IJERT
ISSN : 2278-0181
International Journal of Engineering Research & Technology
Publish & Find Papers @ www.ijert.org
BROWSE
OPEN ACCESS
Call for Papers

Mitigation on Flood Attacks in DTN's

K . Naveen Kumar
Dept. of CSE,
Sree Vidyanikethan Engg. College,
Tirupati, A.P.

Mr . M . Ganesh Karthik
Assistant Professor, Dept. of CSE,
Sree Vidyanikethan Engg. College,
Tirupati, A.P.

***Abstract*— Disruption Tolerant Networks (DTNs) utilize the mobility of nodes and the opportunistic contacts among nodes for data communications. Due to the limitation in network resources such as contact opportunity and buffer space, DTNs are vulnerable to flood attacks in which attackers send as many packets or packet replicas as possible to the network, in order to deplete or overuse the limited network resources. In this paper, we employ rate limiting to defend against flood attacks in DTNs, such that each node has a limit over the number of packets that it can generate in each time interval and a limit over the number of replicas that it can generate for each packet. We propose a distributed scheme to detect if a node has violated its rate limits. Our idea is Spread and Swap model which detects attacker and address the attacker node to its neighbouring node. Which gives analysis on the probability of detection and evaluate the effectiveness and efficiency of our scheme simulations.**

Index Terms— DTN, security, flood attack, detection

I. INRTODUCTION

Disruption Tolerant Networks is mainly used for data transfer between mobile nodes which carried by human beings vehicles etc. DTNs provides data transfer when mobile nodes are only intermittently connected, making them applicable for applications wherever no communication infrastructure is available such as military situation and rural areas. Due to this inconsistency, two nodes can transfer data when they enter into communication range of each other.

Data is transferred via store-carry-forward method. This approach nodes store packets if they cannot find a next-hop node to deliver them to destinations. The each node first stores packets in its memory and then selectively transmits packets when it encounters other nodes based on various metrics including the last encounter time, the numbers of previousencounters, and the estimated packet delivery probability values to othernodes. Such metrics are derived from information provided by forwarding nodes themselves and it is hard to verify due to the network sparseness as well as the intermittent connectivity between nodes.

However DTN's has limitations such as low bandwidth and buffer space. Due to this they are liable to flood attacks. A flood attack is one in which the attackers send as many packet into the network and overuse the limited resources. Two types of flood attacks are packet flood attack and replica flood attack.

Also, mobile nodes may have limited buffer space. Due to the limitation in bandwidth and buffer space, DTNs are vulnerable to flood attacks. In flood attacks, maliciously or selfishly motivated attackers inject as many packets as possible into the network, or instead of injectingdifferent packets the attackers forward replicas of the same packet to as many nodes as possible. For convenience, we call the two types of attack packet flood attack and replica flood attack, respectively.

Flooded packets and replicas can waste the precious bandwidth and buffer resources, prevent begin packets from being forwarded and thus degrade the network service provided to good nodes. Moreover, mobile nodes spend much energy on transmitting/receiving flooded packets and replicas which may shorten their battery life. Therefore, it is urgent to secure DTNs against flood attacks.

Although many schemes have been proposed to defend against flood attackson the Internet and in wireless sensor networks, they assume persistent connectivity and cannot be directly applied to DTNs that have intermittent connectivity. There are many methods to prevent flood attacks, but none has been inducted for DTN's. A flood attack caused by outsider (unauthorized) can be prevented by authentication techniques. However it is not possible to prevent for attack caused by insiders (authorized). Thus, it is still anopen problem is to address flood attacks in DTNs.

In this paper, They employ rate limiting to defend against flood attacks in DTNs. In our approach, each and every node has a limit over the number of packets that it, as a sourcenode, cansend to the network in each time interval. Each node also has a limit over the number of replicas that it can generate for each packet (i.e., the number of nodes that it can forward each packet to). The two limits are used to mitigate packet flood and replica flood attacks, respectively. If a node violates its rate limits, it will be detected and its data traffic will be filtered. In this way, the amount of flooded traffic can be controlled.

Our basic idea is spread and swap model. Each node spread packets to neighbour nodes and trusted nodes swap information to source node as well as other neighbour node with secured keys. And process these spread and swap model until packet reached to destination node.

If an attacker floods more packets or replicas than its limit,By using rate limit certificate we can find flood attacker when they exceed rate limit of that attacker node.we use different cryptographic constructions to detect packet flood and replica flood attacks.

We provide a lower and upper bound of detection probability and investigate the problem of parameter selection to maximize detection probability under a certain amount of exchanged claims. The effectiveness and efficiency of our scheme are evaluated with extensive trace-driven simulations.

This paper is structured as follows. Section 2 motivates our work. Section 3 presents our models and basic ideas. Sections 4 and 5 present our scheme. Section 6 presents security and cost analysis. Section 7 presents simulation results. The last two sections present related work and conclusions, respectively.

II. MOTIVATION

A. The Potential Prevalence of Flood Attacks

Many nodes may launch flood attacks for malicious or selfish purposes. Malicious nodes, which can be the nodes deliberately deployed by the adversary or subverted by the adversary via mobile phone worms, launch attacks to congest the network and waste the resources of other nodes. Selfish nodes may also exploit flood attacks to increase their communication throughput. In DTNs, a single packet usually can only be delivered to the destination with a probability smaller than 1 due to the opportunistic connectivity. If a selfish node floods many replicas of its own packet, it can increase the likelihood of its packet being delivered, since the delivery of any replica means successful delivery of the packet. With packet flood attacks, selfish nodes can also increase their throughput, albeit in a subtler manner.

For example, suppose Alice wants to send a packet to Bob. Alice can construct 100 variants of the original packet which only differ in one unimportant padding byte, and send the 100 variants to Bob independently. When Bob receives any one of the 100 variants, he throws away the padding byte and gets the original packet.

B. The Effect of Flood Attacks

To study the effect of flood attacks on DTN routing and motivate our work, we run simulations on the MIT Reality trace (see more details about this trace in Section 7). We consider three general routing strategies in DTNs. 1) Single-copy routing after forwarding a packet out, a node deletes its own copy of the packet. Thus, each packet only has one copy in the network. 2) Multicopy routing: the source node of a packet sprays a certain number of copies of the packet to other nodes and each copy is individually routed using the single-copy strategy. The maximum number of copies that each packet can have is fixed. 3) Propagation routing: when a node finds it appropriate (according to the routing algorithm) to forward a packet to another encountered node, it replicates that packet to the encountered node and keeps its own copy. There is no preset limit over the number of copies a packet can have. In our simulations, SimBet, Spray-and- Focus (three copies allowed for each packet) and Propagation are used as representatives of the three routing strategies, respectively. In Propagation, a node replicates a packet to another encountered node if the latter has more frequent contacts with the destination of the packet.

Two metrics are used, The first metric is packet delivery ratio, which is defined as the fraction of packets delivered to their destinations out of all the unique packets generated. The second metric is the fraction of wasted transmissions (i.e., the transmissions made by good nodes for flooded packets). The higher fraction of wasted transmissions, the more network resources are wasted. We noticed that the effect of packet flood attacks on packet delivery ratio has been studied by Burgess using a different trace . Their simulations show that packet flood attacks significantly reduce the packet delivery ratio of single-copy routing but do not affect propagation routing much. However, they do not study replica flood attacks and the effect of packet flood attacks on wasted transmissions.

In our simulations, a packet flood attacker floods packets destined to random good nodes in each contact until the contact ends or the contacted node's buffer is full. A replica flood attacker replicates the packets it has generated to every encountered node that does not have a copy. Each good node generates thirty packets on the 121st day of the Reality trace, and each attacker does the same in replica flood attacks. Each packet expires in 60 days. The buffer size of each node is 5 MB, bandwidth is 2 Mbps and packet size is 10 KB.

Fig. 1 shows the effect of flood attacks on packet delivery ratio. Packet flood attack can dramatically reduce the packet delivery ratio of all three types of routing. When the fraction of attackers is high, replica flood attack can significantly decrease the packet delivery ratio of singlecopy and multicopy routing, but it does not have much effect on propagation routing. Packet flood attack can waste more than 80 percent of the transmissions made by good nodes in all routing strategies when the fraction of attackers is higher than 5 percent.

When 20 percent of nodes are attackers, replica flood attack can waste 68 and 44 percent of transmissions in single-copy and multicopy routing, respectively. However, replica flood attack only wastes 17 percent of transmissions in propagation routing. This is because each good packet is also replicated many times. Remarks. The results show that all the three types of routing are vulnerable to packet flood attack. Single-copy and multicopy routing are also vulnerable to

replica flood attack, but propagation routing is much more resistant to replica flood. Motivated by these results, this paper addresses packet flood attack without assuming any specific routing strategy, and addresses replica flood attack for single-copy and multicopy routing only.

III. OVERVIEW

A. Problem Definition

3.A.1 Defense against Packet Flood Attacks

We consider a scenario where each node has a rate limit L on the number of unique packets that it as a source can generate and send into the network within each time interval T. The time intervals start from time 0, T, 2T, etc. The packets generated within the rate limit are deemed legitimate, but the packets generated beyond the limit are deemed flooded by this node. To defend against packet flood attacks, our goal is to detect if a node as a source has generated and sent more unique packets into the network than its rate limit L per time interval. A node's rate limit L does not depend on any specific routing protocol, but it can be determined by a service contract between the node and the network operator as discussed in Section 3.1.3. Different nodes can have different rate limits and their rate limits can be dynamically adjusted. The length of time interval should be set appropriately. If the interval is too long, rate limiting may not be very effective against packet flood attacks. If the interval is too short, the number of contacts that each node has during one interval may be too nondeterministic and thus it is difficult to set an appropriate rate limit. Generally speaking, the interval should be short under the condition that most nodes can have a significant number of contacts with other nodes within one interval, but the appropriate length depends on the contact patterns between nodes in the specific deployment scenario.

3.A.2 Defense against Replica Flood Attacks

As motivated in Section 2, the defense against replica flood considers single-copy and multicopy routing protocols. These protocols require that, for each packet that a node buffers no matter if this packet has been generated by the node or forwarded to it, there is a limit l on the number of times that the node can forward this packet to other nodes. The values of l may be different for different buffered packets. Our goal is to detect if a node has violated the routing protocol and forwarded a packet more times than its limit l for the packet.

A node's limit l for a buffered packet is determined by the routing protocol. In multicopy routing, l ¼ L0 (where L0 is a parameter of routing) if the node is the source of the packet, and l ¼ 1 if the node is an intermediate hop (i.e., it received the packet from another node). In single-copy routing, l ¼ 1 no matter if the node is the source or an intermediate hop. Note that the two limits L and l do not depend on each other. We discuss how to defend against replica flood attacks for quota-based routing.

3.A.3 Setting the Rate Limit L

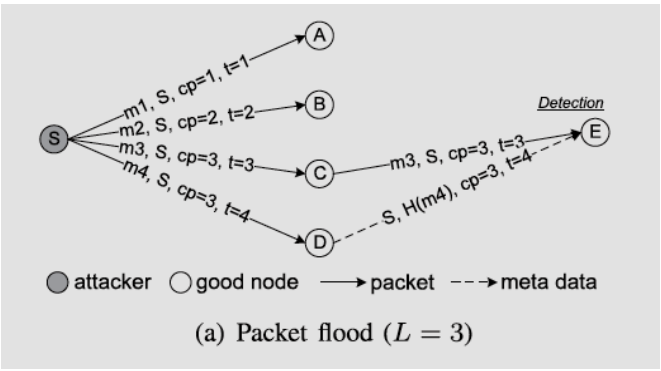
One possible method is to set L in a request-approve style. When a user joins the network, she requests for a rate limit from a trusted authority which acts as the network operator. In the request, this user specifies an appropriate value of L based on prediction of her traffic demand. If the trusted authority approves this request, it issues a rate limit certificate to this user, which can be used by the user to prove to other nodes the legitimacy of her rate limit. To prevent users from requesting unreasonably large rate limits, a user pays an appropriate amount of money Fig. 1. The effect of flood attacks on packet delivery ratio. In absent node, attackers are simply removed from the network. Attackers are selectively deployed to high-connectivity nodes. Fig. 2. The effect of flood attacks on the fraction of wasted transmission. Attackers are randomly deployed. virtual currency (e.g., the credits that she earns by forwarding data for other users for her rate limit. When a user predicts an increase (decrease) of her demand, she can request for a higher (lower) rate limit. The requestand approval of rate limit may be done offline. The flexibility of rate limit leaves legitimate users' usage of the network unhindered. This process can be similar to signing a contract between a smartphone user and a 3G service provider: the user selects a data plan (e.g., 200 MB/month) and pays for it; she can upgrade or downgrade the plan when needed.

B. Models and Assumptions

3.B.1 Network Model

In DTNs, since contact durations may be short, a large data item is usually split into smaller packets (or fragments) to facilitate data transfer. For simplicity, we assume that all packets have the same predefined size. Although in DTNs the allowed delay of packet delivery is usually long. Thus, we assume that each packet has a lifetime. The packet becomes meaningless after its lifetime ends and will be discarded.

We assume that every packet generated by nodes is unique. This can be implemented by including the source node ID and a locally unique sequence number, which is assigned by the source for this packet, in the packet header.

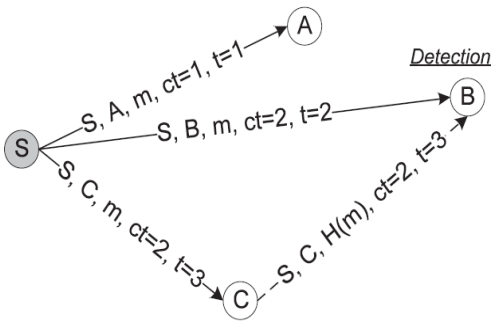


(a) Packet flood ($L = 3$)

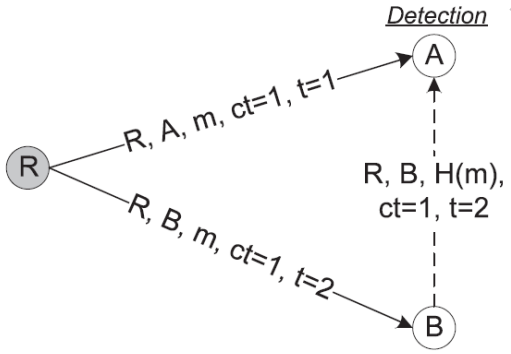
We also assume that time is loosely synchronized, such that any two nodes are in the same time slot at any time. Since the inter contact time in DTNs is usually at the scale of minutes or hours, the time slot can be at the scale of one minute. Such loose time synchronization is not hard to achieve.

3.B.2 Adversary Model

There are a number of attackers in the network. An attacker can flood packets and/or replicas. When flooding packets, the attacker acts as a source node. It creates and injects more packets into the network than its rate limit L. When flooding replicas, the attacker forwards its buffered packets (which can be generated by itself or received from other nodes)
more times than its limit l for them. The attackers may be insiders with valid cryptographic keys. Some attackers may collude and communicate via out-band channels.



(b) Replica flood by the source ($l = 2$)



(c) Replica flood by a relay ($l = 1$)

3.B.3 Trust Model

We assume that a public-key cryptography system is available. For example, Identity-Based Cryptography (IBC) has been shown to be practical for DTNs. In IBC, only an offline Key Generation Center (KGC) is needed. KGC generates a private key for each node based on the node's id, and publishes a small set of public security parameters to the node. Except the KGC, no party can generate the private key for a node id. With such a system, an attacker cannot forge a node id and private key pair. Also, attackers do not know the private key of a good node (not attacker).

Each node has a rate limit certificate obtained from a trusted authority. The certificate includes the node's ID, its approved rate limit L, the validation time of this certificate and the trusted authority's signature. The rate limit certificate can be merged into the public key certificate or stand alone.

C. Basic Idea: swap and spread models

3.C.1 Packet Flood Detection

To detect the attackers that violate their rate limit L, we must find the number of unique packets that each node as a source has generated and sent to the network in the current interval. However, since the node may send its packets to any node it contacts at any time and place, no other node can monitor all of its sending activities. To address this challenge, our idea is to let the node itself count the number of unique packets that it, as a source, has sent out, and carry information that up-to-date packet count (together with a little auxiliary information such as its ID and a timestamp) in each packet sent out. The node's rate limit certificate is also attached to the packet, such that other nodes receiving the packet can learn its authorized rate limit L. If an attacker is flooding more packets than its rate limit, since the real value is larger than its rate limit and thus a clear indicator of attack. The nodes which have received packets from the attacker carry and also included in those packets when they spread. When two of them swap, they check if there is any inconsistency between their collected packets. The attacker is identified when an inconsistency is found.

3.C.2 Replica Flood Detection

Swap and spread models can also be used to detect the attacker that forwards a buffered packet more times than its limit l. Specifically, when the source node of a packet or an intermediate hop transmits the packet to its next hop, it claims a transmission count which means the number of times it has transmitted this packet (including the current transmission). Based on if the node is the source or an intermediate node and which routing protocol is used, the next hop can know the node's limit l for the packet. Thus, if an attacker wants to transmit the packet more than l times, Similarly as in packet flood attacks, the attacker can be detected.

IV. OUR SCHEME

Our scheme uses two different cryptographic constructions to detect packet flood and replica flood attacks independently. When our scheme is deployed to propagation routing protocols, the detection of replica flood attacks is deactivated.

The detection of packet flood attacks works independently for each time interval. Without loss of generality, we only consider one time interval when describing our scheme. For convenience, we first describe our scheme assuming that all nodes have the same rate limit L, and relax this assumption in Section 4.8. In the following, we

use SIGið_Þ to denote node i's signature over the content in the brackets.

A. Construction for packet

Two pieces of metadata are added to each packet (see Fig. 4), Packet Spread (PS) and Transmission Swap (TS). PS and TS are used to detect packet flood and replica flood attacks, respectively. PS is added by the source and transmitted to later hops along with the packet. TS is generated and processed hop-by-hop. Specifically, the source generates a TS and appends it to the packet. When the first hop receives this packet, it peels off the TS; when it forwards the packet out, it appends a new TS to the packet. This process continues in later hops. Each hop keeps the PS of the source and the TS of its previous hop to detect attacks.

4.A.1 PS

When a source node S sends a new packet m (which has been generated by S and not sent out before) to a contacted node, it generates a PS as follows:

PS: S, Cp, t, H(m),
SIGs(H(H(m)|S|Cp|t)).

Here, t is the current time. Cp is the packet count of S, which means that this is the cth p new packet S has created and sent to the network in the current time interval.S increases cp by one after sending m out. The PS is attached to packet m as a header field, and will always be forwarded along with the packet to later hops. When the contacted node receives this packet, it verifies the signature in the PS, and checks the value of cp. If cp is larger than L, it discards this packet; otherwise, it stores this packet and the PS.

4.A.2 TS

When node A transmits a packet m to node B, it appends a TS to m. The TS includes A's current transmission count ct for m (i.e., the number of times it has transmitted m out) and the current time t. The TS is

TS:A,B,H(m),Ct,t,SIGA(H(A|B|H(m)|Ct|t)).

B checks if ct is in the correct range based on if A is the source of m. If ct has a valid value, B stores this TS. In single-copy and multicopy routing, after forwarding m for enough times, A deletes its own copy of m and will not forward m again.

B. Protocol

Suppose two nodes contact and they have a number of packets to forward to each other. Then our protocol is sketched in Algorithm 1.

Algorithm 1. The protocol run by each node in a contact. Metadata (PS and TS) exchange and attack detection

1: if Have packets to spread then
2: For each new packet, generate a PS;
3: For all packets, generate their TSs and sign them with a hash function;
4: Send every packet with the PS and TS attached;
5: end if
6: if Receive a packet then
7: if Signature verification fails or the count value in its PS or TS is invalid then
8: Discard this packet;
9: end if
10: Check the PS against those locally collected and generated in the same time interval to detect inconsistency;
11: Check the TS against those locally collected for inconsistency;
12: if Inconsistency is detected then
13: Tag the signer of the PS (TS, respectively) as an attacker and add it into a blacklist;
14: Disseminate an alarm against the attacker to the network;
15: else
16: Store the new PS and swap (TS, respectively);
17: end if
18: end if

When a node forwards a packet, it attaches a TS to the packet. Since many packets may be forwarded in a contact and it is expensive to sign each TS separately, an efficient signature construction is proposed. The node also attaches a PS to the packets that are generated by itself and have not been sent to other nodes before. When a node receives a packet, it gets the PS and TS included in the packet. It checks them against the claims that it has already collected to detect if there is any inconsistency. Only the PSs generated in the same time interval (which can be determined by the time tag) are cross-checked. If no inconsistency is detected, this node stores the PS and TS locally.

To better detect flood attacks, the two nodes also exchange a small number of the recently collected PSs and TSs and check them for inconsistency. This metadata exchange process is separately presented.

When a node detects an attacker, it adds the attacker into a blacklist and will not accept packets originated from or forwarded by the attacker. The node also disseminates an alarm against the attacker to other nodes.

C. Local Data Structures

Each node collects PSs and TSs from the packets that it has received and stores them locally to detect flood attacks. Let us look at a received packet m and the PS and TS included in this packet. Initially, this pair of PS and TS are stored in full with all the components. When this node removes m from its buffer (e.g., after m is delivered to the destination or dropped due to expiration), it compacts this pair of claims to reduce the storage cost. If this pair of claims have been sampled for metadata exchange, they will be stored in full until the exchange process ends and be compacted afterward.

4.C.1 Inconsistency Check with PS

From the PS node W gets: the source node ID S, packet count cp, timestamp t, and packet hash H. To check

inconsistency, W first uses S and t to map the PS to the structure CiS (see (4)). Then it reconstructs the hash remainder of H using the locators in Ci S. If the bit indexed by the packet count cp is set in the bit-vector but the hash remainder is not included in Ci S, count reuse is detected and S is an attacker. The inconsistency check based on compact PSs does not cause false positive, since a good node never reuses any count value in different packets generated in the same interval. The inconsistency check may cause false negative if the two inconsistent PSs have the same hash remainder. However, since the attacker does not know which bits constitute the hash remainder, the probability of false negative is only 2_8. Thus, it has minimal effect on the overall detection probability.

4.C.2 Inconsistency Check with TS

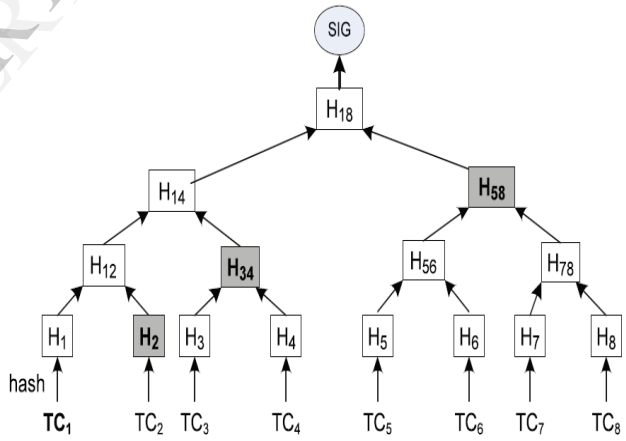
From the TS node W gets: the sender ID R, receiver ID Q and transmission count ct. If Q is W itself (which is possible if the TS has been sent out by W but returned by an attacker), W takes no action. Otherwise, it uses R to map the TS to the structure CR (see (6)). If there is a 2-tuple ½ e H0 32; c0 t_ in CR that satisfies 1) e H0 32 is the same as the remainder of H, and 2) c0 t ¼ ct, then the issuer of the TS (i.e., R) is an attacker. The inconsistency check based on compact TSs does not cause extra false negative. False positive is possible but it can be kept low as follows: node W may falsely detect a good node R as an attacker if it has received two TSs generated by R that satisfy two conditions: 1) they are generated for two different packets, and 2) they have the same hash remainder. For 32-bit hash remainder, the probability that each pair of TSs lead to false detection is 2_32. In most cases, we expect that the number of TSs generated by R and received by W is not large due to the opportunistic contacts of DTNs, and thus the probability of false detection is low. As W receives more TSs generated by R, it can use a longer (e.g., 64-bit) hash remainder for R to keep the probability of false detection low. Moreover, such false detection is limited to W only, since W cannot convince other nodes to accept the detection with compact TS.

D. Alarm

Suppose in a contact a node receives a claim CCr from a forwarded data packet or from the metadata exchange process (see Section 5.3) and it detects inconsistency between CCr and a local claim CCl that the node has collected. CCr is a full claim as shown in Formula 1 (or 2),but CCl may be stored as a full claim or just a compact structure shown in Formula 3 (or 5).If CCl is a full claim, the node can broadcast (via Epidemic routing a global alarm to all the other nodes to speed up the attacker detection process. The alarm includes the two full claims CCl and CCr. When a node receives an alarm, it verifies the inconsistency between the two included claims and their signatures. If the verification succeeds, it adds the attacker into its blacklist and broadcasts the alarm further; otherwise, it discards the alarm.

The node also discards the alarm if it has broadcast another alarm against the same attacker. If the detecting node stores CCl as a compact structure, it cannot convince other nodes to trust the detection since the compact structure does not have the attacker's signature.Thus it cannot broadcast a global alarm. However, since the attacker may have reused the count value of CCr to other claims besides CCl, the detecting node can disseminate a local alarm that only contains CCr to its contacted nodes who have received those claims. These contacted nodes can verify the inconsistency between CCr and their collected claims, and also detect the attacker. If any of these nodes still stores a full claim inconsistent with CCr, it can broadcast a global alarm as done in the previous case; otherwise, it disseminates a local alarm. As this iterative process proceeds, the attacker can be quickly detected by many nodes. Each node only disseminates one local alarm for each detected attacker.

A local alarm and a global alarm against the same attacker may be disseminated in parallel. If a node receives the global alarm first and then receives the local alarm, it discards the local alarm. If it receives the local alarm first, when it receives the global alarm later, it discards the local alarm and keeps the global alarm. An attacker may falsify an alarm against a good node. However, since it does not have the node's private key (as our assumption), it cannot forge the node's signatures for the claims included in the alarm. Thus, the alarm will be discarded by other nodes and this attack fails.



E. Efficient TS Authentication

The TSs of all the packets transmitted in a contact should be signed by the transmitting node. Since the contact may end at any unpredictable time, each received TS must be individually authenticated. A naive approach is to protect each TS with a separate public-key signature, but it has high computation cost in signature generation and verification. computation cost of public-key-based signature on all the TSs that the node sends out in a contact. Specifically, after a node generates the TSs (without signature) for all the packets it want to send, it constructs a hash tree upon these partial TSs, and signs the root of the tree with a public-key-based signature. Then the signature of a TS includes this root signature and a few elements of the tree. In this way, for all

the TSs sent by the sender in a contact, only one public-key based signature is generated by the sender and verified by the receiver.

F. Dealing with Different Rate Limits

Previously we have assumed that all nodes have the same rate limit L. When nodes have different rate limits, for our detection scheme to work properly, each intermediate nodethat receives a packet needs to know the rate limit L of the source of the packet, such that it can check if the packet count is in the correct range 1; 2; . . . ; L. To do so, when a source node sends out a packet, it attaches its rate limit certificate to the packet. The intermediate nodes receiving this packet can learn the node's authorized rate limit from the attached certificate.

V. METADATA EXCHANGE

When two nodes contact they exchange their collected PSs and TSs to detect flood attacks. If all claims are exchanged, the communication cost will be too high. Thus, our scheme uses sampling techniques to keep the communication cost low. To increase the probability of attack detection, one node also stores a small portion of claims exchanged from its contacted node, and exchanges them to its own future contacts. This is called redirection.

A.Sampling

Since PSs and TSs are sampled together (i.e., when a PS is sampled the TS of the same packet is also sampled), in the following we only consider PSs. A node may receive a number of packets (each with a PS) in a contact. It randomly samples Z (a system parameter) of the received PSs, and exchanges the sampled PSs to the next K (a system parameter) different nodes it will contact, excluding the sources of the PSs and the previous hop from which these PSs are received.

However, a vulnerability to tailgating attack should be addressed. In tailgating attack, one or more attackers tailgate a good node to create a large number (say, d) of frequent contacts with this node, and send Z packets (not necessarily generated by the attackers) to this node in each created contact. If this good node sends the Zd PSs of these contacts to the next K good nodes it contacts, much effective bandwidth between these good nodes will be wasted, especially in a large network where K is not small.

To address this attack, the node uses an inter-contact sampling technique to determine which PSs sampled in historical contacts should be exchanged in the current contact. Let SK denote a set of contacts. This set includes the minimum number of most recent contacts between thisnode and at least K other different nodes. Within this set, all the contacts with the same node are taken as one single contact and a total of Z PSs are sampled out of these contacts. This technique is not vulnerable to the tailgating attack since the number of claims exchanged in each contact is bounded by a constant.

B. Redirection

There is a stealthy attack to flood attack detection. For replica flood attacks, the condition of detection is that at least two nodes carrying inconsistent TSs can contact. However, suppose the attacker knows that two nodes A and B never contact. Then, it can send some packets to A, and invalidly replicate these packets to B.

In this scenario, this attacker cannot be detected since A and B never contact. Similarly, the stealthy attack is also harmful for some routing protocols like Spray-and-Wait in which each packet is forwarded from the source to a relay and then directly delivered from the relay to the destination. To address the stealthy attack, our idea is to add one level of indirection. A node redirects the Z PSs and TSs sampled in the current contact to one of the next K nodes it will contact, and this contacted node will exchange (but not redirect again) these redirected claims in its own subsequent contacts. Look at the example in Fig. 6. Suppose attacker S sends mutually inconsistent packets to two nodes A and B which will never contact.

Suppose A and B redirect their sampled PSs to node C and D, respectively. Then so long as C and B or D and A or C and D can contact, the attack has a chance to be detected. Thus, the successful chance of stealthy attack is significantly reduced.

C. The Exchange Process

Each node maintains two separate sets of PSs (TSs, respectively in the following) for metadata exchange, a sampled set which includes the PSs sampled from the most recent contacts with K different nodes (i.e., SK in Section 5.1), and a redirected set which includes the PSs redirected from those contacts. Both sets include Z PSs obtained in each of those contacts. When two nodes A and B contact, they first select KZ from each set with the inter-contact sampling technique (see Section 5.1), and then send these PSs to each other.

When A receives a PS, it checks if this PS is inconsistent with any of its collected PSs using the method described in Section 4.5. If the received PS is inconsistent with a locally collected one and the signature of the received PS is valid, A detects that the issuer (or signer) of the received PS is an attacker. Out of all the PSs received from B, A randomly selects Z of the PSs from the sampled set of B, and stores them to A's redirected set. All other PSs receivedfrom B are discarded after inconsistency check.

D. Metadata Deletion

A node stores the PSs and TSs collected from received data packets for a certain time denoted by _ and deletes them afterward. It deletes the claims redirected from

other nodes immediately after it has exchanged them to K different nodes.

VI. ANALYSIS OF SPREAD AND SWAP

We assume the system model described in Section 3. For completeness, we state bounds on time in system and number in system for the random spread model. We then analyze random swap in detail.

A. Results for the Random Spread Model

For the sake of analysis, we approximate the random walk mobility model by the i.i.d walk mobility model, where in each slot a node can move to any random location on the grid with equal probability. In this model, since the transmission radius is p(2), the probability that a node has at least one node within its transmission radius is given by $p = 1-(1- 9/W^2)(N-1)$. We give bounds for the time in the system and average number of copies of a query in the system for the random spread model. We leave out the proofs of the next two theorems for the sake of brevity.

B. Analysis of Random Swap

If two nodes are neighbors, then they each randomly choose a buffer location and swap the queries in these locations. We assume that M>>B and k<<N.1 This implies that we can approximate the the distribution of queries in each buffer to be uniform and that we can treat each of the k copies of a query independently. The analysis in this section depends only on the above reasonable assumptions.

C. Match AnalysisWith a Genie

In this section, we compute an upper bound on what the probability of match could be. When two nodes are neighbours of each other, we assume that there is a genie which looks into both buffers and generates all possible matches.

D. The Basic Attack

First we consider a basic attack in which an attacker S floods two sets of mutually inconsistent packets to two good nodes A and B, respectively. Each flooded packet received by A is inconsistent with one of the flooded packets received by B. In the contacts with A and B, S also forwards some normal, not flooded, packets to A and B to make the attack harder to detect. Let y denote the proportion of flooded packets among those sent by S. For simplicity, we assume y is the same in both contacts.

Suppose A and B redirect the claims sampled in the contact with S to C and D, respectively. To consider the worst case performance, suppose theflooded packets are not forwarded from A and B to other nodes i.e., only A and B have the inconsistent claims.

Note that the analysis also applies to the detection of replica flood attacks. For convenience, we define node A's (or B's) detection window as from the time it receives the flooded packets to the time it exchanges the sampled claims to K nodes, and node C's (or D's) detection window as from the time it receives the redirected claims to the time it exchanges them to K nodes. The attacker has a chance to be detected if node pairs hA;Bi, hA;Di, hC;Bi and hC;Di can contact within their detection windows. Table 1 shows the variables used in the analysis.

E. Cost Analysis

6.E.1 Communication

The communication cost mainly has two parts. One part is the PS and TS transmitted with each packet, and the other part is the partial claims transmitted during metadata exchange. As to the latter, at most 4ZK PSs and 4ZK TSs are exchanged in each contact, with one half for sampled and the other half for redirected claims.

6.E.2 Computation

As to signature generation, a node generates one signature for each newly generated packet. It also generates one signature for all its TSs as a whole sent in a contact. As to signature verification, a node verifies the signature of each received packet. It also verifies one signature for all the TSs as a whole received in one contact.

6.E.3 Storage

Most PSs and TSs are compacted when the packets are forwarded. The Z sampled PSs and TSs are stored in full until the packets are forwarded or have been exchanged to K nodes, whichever is later, and then compacted. For each received packet, less than 20 bytes of compact claims are stored for time duration.

F. Collusion Analysis

6.F.1 Packet Flood Attack

One attacker may send a packet with a dishonest packet count to its colluder, which will forward the packet to the network. Certainly, the colluder will not exchange the dishonest PS with its contacted nodes.

However, so long as the colluder forwards this packet to a good node, this good node has a chance to detect the dishonest claim as well as the attacker. Thus, the detection probability is not affected by this type of collusion.

6.F.2 Replica Flood Attack

When attackers collude, they can inject invalid replicas of a packet without being detected, but the number of flooded replicas is effectively limited in our scheme. More specifically, in our scheme for a unique packet all the M

colluders as a whole can flood a total of M-1 invalid replicas without being detected.

To the contrast, when there is no defense, a total of N-M invalid replicas can be injected by the colluders for each unique packet. Since the number of colluders is not very large, our scheme can still effectively mitigate the replica flood attack.

VII. RELATED WORK

Our scheme bears some similarity with previous approaches that detect node clone attacks in sensor networks. Both rely on the identification of some kind of inconsistency to detect the attacker. However, their approaches assumes consistent connectivity between nodes which is unavailable in DTNs. Moreover, in multi-hop systems there is also the issue of providing incentives to users to relay information for each other. There are a number of papers on ad hoc networks which address this issue .

These typically involve mechanisms where users pay each other, or users help each other in relaying traffic in the hope that they will be helped similarly in the future. However implementing such mechanisms in practice is extremely difficult. In contrast, in our architecture, since the cellular infrastructure is utilized, the network operator can provide incentives for users to relay information for each other.

Also since the information will be short messages, providing incentives is not expensive.However, these work do not address flood attacks. Other work deter abuse by correlating the amount of network resources that a node can use with the node's contributions to the network in terms of forwarding. This approach has been proposed for mobile ad hoc networks, but it is still not clear how the approach can be applied to DTNs, where nodes are disconnected most of the time.

Another recent work proposed a batch authentication protocol for DTNs, which verifies multiple packet signatures in an aggregated way to save the computation cost. This work is complementary to ours, and their protocol can also be used in our scheme to further reduce the computation cost of authentication.

Parallel to our work, also proposed a scheme to detect resource misuse in DTNs. In their scheme, the gateway of a DTN monitors the activities of nodes and detects an attack if there is deviation from expected behavior. Different from their work that requires a special gateway for counting, our scheme works in a totally distributed manner and requires no special nodes.

VIII. CONCLUSIONS

In this paper, we employed rate limiting to mitigate flood attacks in DTNs, and proposed a scheme which detects and identified attacker node by using Spread and Swap model to probabilistically detect the violation of rate limit in DTN environments. Our scheme uses efficient constructions to keep the computation, communication and storage cost low. Extensive trace-driven simulations showed that our scheme is effective to detect flood attacks and it achieves such effectiveness in an efficient way.

REFERENCES

[1]. K. Fall, "A Delay-Tolerant Network Architecture for Challenged Internets," Proc. ACM SIGCOMM, pp. 27-34, 2003.
[2]. P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket Switched Networks and Human Mobility in Conference Environments," Proc. ACM SIGCOMM, 2005.
[3]. M. Motani, V. Srinivasan, and P. Nuggehalli, "PeopleNet: Engineering a Wireless Virtual Social Network," Proc. MobiCom, pp. 243-257, 2005.
[4].J. Burgess, B. Gallagher, D. Jensen, and B. Levine, "Maxprop:Routing for Vehicle-Based Disruption-Tolerant Networks," Proc. IEEE INFOCOM, 2006.
[5].S.J.T.U.Grid Computing Center, "Shanghai Taxi Trace Data," http://wirelesslab.sjtu.edu.cn/, 2012.
[6].J. Mirkovic, S. Dietrich, D. Dittrich, and P. Reiher, Internet Denial of Service: Attack and Defense Mechanisms. Prentice Hall, 2005.
[7]. C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," Proc. IEEE First Int'l Workshop Sensor Network Protocols and Applications, 2003.
[8].E. Daly and M. Haahr, "Social Network Analysis for Routing in Disconnected Delay-Tolerant MANETs," Proc. MobiHoc, pp. 32-40, 2007.
[9].W. Gao, Q. Li, B. Zhao, and G. Cao, "Multicasting in Delay Tolerant Networks: A Social Network Perspective," Proc. ACM MobiHoc, 2009.
[10]. F. Li, A. Srinivasan, and J. Wu, "Thwarting Blackhole Attacks in Distruption-Tolerant Networks Using Encounter Tickets," Proc. IEEE INFOCOM, 2009.