Denial in DTNs

Md Yusuf Sarwar Uddin
mduddin2@illinois.edu

Ahmed Khurshid
khurshi1@illinois.edu

Hee Dong Jung
jung42@illinois.edu

Carl Gunter
cgunter@illinois.edu

Abstract

Disruption-tolerant network (DTN) is an intermittently connected network where the traditional end-to-end data communication between a source-destination pair is hardly possible. Instead, nodes opportunistically replicate the same packet several times and try to deliver them through multiple paths. Due to this multi-copy forwarding scheme, denial of service (DoS) attacks in DTN is believed to be a non-trivial task. Unlike classical DoS attacks for the Internet where an attacker renders some service at a particular node unavailable for legitimate users, in DTN, attackers do not have any node-level attack objective. They rather seek to diminish some network-wide global performance (e.g., total packet delivery). In this report, we describe a few possible DoS attacks for DTNs and propose a set of handful countermeasures against those attacks under the assumptions of our threat model. We show that in the presence of adversaries network performance declines and our protection mechanisms attempt to limit the overall effect to a considerable amount. We simulate our scheme in ns-2 and show that significant trade-off exists in choosing protection mechanisms commensurately with attack models and resource utilization.

1 Introduction

Disruption-tolerant networks (DTNs) [3] experience intermittent connectivity due to various kinds of disruptions such as limited transmission range of sparsely located nodes, mobility of nodes, limited energy resources, attack, and noise. Providing communication in DTNs, therefore, is more challenging than in traditional wireless networks where continuous connectivity generally prevails. In DTNs, links are frequently disrupted and not always available to construct end-to-end paths between pairs of nodes. Consequently, traditional single-copy routing protocols cannot be applied in DTNs. Most routing protocols for DTNs allow multiple copies of a given packet to be replicated and forwarded to various nodes. The goal is to increase the possibility to deliver the packet to the ultimate destination.

Denial of Service (DoS) attack is one of the most critical threats to fixed infrastructure networks, such as the Internet. These DoS attack models, however, cannot be directly applied to DTNs due to their aforementioned unique characteristics. It does not imply that there could not be any DoS attacks possible in DTNs. The characteristics of DTNs introduce new vulnerabilities to the network as well. Attacks that are common to DTNs are dropping of packets, flooding the network with unnecessary spurious packets, spoofing a different node's address to intercept all the packets destined to that node, corrupting routing states and counterfeiting network acknowledgments. Although DTNs have not been widely deployed today, it is important to speculate what kind of attacks are possible and what could be the countermeasures to those attacks. This is due to the reason that once it has been deployed largely, it is extremely hard to retrofit the network infrastructure and protocols to protect and prevent it from the attacks.

In this report, we design and implement two different DoS attack models in a DTN along with countermeasures. We believe that the amount of suffering due to attacks depends on the routing algorithms used by DTN nodes. And since there is no de facto routing protocol for DTNs, we

choose Spray-and-Wait [13] protocol as our base protocol. We prefer this protocol because it is simple and requires less resources. In our attack models, malicious nodes drop all the packets they receive instead of forwarding and could also spoof other nodes' address to absorb all the packets intended for those nodes. We introduce two types of countermeasures such as opportunistically following Spray-and-Wait protocol, and using tokens to loosely identify peers.

We extend the ns-2 [5] simulator to build specialized DTN agents and protocols as ns-2 currently does not support DTN environments. The simulation results show that packet delivery rate decreases significantly in the presence of malicious nodes, i.e., packet droppers and/or address spoofers. The results also show that delivery rate is increased with our countermeasures. In addition to that, we also measure the overheads caused by the countermeasures in terms of number of copies of a single packet.

The rest of this report is organized as follows. In Section 2, we describe some existing works on DTN security. We present a few well-known DTN routing protocols in Section 3. In Section 4 and Section 5, we describe two DoS attack models and our proposed protection mechanism. We describe our simulation environments, evaluate our protection schemes and analyze the simulation results in Section 6. Finally we conclude with pointers to some future works in Section 8.

2 Related Works

Our work is related to three areas of prior works: DTN routing schemes, DTN security, and DoS attacks on DTN.

There are several DTN routing schemes proposed in the literature. Four major ones could be Epidemic routing [15], PROPHET [10], MaxProp [7], and Spray-and-Wait [13]. Epidemic routing simply makes multiple copies of packets to flood the network in a hope that any one of them will be delivered to the destination. This protocol performs best in terms of packet delivery and latency when network bandwidth and storage are unlimited. But it is not the case in practice. PROPHET estimates delivery predictability to destinations using the history of encounters. MaxProp computes a rank for each packet in terms of delivery probability and sorts packets in the transfer buffer accordingly. Upon transfer opportunity, packets are replicated in the order of their ranks. Spray-and-Wait follows a flooding scheme, but limits the total number of copies per packet.

There have been a few research works on DTN security. Work by Seth et al. [12] has shown that traditional mechanisms including a combination of Public Key Infrastructure (PKI) certificates issued by trusted third parties and Certificate Revocation Lists (CRLs) are not suitable for DTNs. They develop a cryptosystem for DTNs using Hierarchical Identity-Based Cryptography (HIBC) for creating secure channels, providing mutual authentication, and key revocation. They, however, do not discuss how the disconnectedness and opportunistic nature of communication can be exploited by malicious agents to disrupt packet flow between legitimate nodes and do not provide any simulation or experimental results of their architecture. There is also Bundle Security Protocol Specification [14] that tries to supplement the Bundle Protocol (BP) with security aspects. This RFC draft addresses how the key distribution and management should be done in DTNs. Other approaches that try to address security issues in DTNs include [8, 9]. However, they all tend to address security issues like confidentiality and integrity rather than availability.

Burgess et al. [6] show that DTNs are more robust to adversaries than usual connected networks. They compare single-copy methods with multi-copy methods and prove that multi-copy methods give higher delivery rate in the presence of malicious nodes. They compare four different routing algorithms (MaxProp and its three variants) against four different attack models: dropping all packets, flooding of packets, routing table falsification and counterfeiting delivery acknowledgments. They also show that performance of DTNs degrades merely by 15% even in the presence of 30% compromised nodes for a particular set of attack models. However, we show that the performance of DTNs can decrease significantly under different protocols and attacks.

3 DTN Routing Protocols

There are a couple of DTN models available to date, namely IPN (InterPlaNetary Internet) [2], DakNet [11], DieselNet [1] and so on. Here we consider a vehicle-based DTN where a set of nodes move in a large area and disruption is mainly induced by mobility and short range of radio communication. A particular instance can be a network of public buses in a city (DieselNet), or an emergency response network activated after a large scale disaster. In this network, nodes sparsely move and occasionally meet (come closer within the radio range of) other nodes. DTN routing protocols exploit this mobility to carry packets and opportunistically transfer them when nodes get transfer chance. Protocols are essentially of multi-copy kind in the sense that the same packet is replicated several times. This makes several nodes to retain the same packet (preferably for a long time) so that anyone of them is able to deliver the packet to the destination.

In DTN, all packet transfer decisions are made upon a contact (meeting of a pair of nodes). Based on the identity of the peer node, packet transfer event can be of two different types: delivery and replication.

- **Delivery:** Packet delivery happens when a node encounters another node for which it holds a set of packets destined to that node. On delivery, packets are transferred to the destination and are usually dropped from the sender buffer.
- **Replication:** Replication happens when a node transfers a set of packets to its peer node that is not the destination of those packets. The peer acts as an intermediary node en route to the destination. When replicated, packets reside at the sender, but some state of those packets in sender side may be changed based on the routing protocol used.

How to attack a network is mainly dependent on what kind of routing scheme the network uses. The protocols that compute local states by sharing information with other peers and make replication/delivery decisions based on that are prone to attack. Adversary nodes can arbitrarily tamper their own states or can pass intentionally modified information to peers to corrupt the local states of honest nodes. Therefore, it is obvious that the routing protocol that maintains routing table like in-node states can be subject to severe DoS attacks. Spray-and-Wait protocol is a stateless protocol in that nodes do not maintain any routing states, instead a tiny state is kept in each packet header. This is why we prefer Spray-and-Wait as our base protocol (we call it Spray hereafter). Our description of attacks and countermeasures are specially tailored to this. In the following, we describe this protocol in more detail.

Spray puts limit on the number of replicas (copies) per packet. The protocol runs in two phases. In the first phase, each packet is replicated to several nodes (spray phase), and in the second phase, those nodes hold the packet until they encounter the destination node and deliver it (wait phase). A special field (replica count) in the packet header determines how many times a packet can be replicated. When a node meets another node, it picks a packet (with replica count k) from its buffer, retains half of the copies ($\lceil \frac{k}{2} \rceil$ copies) and replicates the other half ($\lfloor \frac{k}{2} \rfloor$) to the peer. Of course, the packet is not sent repeatedly, rather just once, but the replica count field is adjusted accordingly. This keep-half-give-half continues until the replica count of that packet becomes 1. In that case, the packet can no longer be replicated but can only be delivered. Figure 1 demonstrates the operation of Spray.

4 Attack/Threat Models

In our work, we don't look into confidentiality or integrity of network services, rather their availability. Key-based encryption mechanisms and cryptographic signature which are de facto standards for preserving data confidentiality and integrity do hardly provide any protection against DoS attacks. Again, because of pure ad hoc deployment, distributing keys and having a centralized certification

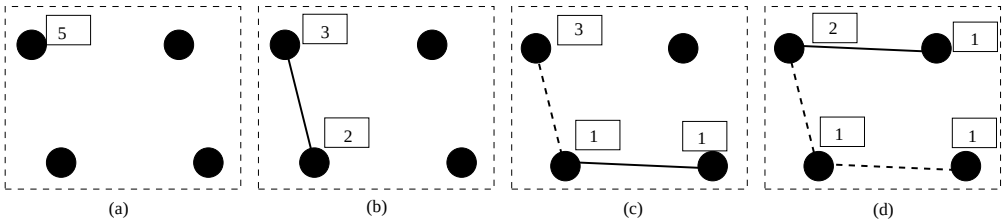


Figure 1: Spray-and-Wait. Active connection is shown in solid line. The box indicates a packet and the number inside is the replica count.

authority is not feasible for DTN. Given this, there is no way to establish trust among nodes, and identification fraud could be one of the most potential attacks in this type of network. Moreover, any node can be compromised with all of its credentials and can participate in protocol operation without any least of suspect from other nodes.

In our attack model, the prime purpose of malicious nodes performing DoS attacks is to reduce the amount of packets delivered to their intended destinations. To achieve this purpose, adversaries need some strategies so that packets are not able to traverse in the network through honest nodes. In our project we have implemented two types of attacks in a DTN. They are packet dropping and address spoofing.

4.1 Packet Dropping

In this attack, a compromised node (adversary) does not replicate packets that it receives from other nodes. This type of adversaries acts as black-holes in the network and hence affects packet propagation within the network. However, packet dropping does not drain out all the copies of a particular packet as the Spray tries to send multiple copies of each packet through different nodes. If one of these packets is able to reach the intended destination then this type of attack does not create a major problem in the network.

4.2 Address Spoofing

In the second form of attack, an adversary pretends to be someone else when it meets another node (honest or adversary) in the network. It performs this by spoofing someone else's address. So, an honest node sends the packets destined to this spoofed address to its peer and removes those from its buffer after delivery. If the sender was not able to replicate some of these packets earlier, then they are simply lost and cannot be delivered to the actual destination. In fact, if a spoofer is able to consume packets having a high replica count, then successful delivery of those packets becomes less likely. So, spoofing creates more havoc in the network with respect to packet delivery than dropping. However, an adversary can perform both types of attacks simultaneously.

5 Protection Mechanisms

One possible protection against attacks committed by adversaries is to identify and avoid them. As mentioned earlier, lack of infrastructure and intermittent connectivity prevent the use of PKI based techniques to filter out adversary nodes. So, in order to ensure a certain level of service guarantee (e.g., moderate packet delivery rate), honest nodes need to rely on local information to decide what to do when a connection is established with a peer node. Under this situation, honest nodes have very limited choices to counteract DoS attacks. Basically, an honest node has two obvious choices; one is not to remove packets from its buffer upon delivery and the other is to create more packet replicas. Apparently, these two schemes are able to improve packet delivery

rate. However, the challenge is to adopt these two schemes without overburdening the networks with too many replicas. This is the fundamental insight of our proposed protection mechanisms.

We propose two protection mechanisms against DoS attacks in a DTN. They are opportunistic protection (OP) and token-based protection (TP).

5.1 Opportunistic Protection (OP)

Opportunistic protection (OP) is a naïve protection against both dropping and spoofing attacks. In this strategy, the sender follows the Spray protocol but opportunistically retains *all* copies of the packets that it sends to its peer. The sender does this by tossing an unbiased coin. So, after sending packets to the peer, the sender retains the copies (i.e. does not decrease the replica count or does not remove packets from the buffer after successful delivery) with a fixed probability (we used 0.5). This retention leads to further opportunity for these packets to be replicated and eventually reach the destination. However, if the receiving node is an honest one, then this strategy creates additional copies of the packets beyond the default replica count maintained by the Spray. In Section 6, we present the level of protection that OP provides against dropping and spoofing attacks along with associated overhead.

5.2 Token-based Protection (TP)

Opportunistic protection blindly replicates packets without knowing whether the network has any adversary or not. We present a protection strategy where nodes try to assess the presence of adversary in the network. The primary intention is to detect whether anyone is spoofing addresses. This is done as follows. Every node generates a unique secret bit string (token) for each peer node it encounters and nodes exchange these tokens when they first meet. In subsequent meetings between the same pair of nodes, each of them does the following. They reproduces the tokens that they received from each other. The receiver then checks whether the token reproduced by its peer is the same token that it generated and supplied to this peer in their first meeting. In the absence of adversaries, a node should experience at most one token mismatch for each peer address (during their first meeting). When a node experiences more than one token mismatch (collision) for a particular address, it can suspect that the particular address has been claimed by several nodes, hence spoofed. Each node counts the number of collisions per address and uses those counts to realize the protection strategy. The following subsections describe this protection strategy in detail.

5.2.1 Protection using Collision Count

As nodes count collisions per address, if the collision count for a particular peer address is greater than 1, the node expects that there can be at most one honest node carrying the real address and others being adversaries spoofing the same address. Let p_h be the probability that a node claiming a particular address is honest. We have $p_h = 1/\text{collision count}$. A low value of p_h indicates the presence of a good number of spoofers and hence should incite more replicas to put into the network. Each node computes p_h upon meeting a peer and makes its delivery/replication decision using p_h.

In case of delivery, the node drops the packet from the buffer with probability p_h (peer is assumed to be honest), otherwise replica count is raised to 2 if this is the last copy. This is done just to let the packet to cross one more hop in case it has been blocked by an adversary. Use of p_h during replication needs more explanation. Let the current replica count of a packet be k. Recall that during replication, an honest node keeps $\lceil \frac{k}{2} \rceil$ copies of a packet and the peer gets $\lfloor \frac{k}{2} \rfloor$ copies. So, in the absence of any adversary, there can be at most k copies of that packet. However, if the peer node is an adversary that drops packets then $\lfloor \frac{k}{2} \rfloor$ copies of the packet will not get any chance to remain in the network. So, honest nodes need to boost the number of copies in order to maintain the 'k copy' invariant. The amount of boosting should depend on p_h and needs to be controlled so

that excessive copies do not overburden the network. TP tries to restrict the expected number of copies per packet to k.

Let an honest node raise the replica count of a packet P by a factor α before sending the packet to its peer and K be the total number of replicas of P. Assuming that an adversary drops all the packets, the expected number of replicas of a packet in the network after this replication will be:

$$E[K] = p_h.\alpha k + (1 - p_h).\frac{\alpha k}{2}$$

If we want to keep the expected number of copies to k, then:

$$\begin{aligned} E[K] &= k \\ p_h.\alpha k + (1 - p_h).\frac{\alpha k}{2} &= k \end{aligned}$$

Simplifying the above equation we get:

$$\alpha = \frac{2}{1 + p_h}$$

So, while replicating each packet, the corresponding replica count is multiplied by $\frac{2}{1+p_h}$, that eventually should generate k copy of packet P in the network on the average. For base check, if there is no adversary, i.e. $p_h = 1$, then, $\alpha = 1$ and this scheme works exactly as the basic Spray. We augment the basic Spray to incorporate TP in it, as shown in Table 1.

We argue that the protection provied by TP is originated from the following two observations:

- An adversary cannot produce the token that an honest node generates for a particular peer, if it has not received that token from that honest node in the first place.
- An adversary cannot avoid collision at the honest node when it spoofs.

During packet delivery	**During packet replication**
Send a packet P	
Find collision count for the peer and compute p_h	Find collision count for the peer and compute p_h
After successful transfer	$\alpha \leftarrow 2/(1 + p_h)$
Generate a random value $r_v \sim$ uniform$(0, 1)$	P.replica-count $\leftarrow \lceil \alpha \times P$.replica-count$\rceil$
if not the first meeting and $(r_v < p_h)$	After successful transfer
Remove P from buffer	P.replica-count $\leftarrow P$.replica-count / 2
else	
P.replica-count $\leftarrow$ max(2, P.replica-count)	
endif	

Table 1: Algorithm for token-based protection (TP)

5.2.2 Token Management

Each node maintains a list of tokens that it supplies to its peer upon their first meeting. These tokens can simply be generated using a hash function (e.g., SHA-1) on a per node secret key and the peer address. All nodes also maintain a peer token table (PTT) where they store the tokens received from their peers, indexed by peer address. Initially all the entries in PTT are empty. When two nodes come in contact, they first learn each other's address by exchanging HELLO messages. Then they extract the corresponding peer token from PTT (if available) and send it in the HELLO-REPLY message. After receiving the HELLO-REPLY message, a node matches the

reproduced token with the token it expects from the peer (in case they met earlier). If the tokens do not match, it records this event as a new collision and sends a SET-TOKEN message to its peer supplying the token it generated for this particular peer address. The peer is expected to reproduce this token in subsequent meetings.

5.2.3 Limitations

TP suffers from a few limitations. If p_h is 1, TP does not boost the number of copies. However, it may happen that an honest node always meets the same adversary spoofing the same address and that honest node never meets with the actual address holder or any other adversary spoofing that address. In that case, the honest node does not suspect this peer to be an adversary and always follows basic Spray as p_h remains 1. But this case is less likely due to the random connectivity pattern present in a DTN.

Moreover, TP only works when adversary nodes spoof addresses. It fails to provide any protection when adversaries do not spoof but simply drop packets. Here, we are assuming that an adversary node advertises its own address during meeting, and always announces successful reception of packets without replicating the packets it receives. So, it is impossible for the sender to determine whether the peer node is dropping packets or not. In this case, collision counts never exceed 1 and basic Spray is followed all the time. We evaluate the effect of this situation in Section 6.

5.3 Stateless vs. Stateful Attacks

Since collision count brings forth more replicas into the network, why does not an adversary remember the token it receives from a peer and spoofs the same address when it meets that particular node? An adversary may avoid further address collisions by this, and the collision count at the (honest) peer may not increase. This generates less replicas and delivery rate is affected. This type of attack is called *stateful* (SF). This attack requires an adversary to remember the token it obtains from a node and the associated spoofed address it should be using for that node. Another variant of (spoofing) attack that just chooses a random address to spoof can be regarded as *stateless* (SL).

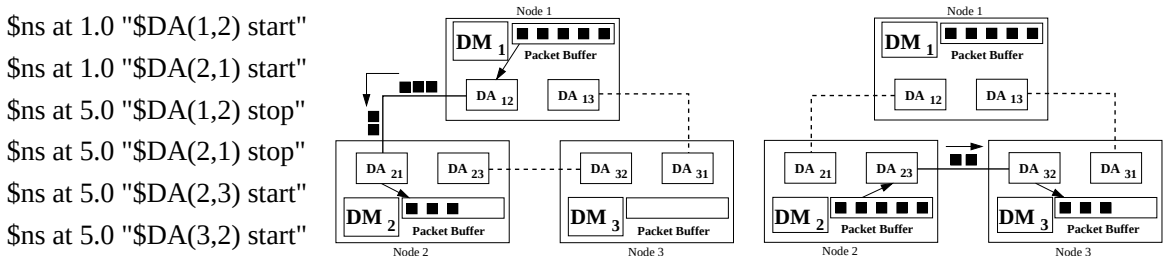
6 Evaluation

In order to test the effects of different types of DoS attacks in a DTN and our proposed protections, we use ns-2 as the simulation platform. In this section, we describe our simulation setup and explain the simulation results.

6.1 Simulating DTN in ns-2

Ns-2 currently does not provide any facility for simulating a DTN. So, we had to write specialized classes to incorporate DTN-like functionalities in ns-2. We are interested in two aspects of simulation; disrupted connection among different nodes and a node's ability to transfer packets to any other node in the network. Currently ns-2 requires all communicating agents (say, Agent/TCP) to be bound (statically) to each other and to their respective nodes before running the simulation. In our implementation, every DTN node creates a separate agent for every other node in the network and all these agents are statically bound pair-wise beforehand. When it is time for two nodes to communicate, the associated agent pair is simultaneously activated. A pre-generated connection pattern file containing activation/deactivation schedule is provided to simulate disruption. Details of two newly added classes are presented below.

DTNAgent This class derives from the **Agent** class present in ns-2. So, links can be added between two DTNAgents and packets can be sent between them. Currently we implement DTNAgent



(a) Schedule (b) At time: 1.0 sec (c) At time: 5.0 sec

Figure 2: Illustrating DTN activity

as a constant bit rate packet generator without any congestion control. This design decision does not affect the results as links in a DTN does not typically face any congestion. However, wireless nodes do face contention while accessing the shared channel, but that is different from congestion. Upon activation, a DTNAgent contacts its DTNManager to learn its address and exchanges HELLO, HELLO-REPLY and SET-TOKEN messages (if required) with its peer DTNAgent. Then the DTNAgent fetches packets from its DTNManager (explained later) and sends those to its peer at a pre-configured rate. It stops its transaction for the ongoing session once it is deactivated according to the schedule.

DTNManager Every node has a DTNManager that runs the Spray protocol, manages the packet buffer and employs the protection mechanism. It holds the address of its node and passes this address to its DTNAgents when they communicate with peers. It also manages the packet buffer that is shared by all DTNAgents in the same node. It is this entity that changes its behavior while in attacking mode. It has a set of flags that control its behavior in the network. The flags are – adversary mode {none, drop packets, spoof address, both drop and spoof}, attack method {stateless, stateful} and protection {none, opportunistic, token-based}.

Based on the adversary mode and attack method, a DTNManager drops packets or supplies false addresses to the DTNAgents during connection setup time. Figure 2 illustrates the DTN architecture and the agent activation/deactivation schedule based on a simple connection pattern file in a three-node network.

6.2 Simulation Parameters

We have used the ONE (Opportunistic Network Environment) simulator [4] to generate the connection pattern file used in our simulations. The connection pattern file contains a list of timestamped `ns` commands to activate/deactivate pair of agents when the corresponding pair of nodes 'meets' (Figure 2(a)). We use a city mobility model (comes with ONE, Figure 3) with three moving entities; pedestrians (random movement within a confined area), cars (random movement in the entire area) and trams (moving in 3 fixed cyclic routes). These moving entities are equipped with radios and act as DTN nodes. They run the Spray protocol with the initial replica count 10 and use the parameters as shown in Table 2.

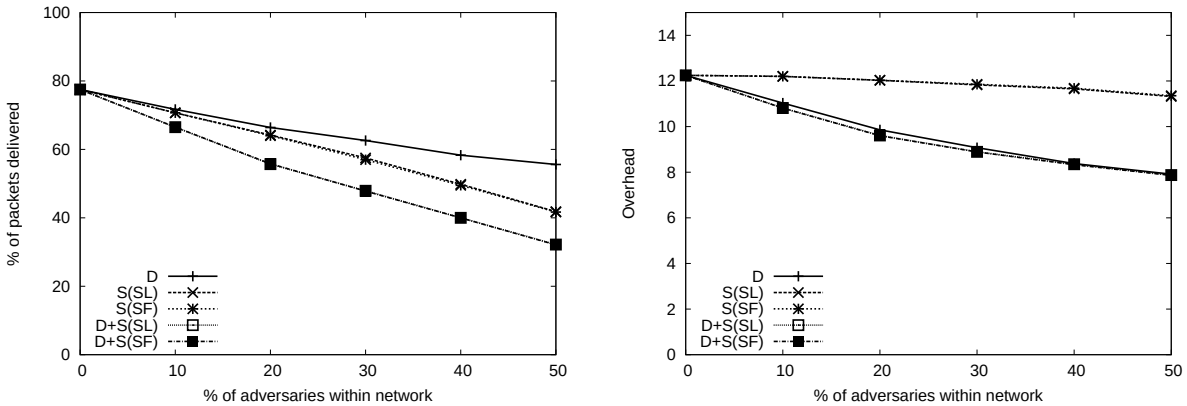
Traffic type	**Wait-time (s)**	**Tx range (m)**	**Speed (m/s)**	**# of nodes**
Pedestrian	0 – 120	50	0.5 – 1.5	20
Car	0 – 120	50	2.7 – 13.9	20
Tram	10 – 30	50	7.0 – 10.0	10 (4+3+3)
			Total nodes	50

Table 2: Simulation parameters

Figure 3: Simulation scenario

7 Simulation Results

In this section, we present the simulation results of the DoS attacks and the performance of our proposed countermeasures. We are interested in two metrics of the network: delivery rate and overhead. Delivery rate is the fraction of uniquely generated packets that has been successfully delivered to the corresponding destinations. On the other hand, overhead is computed as the total number of replicas of all packets divided by the total number of unique packets created, i.e., average number of replicas per packet. We use packet created instead of packet delivered to pull apart delivery rate from the overhead metric. In most cases, we show these two metrics against the percentage of adversary nodes in the network.

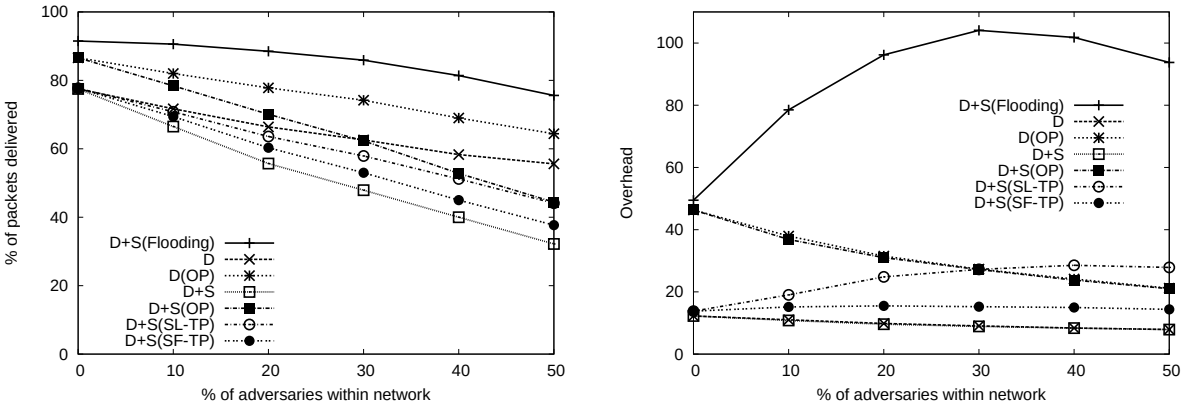


(a) Delivery rate (b) Overhead

Figure 4: Effect of DoS attacks (dropping (D), spoofing (S), and both (D+S))

In Figure 4, we show the effects of adversary nodes on delivery rate and overhead when no protection mechanism is adopted. As we see in Figure 4(a), delivery rate declines as the number of adversary increases in the network. It also reveals that spoofing attack produces more adverse effects than dropping. This is due to the fact that a spoofer, by claiming a false identity, grabs

packets destined for another node. Once (falsely) delivered, no more packets are replicated for those delivered packets and the actual destination fails to receive them. On the contrary, overhead does not depend much on the number of adversaries when the attack is only 'spoofing' (Figure 4(b)), but the overhead reduces when adversaries drop packets. Because no replica is created for those dropped packets. Spoofing with dropping has slightly less overhead than only spoofing. When attackers drop packets with spoofed addresses, packets are both dropped and falsely delivered. This results in less replica compared to just dropping without spoofing. Moreover, without any protection, stateless and stateful (spoof) attacks do not have much effect on either delivery or overhead.



(a) Delivery rate (b) Overhead

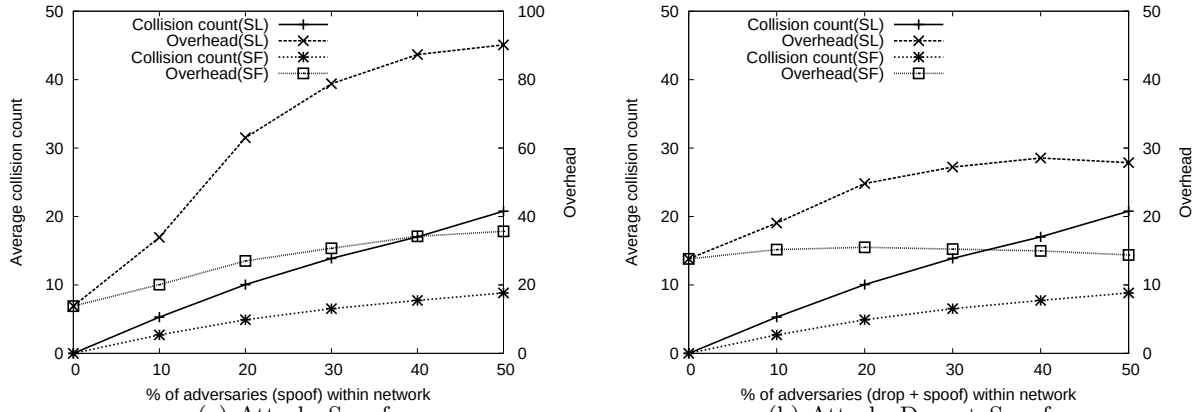
Figure 5: Protection mechanisms

Figure 5 shows the results when countermeasures are applied to the attacks. As we mentioned earlier, we have two protection mechanisms, namely opportunistic and token-based. As we see in the figure, when opportunistic protection (OP) is applied to 'drop' attack, a somewhat persistent improvement is achieved in packet delivery rate (with a declining overhead) as the number of adversary nodes rises. This is for the reason that even though adversaries drop packets, OP aggressively replicates packets with some (fixed) probability. The same trend is observed when spoofing is added with dropping, but with smaller delivery rate and almost the same overhead.

For token-based protection (TP) we see that stateless (SL) (spoof) attack results in more delivery along with increased overhead (since, more and more replicas are created due to address collision). On the other hand, in case of stateful (SF) attack, delivery rate is lower than the stateless one, i.e., SL is better attack than SF. However, SF creates less overhead compared to SL. Adversaries make greater damage to the network in terms of delivery by becoming a stateful spoofer; but this allows protection to be enforced with less expense.

When attackers become stateless, protection becomes expensive (overhead is high). What about just flooding packets instead of limited spraying? To compare this, we run a flooding scheme, where a node replicates all packets in its buffer to its peer ignoring replica count whatsoever. We observe that flooding offers persistently higher delivery rate ($> 80\%$) (Figure 5) irrespective of number of nodes being adversary. But flooding achieves this at a very high price of transmission energy due to massive replication. Depending on the number of adversary nodes, flooding creates $60 - 100$ replicas per packet that is too costly for an energy-constrainted DTN and is hardly affordable. Therefore, even though TP in the presense of stateless attack gives rise to high overhead, this overhead is far less compared to pure flooding.

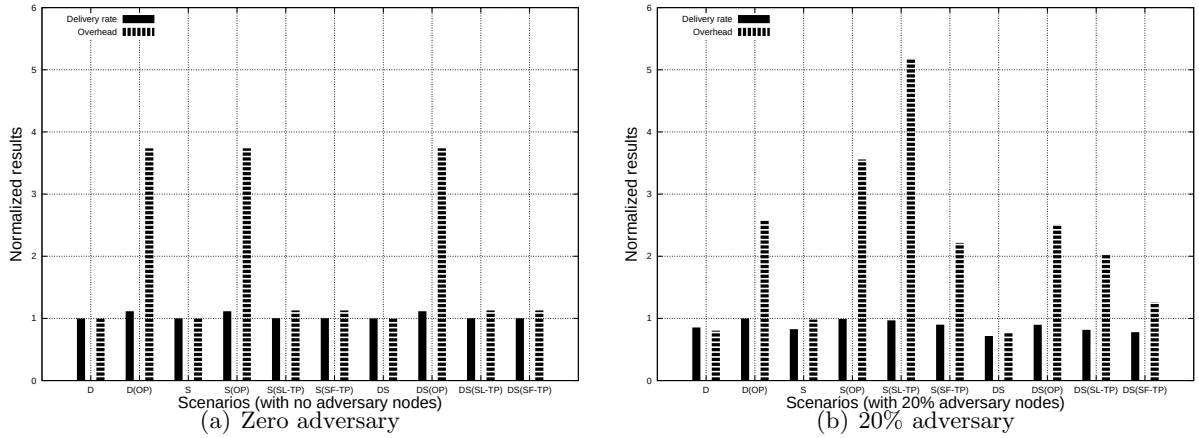
In Figure 6, we analyze the token-based protection (TP) mechanism more closely. As we described earlier, TP keeps track of address collision and replicates packets based on the collision count. Figure 6 depicts the relationship between the average collision count per node and the overhead exerted in the network thereby. We see that as the number of adversary nodes increases,



(a) Attack: Spoof (b) Attack: Drop + Spoof

Figure 6: Effect of collisions on overhead

nodes experience more collisions that leads to higher overhead. Obviously, stateless attack produces more collisions than stateful one, and this is why stateless attack yields more replicas into the network. However, in case of dropping attack, overhead gets down (Figure 6(b)). Although replica count is very high for stateless (spoof only) attack (Figure 6(a)), honest nodes as well as adversaries bear this overhead.



(a) Zero adversary (b) 20% adversary

Figure 7: Normalized results

We show a comprehensive network-wide performance in various cases of attack models and protection mechanisms in Figure 7 for zero and 20% adversary nodes. All metric values are normalized against the results obtained in case of zero adversary and no protection in action. We find from Figure 7(b) that opportunistic protection (OP) offers the best delivery irrespective of attack types (dropping, spoofing or both), but at the expense of huge overhead. Whereas, token-based protection (TP) provides a good trade-off between delivery rate and overhead. While attackers mostly intend to cause delivery to suffer, sometimes they might be interested in immense flooding of packets. In that case, attackers switch their purpose of attack to exhaust resources by replica brust instead of affecting delivery rate. Given that adversaries know what protection scheme the network uses, Table 3) shows what type of attack an adversary node may choose to achieve a particular purpose.

Figure 7(a) presents what happens to delivery rate and overhead when protection is in action, but there is no adversary. We see that if TP is applied, both delivery rate and overhead remain fairly unchanged. Therefore, TP can be regarded as non-intrusive in the sense that in the absence of adversaries it does not introduce any extra overhead to the existing available services. But this

Protection	Attack choice	Purpose of attack
NONE	Drop + spoof	Delivery deterioration
OP	Spoof	Resource exhaustion
TP	Stateless spoof	Resource exhaustion
TP	Stateful spoof + drop	Delivery deterioration

Table 3: Choice of attacks based on known protection mechanisms

is not true for opportunistic approach. It constantly warrants extra resource overhead even if there is no adversary. Considering all these, we can conclude that stateful spoofing with dropping is the 'best' attacking strategy for adversaries, whereas token-based protection is the most effective protection mechanism.

8 Conclusion and Future Works

In this paper, we propose two forms of DoS attacks for an energy constrainted DTN. It has been shown that these two seemingly simple attacks significantly knock down the delivery rate to a low value. The countermeasures we devised raise the delivery rate to a considerable amount with a slight increase of transmission resources.

In future, we may investigate the possibility of having a few but uncompromisable trusted nodes in the network that are capable of generating certificates for other nodes. Possible work could be to look deep into this to understand whether having some structure instead of a purely non-structured network could help in attaining good measures against DoS attacks. Devising countermeasures for other routing protocols is also another way of extending the current work. It may be interesting to study the effects of colliding attackers who can coordinate and pass information among themselves in the form of some botnets. We also intend to run our current experiments on real traces. A few good traces are publicly available at `http://crawdad.cs.dartmouth.edu/`.

References

[1] DieselNet. http://prisms.cs.umass.edu/dome/umassdieselnet.

[2] InterPlaNetary Internet. http://www.ipnsig.org/.

[3] IRTF DTN Research Group. http://www.dtnrg.org.

[4] Opportunistic Network Environment (ONE) Simulator. http://www.netlab.tkk.fi/jo/dtn/.

[5] The Network Simulator – ns-2. http://www.isi.edu/nsnam/ns/.

[6] J. Burgess, G. D. Bissias, M. Corner, and B. N. Levine. Surviving attacks on disruption-tolerant networks without authentication. In *MobiHoc '07*, pages 61–70, New York, NY, USA, 2007. ACM.

[7] J. Burgess, B. Gallagher, D. Jensen, and B. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *INFOCOM '06*, pages 1–11, April 2006.

[8] S. Farrell and V. Cahill. *Delay- and Disruption-Tolerant Networking*. Artech House, Inc., 2006.

[9] A. Kate, G. Zaverucha, and U. Hengartner. Anonymity and security in delay tolerant networks. In *SecureComm 2007 Workshop on Security and Privacy in Comm. Networks*, pages 504–513, 2007.

[10] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks. *Mobile Computing and Communications Review*, 7(3):19–20, July 2003.

[11] A. Pentland, R. Fletcher, and A. Hasson. Daknet: rethinking connectivity in developing nations. *Computer*, 37(1):78–83, 2004.

[12] A. Seth and S. Keshav. Practical security for disconnected nodes. *IEEE Workshop on Secure Network Protocols*, 0:31–36, 2005.

[13] T. Spyropoulos, K. Psounis, and C. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *SIGCOMM workshop WDTN '05*, pages 252–259, 2005.

[14] S. Symington, S. Farrell, and H. Weiss. Bundle security protocol specification, May 2006.

[15] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-2000-06, Department of Computer Science, Duke University, April 2000.