Topology*: We vary the hop distance between source and destination to show the effect in a string topology for the protocols under consideration. This scenario is simulated for 11 number of nodes in a static network of area size $1500\,\text{m} \times 1500\,\text{m}$ $1500\,\text{m} \times 1500\,\text{m}$. **Figure 1** shows an example of a string topology with 11 number of nodes.

Scenario 3—*Grid Topology*: **Figure 2** shows an example of a grid topology of size 7×7. Under this scenario, the network load is varied by increasing the number of TCP connections from 1 to 5. This simulation is done for a grid of size 7×7 in a static network.

The following simulation parameters are kept common for the above mentioned scenarios. The radio transm-

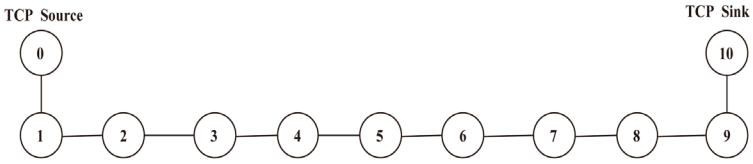


Figure 1. String topology.

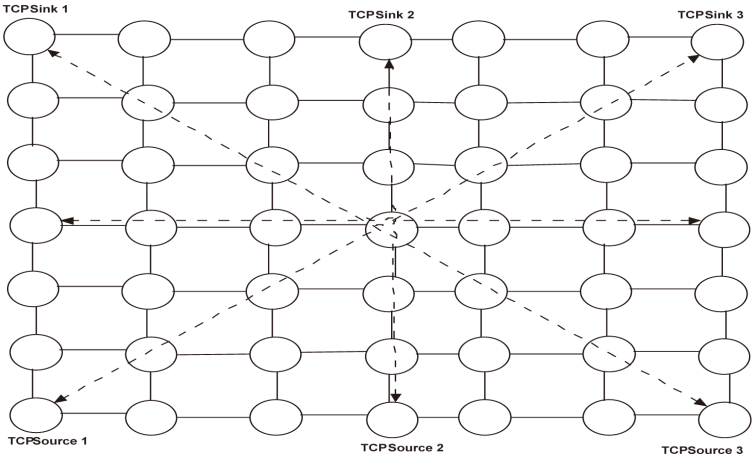


Figure 2. GRID topology.

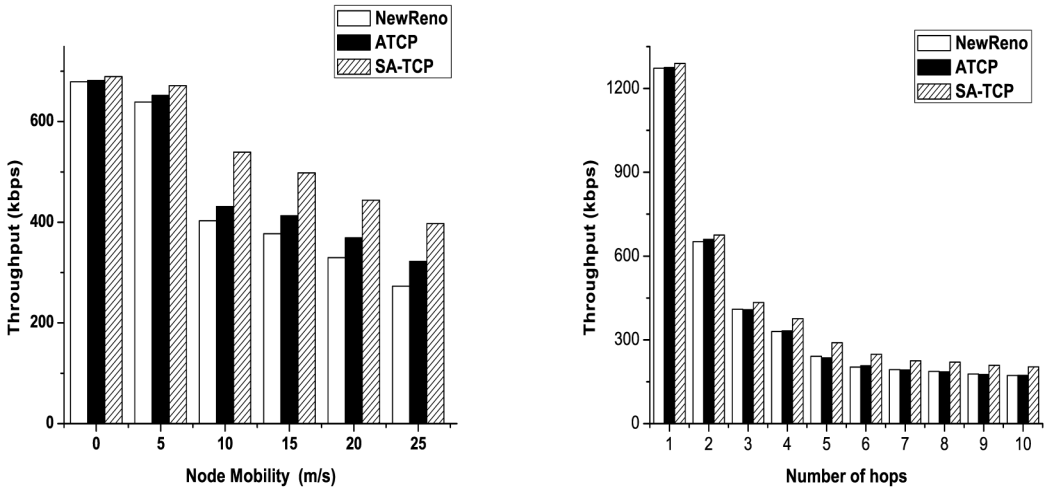
ission range of each node is set to 250 m. *Two-ray ground* propagation model is used and IEEE 802.11 standard has been used as the Medium Access Control layer protocol. *Constant bit rate* (CBR) traffic generator is used of packet size 512 bytes with 4 packets/second traffic generation rate. We have set 1000 s as the total simulation time for each set of simulations.

Simulation Results

Scenario 1—*Dynamic Random Topology*: The impact of node mobility on throughput is shown in **Figure 3(a)**. From the figure we see that the throughput decreases with increased node mobility for all the protocols under consideration. This is due to the increase number of link failures with increased mobility of nodes resulting in segment losses. It can also be seen that the proposed SA-TCP gives better throughput compared to New Reno and ATCP. This is due to the fact that New Reno cannot detect link failures and thereby retransmits duplicate segments unnecessarily. It also mistakes these link failures as congestions in the network and enters into slow start phase. Even though ATCP detects the link failure, the protocol sets CWND to 1 and gradually increments it to the normal size. On the other hand, on detecting a link failure, SA-TCP enters back into the original phase (either congestion avoidance or slow start) as it was at the time of link failure and computes CWND according to the new route condition.

Scenario 2—*String Topology*: **Figure 3(b)** shows the effect of the number of hop distances from source to destination nodes on the network throughput. It can be seen that the throughput decreases with the increased number of hop distances for New Reno, ATCP and the proposed SA-TCP. This is due to the number of MAC layer collisions keeps on increasing with the number of hops. SA-TCP performs better because it sets the mean value of CWND periodically. Further, it tries to reduce the collisions by adjusting the value of CWND according to the network condition and limits the congestion window below a certain value. On the other hand, as no precaution is taken to limit the congestion window by New Reno and ATCP, unnecessary collisions takes place in the network, which in turn triggers retransmissions leading to a domino effect [30].

Scenario 3—*Grid Topology*: **Figure 4** shows throughput of the protocols under consideration in a grid topology. Here, throughput decreases with increased network loads (in terms of number of connections) for all the protocols. As the number of connection increases in the network, the intermediate nodes are also overloaded with number of packets. Therefore, each node on the route has burdens of packet enqueue/dequeue delays



(a) (b)

Figure 3. (a) Throughput vrs. node mobility in dynamic random topology; (b) Throughput vrs. hop distance between source-destination in string topology.

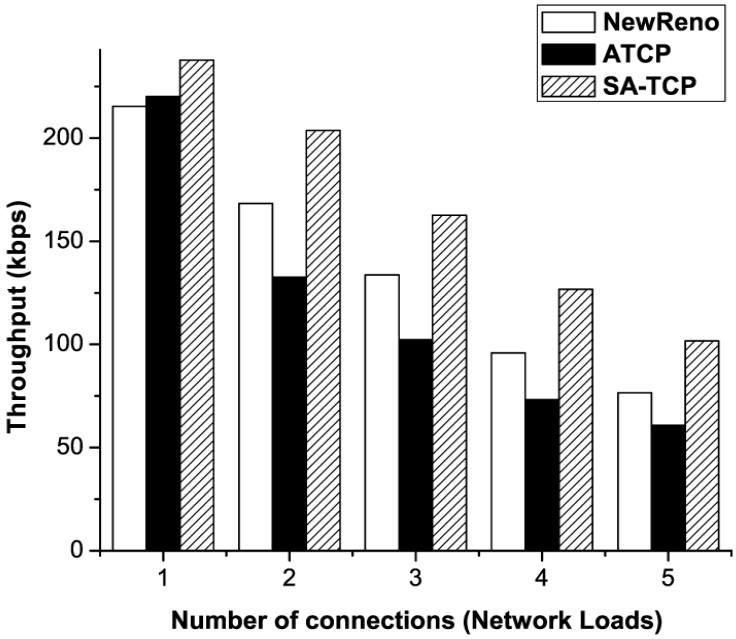


Figure 4. Throughput vrs. network loads in a grid topology.

leading to an overall congestion in the network. SA-TCP adjusts the congestion window efficiently and subsequently it tries to reduce the chance of congestion and MAC layer collisions in the network. The effect is clearly shown in **Figure 4**.

6. Conclusion

This paper proposed a secure and adaptive TCP, called SA-TCP, for wireless ad hoc networks that utilizes network layer information to differentiate different types of packet losses. In order to minimize the chances of congestion and collisions in the network, SA-TCP adjusts the values of *congestion window* (CWND) and *congestion window limit* (CWL) from their previous state values. The proposed protocol uses a low complex identity-based public key cryptography where the public key of a node is evaluated from its ID. Further, all

SA-TCP segments are sent along with MAC tags for the purpose of authentication. Simulation results are given to show that SA-TCP outperforms New Reno and ATCP in standard wireless ad hoc network scenarios.

References

[1] Holland, G. and Vaidya, N. (1999) Analysis of TCP Performance over Mobile AD Hoc Networks. *Proceedings of the* 5*th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Seattle, 15-19 August 1999, 219-230.

[2] Hoe, J.C. (1996) Improving the Start-Up Behavior of a Congestion Control Scheme for TCP. *ACM SIGCOMM Computer Communication Review*, **26**, 270-280. http://dx.doi.org/10.1145/248157.248180

[3] Xu, S. and Saadawi, T. (2002) Performance Evaluation of TCP Algorithms in Multi-Hop Wireless Packet Networks. *Journal of Wireless Communication and Mobile Computing*, **2**, 85-100. http://dx.doi.org/10.1002/wcm.35

[4] Chandran, K., Ragbunathan, S., Venkatesan, S. and Prakash, R. (1998) A Feedback Based Scheme for Improving TCP Performance in Ad-Hoc Wireless Networks. *Proceedings of* 18*th International Conference on Distributed Computing Systems*, 472-479.

[5] Sundaresan, K., Anantharaman, V., Hsieh, H.Y. and Sivakumar, R. (2005) ATP: A Reliable Transport Protocol for Ad Hoc Networks. *IEEE Transactions on Mobile Computing*, **4**, 588-603. http://dx.doi.org/10.1109/TMC.2005.81

[6] Singh, H., Saxena, S. and Singh, S. (2004) Energy Consumption of TCP in ad hoc Networks. *Wireless Network*, **10**, 531-542. http://dx.doi.org/10.1023/B:WINE.0000036456.85213.45

[7] Kopparty, S., Krishnamurthy, S.V., Faloutsos, M. and Tripathi, S.K. (2002) Split TCP for Mobile Ad Hoc Networks. *Proceedings of the IEEE Global Communications Conference* (*GLOBECOM* 2002), Taipei, 17-21 November 2002, 138-142.

[8] Liu, J. and Singh, S. (2001) ATCP: TCP for Mobile ad Hoc Networks. *IEEE JSAC*, **19**, 1300-1315.

[9] Kim, D., Toh, C.K. and Choi, Y. (2000) TCP-BuS: Improving TCP Performance in Wireless Ad Hoc Networks. *Proceedings of IEEE International Conference on Communications* (*ICC*), **3**, 1707-1713.

[10] Wu, B., Chen, J., Wu, J. and Cardei, M. (2007) A Survey of Attacks and Countermeasures in Mobile Ad Hoc Networks. Wireless Network Security. Springer, New York.

[11] Ghosh, U. and Datta, R. (2013) P-TCP: A Prediction Based Secure Transmission Control Protocol for Wireless Ad Hoc Networks. *IETE Journal of Research*, **59**, 364-375. http://dx.doi.org/10.4103/0377-2063.118029

[12] Al Hanbali, A., Altman, E. and Nain, P. (2005) A Survey of TCP over Ad Hoc Networks. *IEEE Communications Surveys Tutorials*, **7**, 22-36. http://dx.doi.org/10.1109/COMST.2005.1610548

[13] Wang, F. and Zhang, Y. (2002) Improving TCP Performance over Mobile Ad Hoc Networks with Out-of-Order Detection and Response. *Proceedings of ACM MOBIHOC*, 217-225.

[14] Dyer, T. and Boppana, R. (2001) A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks. *Proceedings of ACM MOBIHOC*, 56-66.

[15] Leung, K.C. and Li, V.O.K. (2006) Transmission Control Protocol (TCP) in Wireless Networks: Issues, Approaches, and Challenges. *IEEE Communications Surveys Tutorials*, **8**, 64-79. http://dx.doi.org/10.1109/COMST.2006.283822

[16] Zheng, Q., Hong, X., Liu, J. and Tang, L. (2007) A Secure Data Transmission Scheme for Mobile Ad Hoc Networks. *Proceedings of the IEEE Global Telecommunications Conference* (*GLOBECOM*), Washington, 26-30 November 2007, 1006-1010.

[17] Papadimitratos, P. and Haas, Z.J. (2006) Secure Data Communication in Mobile Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications*, **24**, 343-356. http://dx.doi.org/10.1109/JSAC.2005.861392

[18] Ghosh, U. and Datta, R. (2011) Identity Based Secure AODV and TCP for Mobile Ad Hoc Networks. *Proceedings of ACM ACWR*, 339-346.

[19] Diffie, W. and Hellman, M.E. (1976) New Directions in Cryptography. *IEEE Transactions on Information Theory*, **22**, 644-654. http://dx.doi.org/10.1109/TIT.1976.1055638

[20] IP Security Protocol (IPSEC). http://www.ietf.org/html.charters/ipsp-charter.html

[21] Ramakrishnan, K.K., Floyd, S., Black, D. and Ramakrishnan, G.K. (2001) The Addition of Explicit Congestion Notification (ECN) to IP.

[22] de Oliveira, R. and Braun, T. (2002) TCP in Wireless Mobile Ad Hoc Networks. Technical Report.

[23] Kaufman, C., Perlman, R. and Speciner, M. (2002) Network Security Private Communication in a Public World. Prentice Hall PTR, Upper Saddle River.

[24] Ghosh, U. and Datta, R. (2013) SDRP: A Secure and Dynamic Routing Protocol for Mobile Ad Hoc Networks. *IET*

Networks.

[25] Ghosh, U. and Datta, R. (2012) A Novel Signature Scheme to Secure Distributed Dynamic Address Configuration Protocol in Mobile Ad Hoc Networks. *IEEE WCNC*, 2700-2705.

[26] Ghosh, U. and Datta, R. (2011) A Secure Dynamic IP Configuration Scheme for Mobile Ad Hoc Networks. *Ad Hoc Networks*, **9**, 1327-1342. http://dx.doi.org/10.1016/j.adhoc.2011.02.008

[27] Sakai, R., Ohgishi, K. and Kasahara, M. (2000) Cryptosystems Based on Pairings. *The* 2000 *Symposium on Cryptography and Information Security*, 26-28.

[28] Alnumay, W.S., Ghosh, U. and Chatterjee, P. (2014) SA-TCP: A Secure and Adaptive TCP for Wireless Ad Hoc Networks. *Frontier and Innovation in Future Computing and Communications*, 1-9.

[29] Stoica, P., Friedlander, B. and Söderstrom, T. (1986) Least-Squares, Yule-Walker, and Overdetermined Yule-Walker Estimation of AR Parameters: A Monte Carlo Analysis of Finite-Sample Properties. *International Journal of Control*, **43**, 13-27. http://dx.doi.org/10.1080/00207178608933446

[30] Chen, K., Xue, Y. and Nahrstedt, K. (2003) On Setting TCP's Congestion Window Limit in Mobile Ad Hoc Networks. *Proceedings of IEEE International Conference on Communications*, Alaska.

Scientific Research Publishing (SCIRP) is one of the largest Open Access journal publishers. It is currently publishing more than 200 open access, online, peer-reviewed journals covering a wide range of academic disciplines. SCIRP serves the worldwide academic communities and contributes to the progress and application of science with its publication.

Other selected journals from SCIRP are listed as below. Submit your manuscript to us via either submit@scirp.org or Online Submission Portal.