ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

Recognition of Risky Attacks in Wireless Sensor Networks

N.Rajyalakshmi[1], K.Narayana[2]

Student, Dept of Computer Science and Engineering, Seshachala Institute of Technology, Puttur, India[1]

Associate Professor, Dept of Computer Science and Engineering, Seshachala Institute of Technology, Puttur, India[2]

ABSTRACT: Wi-fi indicator systems are vulnerable to the node replicated, and several distribute methods have been suggested to identify this attack. However, two novel node replicated discovering methods with different tradeoffs on system conditions and efficiency. The first one is depending on a allocated hash table (DHT), by which a fully decentralized, key-based caching and verifying system is designed to capture duplicated nodes effectively. The method efficiency on efficient storage spending and great security level is apparently subtracted through a possibility model, and the causing equations, with necessary improvements for real application, are reinforced by the models. Although the DHT-based method have similar interaction cost as previous approach it may be considered a little great for some situation. To address this concern, our second allocated discovering method, known as endlessly instructed research provides good interaction efficiency for heavy indicator systems, by a probabilistic instructed delivering technique along with unique initial direction and boundary dedication. The simulator results maintain the method design and show its efficiency on interaction expense and acceptable recognition possibility. These “Vampire” strikes are not particular to any particular method, but rather depend on the qualities of many popular sessions of guiding methods. We find that all analyzed methods are vulnerable to Creature of the night strikes, which are harmful, difficult to identify, and are convenient to carry out using as few as one hateful expert delivering only protocol-compliant information. In the most severe, a single Creature of the night can increase network-wide energy customized by a factor of O(N), where N in the number of system nodes. We talk about methods to relieve these types of strikes, along with a new proof-of-concept method that provably range the break due to Skeletons during the bundle delivering stage.

1.INTRODUCTION

we present two novel, realistic node replicated discovering methods with different tradeoffs on system circumstances and efficiency. The first offer is depending on a allocated hash table (DHT) ,by which a fully decentralized, key-based caching and verifying system is designed to capture replicated nodes. The protocol’s efficiency on storage intake and a crucial protection measurement are theoretically subtracted through a possibility design, and the causing equations, with necessary modification for real program, are reinforced by the models. In contract with our research, the full simulator results display that the DHT-based method can identify node replicated with high protection level and keeps powerful fight against adversary’s strikes. Our second method, known as arbitrarily instructed evaluation, is future to provide highly efficient interaction efficiency with adequate recognition possibility for heavy indicator systems. In the method, originally nodes deliver declaring information containing a neigh borlist along with a highest possible hop restrict to unintentionally chosen neighbors; then, the following concept transmitting is controlled by a probabilistic instructed strategy to around maintain a line property through the system as well as to have sufficient randomness for better efficiency on interaction and strength against attacker. In addition, boundary take care of procedure is employed to further reduce interaction payload. During sending, advanced nodes discover declaring information for node replicated recognition. By style, this method takes in almost least storage, and the simulator display that it outperforms all other recognition methods in terms of concept cost, while the recognition possibility is acceptable. While these techniques can avoid strikes on the short-term availability of a system, they do not address strikes hat impact long-term availability—the most long lasting being rejected of service strike is to completely wipe out nodes’ battery power power. This is an example of a source fatigue strike, with battery power pack as the source of interest. In this document, we consider how redirecting methods, even those designed to be protected, lack protection from these strikes, which we call Creature of the night strikes, since they use up the life from systems nodes. These strikes are unique from previously analyzed DoS, decrease of quality (RoQ), and redirecting emails strikes as they do not impact immediate convenience of use, but rather perform over time to completely turn off a system. While some of the person strikes are

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

simple, and energy depleting and source fatigue strikes have been mentioned before [53], [59], [68] prior perform has been mostly limited to other levels of the method collection, e.g., method access control (MAC) or program levels, and to our knowledge there is little conversation, and no thorough research or reducing, of routing-layer source failure strikes. Creature of the night strikes are not protocol-specific, in that they do not depend on style qualities or finalization mistakes of particular redirecting methods, but rather create common qualities of method sessions such as link-state, source vector, source redirecting, and geographical and fire redirecting. Neither do these strikes depend on overflow the system with considerable amounts of information, but rather try to deliver as little information as possible to get the biggest energy strain, avoiding a rate caution solution. Since Skeletons use protocol-compliant information, these strikes are very difficult to identify and avoid.

II. DHT-BASED DETECTION PROTOCOL

The principle of our first distributed detection protocol is to make use of the DHT mechanism to form a decentralized caching and checking system that can effectively sense cloned nodes. Essentially, DHT enables sensor nodes to distributively construct an *overlay network* upon a physical sensor network and provides an efficient key-based routing with in the overlay network. A message connected with a key will be transmitted through the overlay network to reach a destination node that is solely strong-minded by the key; the source node does not need to specify or know which node a message's purpose is the DHT key-based routing takes care of moving details by the message's key. More importantly, messages with a same key will be stored in one purpose node. Those facts build the foundation for our first detection protocol. As a beginning of a round of DHT-based clone detection, the initiator broadcasts the action message including a random *seed*. Then, every observer constructs a claiming message for each neighbor node, which is referred to as an *examinee* of the observer and the message, and sends the message with probability p_c independently. The introduction of the claiming probability is p_c intended to reduce the communication overwork in case of a high-node-degree network. In the protocol, a message's DHT key that determines its routing and target is the hash value of concatenation of the seed and the examinee ID. By means of the DHT mechanism, a claiming message will finally be transmitted to a deterministic destination node, which will cache the ID-location pair and check for node clone discovery, acting as an inspector. In addition, some intermediate nodes also behave as inspectors to improve resilience against the adversary in an efficient way.

A. Distributed Hash Table

Before diving into the detection protocol, we briefly introduce DHT techniques. In principle, a distributed hash table is a decentralized distributed system that provides a key-based lookup service similar to a hash table: (key, record) pairs are stored in the DHT, and any participating node can efficiently store and recover records associated with specific keys. By design, DHT distributes responsibility of maintaining the map from keys to records among nodes in an efficient and balanced way, which allows DHT to scale to extremely large networks and be fitting to serve as a facility of distributed node clone detection. There are several different types of DHT proposals, such as CAN, Chord, and Pastry . Generally, CAN has least efficiency than others in terms of communication cost and scalability, and it is rarely employed in real systems. By contrast, Chord is widely used, and we choose Chordasa DHT implementation to demonstrate our protocol. However, our protocol can easily travel to build upon Pastry and present similar security and performance results. The technical core of Chord is to form a massive virtual ring in which every node is located at one point, own a segment of the periphery. To achieve pseudo-randomness on output, a hash function H is used to map an chance input into a b-bit space, which can be conceive as a ring. Each node is assigned with a triad coordinate upon joining the network. Practically for our protocol, a node's Chord point's coordinate is the

hash value of the node's MAC address. All nodes divide the ring into segment by their Chord points. Likewise, the key of a record is the result of the hash function. Every node is responsible for one segment that ends at the node's Chord point, and all records whose keys fall into that segment will be transmit to and stored in that node. As the kernel of efficient key-based routing, every node maintains a *finger table* of size $t = O(\log n)$ to facilitate a binary-tree search. Specifically, the finger table for a node with Chord coordinate y contains information of nodes that are respectively responsible for holding the t keys: $(y+2^{b-i})$mod 2^b fir $i \in [1,t]$. If two nodes are within the ring-segments distance, they are each other's predecessor and successor by the order their coordinates, with respect to predefined g. In theory, a Chord node only needs to know its direct predecessor and finger table. To improve flexibility against network churn

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

and enhance routing efficiency, every node additionally maintains a successor table, containing its g successors. Typical values of g are between 10 and 20.A warm example of a Chord system with small parameters is given in Fig. 1. In this system, if node wants to query a record with key , it first looks up its successor table. Since 97 is not in (109, 61], namely (direct predecessor, the last successor], node proceeds with the finger table and finds that the next forwarding node should be because97 [72: the first item in finger table corresponding to , 109: direct predecessor). When receives this query about, by checking its successor table with two nodes of and , it determines the destination should be , as (88: itself, 109: the first successor]. When node gets the query, it knows itself be the destination because 97 (88: direct predecessor, 109: itself].

B. Protocol Details

As a condition, all nodes cooperatively build a Chord overlay network over the sensor network. Cloned node may not participate in this procedure, but it does not give them any benefit of avoiding detection. The construction of the overlay network is independent of node clone detection. As a result, nodes possess the information of their direct ancestor and successor in the Chord ring. In addition, each node caches information of its g consecutive successors in its *successors table*. Many Chord systems use this kind of cache mechanism to reduce the communication cost and enhance systems strength. More importantly in our protocol, the facility of the successors table contributes to the economical selection of inspectors. One detection round consists of three stages.

Stage 1: Initialization

To activate all nodes starting a new round of node clone detection, the initiator uses a broadcast verification scheme to release an *action message* with a boringly increasing nonce, a random round seed, and an action time. The nonce is future to prevent adversary from launching a DoS attack by repeating broadcasting action messages. The action message is defined by

$$M_{\mathrm{ACT}} = \mathrm{nonce}, \mathrm{seed}, \mathrm{time}, \{\mathrm{nonce} \parallel \mathrm{seed} \parallel \mathrm{time}\}_{K^{-1}_{\mathrm{initiator}}}.$$

Stage 2: Claiming neighbors information

Upon receiving an action message, a node verifies if the message nonce is greater than last nonce and if the message signature is valid. If both pass, the node updates the nonce and stores the seed. At the chosen action time, the node operates as an observer that generates a claiming message for each neighbor (examinee) and transmits the message through the overlay network with respect to the claim probability p_c.The claiming message by observer for examinee is β constructed by

$$M_{\alpha 4\beta} = \mathrm{id}_\beta, L_\beta, \mathrm{id}_\alpha, L_\alpha, \{\mathrm{id}_\beta \parallel L_\beta \parallel \mathrm{id}_\alpha \parallel L_\alpha \parallel \mathrm{nonce}\}_{K^{-1}_\alpha}$$

Where L_α, $L_\beta$are locations of α and β,respectively. Nodes can start transmitting claiming messages at the same time, but then huge traffic may cause serious intrusion and degrade the network capacity. To relieve this problem, we may specify a sending period, during which nodes randomly pick up a transmission time for every claim message.

Stage 3: Processing claiming messages

A claiming message will be forward to its destination node via several Chord intermediate nodes. Only those nodes in the overlay network layer (i.e., the source node, Chord intermediate nodes, and the destination node) need to process a message,

whereas other nodes along the path simply route the message to temporary targets. Algorithm 1 for handling a message is the kernel of our DHT-based detection protocol. If the algorithm returns NIL, then the message has arrived at its destination. Otherwise, the message will be subsequently forwarded to the next node with the ID that is returned by Algorithm 1.

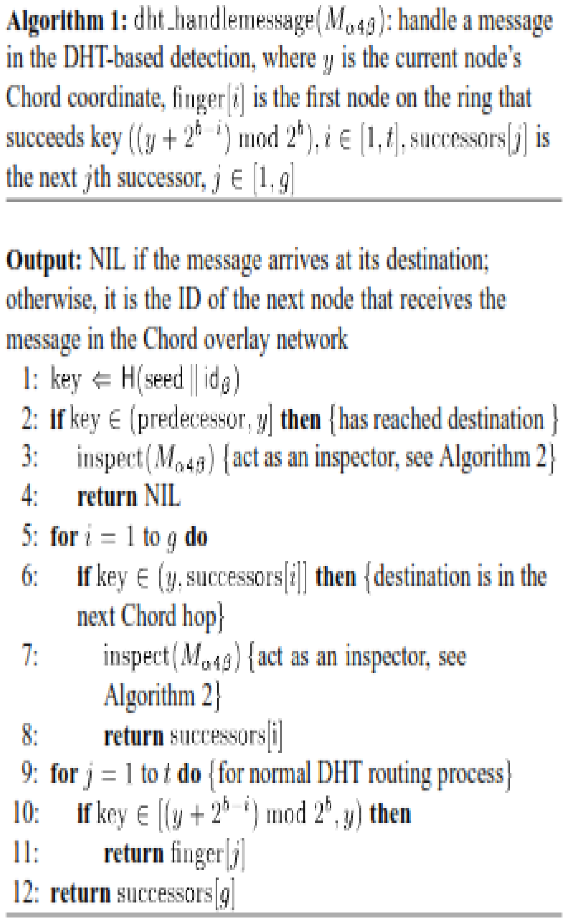
Criteria of determining inspectors*:* During handling a message in Algorithm 1, the node acts as an inspector if one of the following conditions is satisfied.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

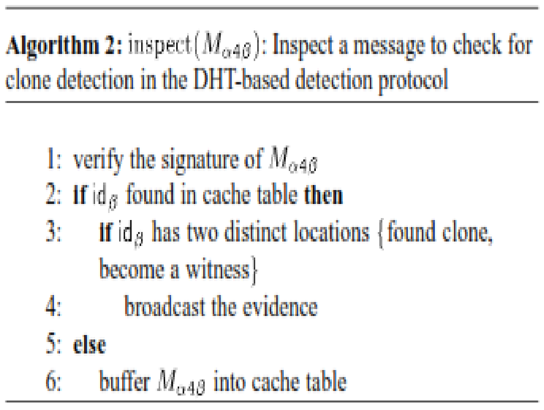
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015



Algorithm 1: dht_handlemessage($M_{\alpha 4\beta}$): handle a message in the DHT-based detection, where y is the current node's Chord coordinate, finger[i] is the first node on the ring that succeeds key $((y + 2^{b-i}) \bmod 2^b)$, $i \in [1, t]$, successors[j] is the next jth successor, $j \in [1, g]$

```
Output: NIL if the message arrives at its destination;
otherwise, it is the ID of the next node that receives the
message in the Chord overlay network
 1: key ⇐ H(seed || $id_\beta$)
 2: if key ∈ (predecessor, $y$] then {has reached destination }
 3:     inspect($M_{\alpha 4\beta}$) {act as an inspector, see Algorithm 2}
 4:     return NIL
 5: for $i = 1$ to $g$ do
 6:     if key ∈ ($y$, successors[$i$]] then {destination is in the
        next Chord hop}
 7:         inspect($M_{\alpha 4\beta}$) {act as an inspector, see
            Algorithm 2}
 8:         return successors[i]
 9: for $j = 1$ to $t$ do {for normal DHT routing process}
10:     if key ∈ [$(y + 2^{b-i}) \bmod 2^b, y$) then
11:         return finger[$j$]
12: return successors[$g$]
```



Algorithm 2: inspect($M_{\alpha 4\beta}$): Inspect a message to check for clone detection in the DHT-based detection protocol

```
1: verify the signature of $M_{\alpha 4\beta}$
2: if $id_\beta$ found in cache table then
3:     if $id_\beta$ has two distinct locations {found clone,
       become a witness}
4:         broadcast the evidence
5: else
6:     buffer $M_{\alpha 4\beta}$ into cache table
```

1) This node is the destination node of the claiming message.
2) The destination node is one of the g successor of the node. In other words, the target node will be reached in the next Chord hop. While the first criterion is intuitive, the second one is subtle and critical for the protocol performance. By Algorithm 1, roughly $\frac{g}{g+1}$ of all claiming messages related to a same examinee's ID will pass through one of the predecessor of

the destination. Thus, those nodes are much more likely to be able to detect a clone than randomly selected inspectors. As a result, this criterion to decide inspectors can increase the average number of witnesses at a little extra memory cost. We will theoretically quantify those performance measurements later. In Algorithm 1, to examine a message for node clone detection, an inspector will invoke Algorithm 2, which compares the message with previous inspected messages that are buffered in the cache table. Naturally, all records in the cache table should have different examinee IDs, as indirect in Algorithm 2. If detecting a clone, which means that there exist two messages $M_{\alpha 4\beta}$ and $M_{\alpha' 4\beta'}$ satisfying $\mathrm{id}_{\beta} = \mathrm{id}_{\beta'}$ and $L_{\beta} \neq L_{\beta'}$, the witness node then broadcasts the evidence to notify the whole network. All integrity nodes verify the evidence $M_{\mathrm{evidence}} = (M_{\alpha 4\beta}, M_{\alpha' 4\beta'})$message and stop communicating with the cloned nodes. To prevent cloned nodes from joining the network in the future, a revocation list of compromised nodes IDs may be maintained by nodes individually. It is worth noting that messages $M_{\alpha 4\beta}$ and $M_{\alpha' 4\beta'}$are authenticated by observers and, respectively. Therefore, the witness does not need to sign the evidence message. If a malicious node tries to launch a DoS attack by broadcasting a bogus evidence message, the next integrity node receiving it can immediately detect the wicked behavior by verifying the signatures of $M_{\alpha 4\beta}$ and $M_{\alpha' 4\beta'}$ before forwarding to other nodes.

C. Security Discussions

Validity of Detection: The identity-based cryptographic system provides reliable identity verification and message authentication for the DHT-based protocol. As a result, the adversary cannot falsify cloned nodes' IDs; neither can the modify messages signed by integrity nodes. Moreover, a cloned node cannot lie to its observers about its location since a forged location would be far deviated from the communication range of the observers, which suffices to alert observers. Therefore, the detection guidelines are healthy provided observers are honest.

Thwarting Framing Attack*:*

A witness cannot forge an confirmation to frame integrity nodes since the evidence eventually is composed of claiming messages from different observers, and any node can verify them. However, for any witness-based discovery protocol, with our two proposals and protocols in Table I, there exists a realistic threat that some malicious observers try to frame innocent nodes by claiming wrong locations for them. To thwart this attack, we introduce a mechanism that requires nodes to buffer evidence messages they received and to maintain a debit table for those observers in When a node is declared as a replica in one or more evidence, assume one of its distinct locations is claimed by q different observers, then each of those observers should be debit by $\frac{1}{q}$. If a node's balance in the debit table exceed a threshold, it should be revoked as well. We advise one for the threshold. In this case, if a mean node tries to frame an integrity node by claim a false location for it, both nodes will be revoked. This one-exchanging-one strategy is quite fair and efficient as we do not need to distinguish which one is bad. When there is no frame attack in the network, integrity nodes are rarely revoke because a clone node's location may be claimed by many observers. At most, the number of revoke integrity nodes will not exceed the number of hateful nodes.

Protecting Witnesses*:*

Technically, the hash functions used in DHT do not need to be cryptographic hash functions. In practice, cryptographic ones are usually employed in the DHT systems because of their well uniformly random sharing of outputs and preventing potential abuse For our protocol, a cryptographic hash function is indeed required because it will hold back the adversary's abilities by application on $\mathrm{H}(\mathrm{seed} \,||\, \mathrm{id})$ as he cannot distinguish which nodes could more likely be witness before a round of detection. After discovery of the random seed, the challenger may want to comprise witnesses to defeat detection. However, there are $g + 1$ potential witnesses that are geographically randomly distributed in the network. formative and capturing all the witnesses will be troublesome for the opponent. Moreover, those witnesses vary round by rounds. Consequently, the opponent cannot stop the discovery by trying to capture a few witnesses.

Coping With Message-Discarding*:*

The cloned nodes may discard claiming messages through them. Our protocol is resilient against it due to the characteristic of full distributiveness and balance of the DHT-based protocol. If there are only a few cloned nodes, the

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

impact of this malicious action will be in significant. When the number of clone nodes increases, more claiming messages will assure sufficient number of witnesses. The simulations later clearly indicate this result. In summary, the DHT-based detection protocol is protected in the adversary model.

2.1 Performance Analysis Of Dht-Based Protocol
For the DHT-based detection protocol, we use the following specific measurements to evaluate its performance:
• *average number of transmitted messages*, representing the protocol's communication cost;
• *average size of node cache tables*, standing for the protocol's storage consumption;
• *average number of witnesses*, serving as the protocol's security level because the detection protocol is deterministic and symmetric.

A. Communication Cost

We denote the average path length between two random nodes by l, which varies from to $O(\log n)$ to $O(\sqrt{n})$, , dependent on underlying sensor networks. According to the Chord's properties, the average Chord-hop of a message that is, the number of transfers in the Chord overlay network is $c \log n$, where is a constant number, usually less than 1. Therefore, the average path hop length of a message is $cl \log n$. .There are $p_c dn$ claiming messages in total for a round of detection. Thus the average number of messages sent per node is given by $p_c dcl \log n$. .Since the , p_c, d, and c are constant, the asymptotic communication cost of the DHT-based protocol is between $O(\log^2 n)$ and $O(\sqrt{n} \log n)$.

B. Storage Consumption and Security Level

For simplicity, we hereby assume that all nodes, including compromise ones, abide by the detection protocol. Later in Section VI, we will see that the spiteful behaviors such as discarding claiming messages only slightly affect those performance measurements. In this detection protocol, claiming messages associated with a same examinee's ID will be transported to one destination node. Because there are n examinees and potential destination, and due to the good pseudo-randomness of the Chord system, on average, every node food one record in its cache table associated with one examinee's ID as its destination, regardless of the number of claim messages per examinee. In addition, for a designated examinee, the g predecessor nodes of the purpose can act as inspectors; thus, they probably hold up to one record related to the examinee. We consider an ideal case that there are m claiming messages for each examinee, and those m messages are independently transmit. Let p_r denote the probability of a predecessor receiving a specific claiming m message, then the probability of a ancestor holding a record for an examinee is $1-(1-p_r)^m$. Consequently, the average cache-table size can be calculated by

$$s = 1 + g(1-(1-p_r)^m).$$

If there are $\delta\ (\delta \geq 2)$ cloned nodes with a same ID in the network, then the destination node of claiming messages related to those nodes as examinees will deterministically become a witness to successfully catch the attack, while the g predecessor nodes of the destination may become witnesses if and only if they receive at least two claim messages associated with different cloned nodes. It is easy to see that this probability is minimized as $(1-(1-p_r)^m)^2$ when $\delta = 2$. . Therefore, as a lower bound for witness number, we only consider the case that there are two cloned nodes. In this case, them average witness number w can be obtained by

$$w = 1 + g(1-(1-p_r)^m)^2.$$

III. CLEAN-SLATE SENSOR NETWORK ROUTING

In this section, we show that a clean-slate secure sensor network routing protocol by Parno et al. ("PLGP" from here on) [53] can be modified to provably resist Vampire attacks during the packet forwarding phase. The original version of the protocol, although designed for security, is vulnerable to Vampire attacks. PLGP consists of a topology discovery phase, followed by a packet forwarding phase, with the former optionally repeated on a fixed schedule to ensure that topology information stays current. (There is no on demand discovery.) Discovery deterministically organizes nodes into a tree that will later be used as an addressing scheme. When discovery begins, each node has a limited view of the network the node knows only itself. Nodes discover their neighbors using local broadcast, and form ever growing “neighborhoods,” stopping when the entire network is a single group. Throughout this process, nodes build a tree of neighbor relationships and group member- ship that will later be used for addressing and routing.

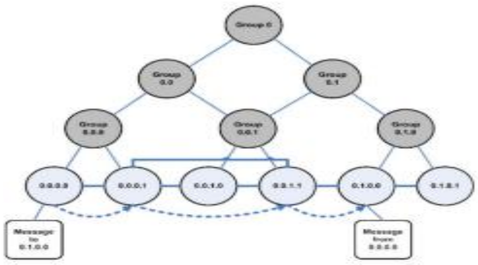


Fig. 6. The final address tree for a fully converged six-node network. Leaves represent physical nodes, connected with solid lines if within radio range. The dashed line is the progress of a message through the network. Note that non leaf nodes are not physical nodes but rather logical group identifiers.

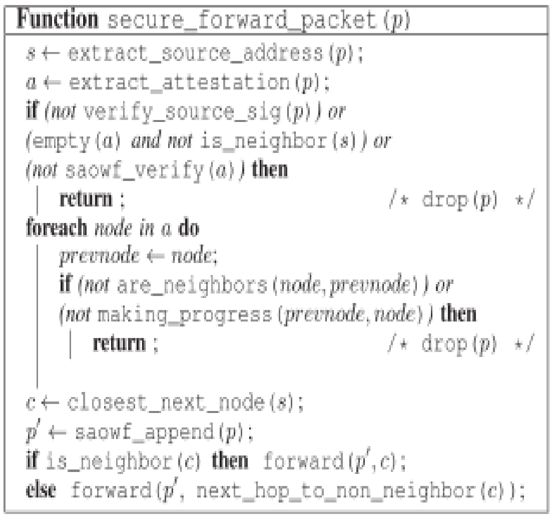
At the end of discovering, each node should estimate the same deal with shrub as other nodes. All foliage nodes in the shrub are actual nodes in the system, and their exclusive details return characters to their place in the shrub (see Fig. 6). All nodes learn each others’ exclusive details and cryptographic important factors. The ultimate deal with shrub is proven after system unity, and all sending choices can be individually confirmed. Furthermore, conceited each genuine system node has a exclusive certification of account (assigned before system deployment), nodes who make an effort to be a part of several categories, generate imitations of themselves in several places, or otherwise deceive during discovering can be recognized and removed. Topology discovering. Discovery begins with a moment restricted period during which every node must declare its existence by transmitting a certification of identification, such as its community key (from now on generally known as node ID), finalized by a reliable off-line power. Each node begins as its own number of dimension one, with a exclusive deal with 0. Nodes who overhear existence broadcasts type categories with their others who live nearby. When two personal nodes (each with an preliminary deal with 0) type a number of dimension two, one of them takes the deal with 0, and the other becomes 1. Groups combine preferentially with the tiniest nearby team, which may be only one node. We may think of categories performing as personal nodes, with choices created using protected multiparty computation. Like personal nodes, each team will originally select a team deal with 0, and will select 0 or 1 when consolidating with another team. Each team participant prepends the team deal with to their own deal with, e.g., node 0 in team 0 becomes 0.0, node 0 in team 1 becomes 1.0, and so on. Everytime two categories combine, the deal with of each node is prolonged by 1 bit. Unquestioningly, this types a binary shrub of all details in the system, with node details as leaved. Observe that this shrub is not a exclusive organize system,as he only details written by the shrub are next door neighbor connections among nodes. Nodes will demand to be a part of with the tiniest team in their area, with connections damaged by team IDs, which are calculated cooperatively by the whole team as a deterministic operate of personal participant IDs. When bigger categories combine, they both transmitted their team IDs (and the IDs of all team members) to each other, and continue with a combine method similar to the two-node case. Groups that have started large enough that some associates are not within stereo variety of other categories will connect through “gateway nodes,” which are within variety of both categories. Each node shops the identification of one or more nodes through which it observed an concept that another team prevails. That node may have itself observed the details second hand, so every node within a team will end up with a next-hop direction to every other team, as in variety vector. Topology discovering continues in this way until all system nodes are associates of only one team. By the end of topology discovering, each node understands every other node’s exclusive deal with, community key, and certification, since every team associates knows the details of all other team associates and the system converges to only one team. Bundle ahead During the sending stage, all choices are created individually by each node. When getting a packet, a node decides the next hop by discovering the most important bit of its deal with that varies from the concept originator’s deal with (see Fig. 6). Thus, every sending occasion (except when a packet is shifting within a team in order to achieve a entrance node to continue to the next group) reduce the sensible variety to the location, since node details should be totally nearer to the location (see Function ahead packet).

3.1 PLGP in the Presence of Vampires

In PLGP, forwarding nodes do not know what path a packet took, allowing adversaries to divert packets to any part of the network, even if that area is logically further away from the destination than the malicious node. This makes PLGP vulnerable to Vampire attacks. Consider forB instance the now-familiar directional antenna attack: a receiving honest node may be farther away from the packet destination than the mean forwarding node, but the honest node has no way to tell that the packet it just received is moving away from the destination; the only information available to the honest node is its own address and the packet destination address, but not the address of the previous hop (who can lie). Thus, the Vampire can move a packet away from its destination without being detected. This packet will traverse at most log N logical hops, with $O(\sqrt{2^i})$ Þ physical hops at the ith logical hop, giving us a theoretical maximum energy increase of $O(d)$, where d is the network diameter and N the number of network nodes. The situation is worse if the packet returns to the Vampire in the process of being forwarded—it can now be rerouted again, causing something similar to the carousel attack. Recall that the damage from the carousel attack is bounded by the maximum length of the source route, but in PLGP the adversary faces no such limitation, so the packet can cycle indefinitely. Nodes may sacrifice some local storage to retain a record of recent packets to prevent this attack from being carried out repeatedly with the same packet. Random direction vectors, as optional in PLGP, would likewise assuage the problem of indefinite cycles by avoiding the same spiteful node during the subsequent forwarding round.

IV. PROVABLE SECURITY AGAINST VAMPIRE ATTACKS

Here, we modify the forward phase of PLGP to provably avoid the above-mentioned attacks. First we introduce the no-backtracking property, satisfied for a given packet if and only if it consistently makes progress toward its destination in the logical network address space. More formally: Definition 1. No-backtracking is satisfied if every packet p traverses the same number of hops whether or not an adversary is present in the network. (Maliciously induced route stretch is bounded to a factor of 1.) This does not imply that every packet in the network must travel the same number of hops regardless of source or destination, but rather that a packet sent to node D by a malicious node at location L will traverse the same number of hops as a packet sent to D by a node at location L that is honest. If we think of this in terms of protocol execution traces, no-backtracking imply that for each packet in the trace, the number of intermediate honest nodes traversed by the packet between source and destination is independent of the actions of malicious nodes. Equivalently, traces that include malicious nodes should show the same network wide energy utilization by honest nodes as traces of a network with no malicious actors. The only notable exceptions are when adversaries drop or mangle packets enroute, but since we are only concerned with packets initiated by adversaries, we can safely ignore this situation: “premangled” packets achieve the same result—they will be dropped by an honest mediator or destination.


```
Function secure_forward_packet (p)
  s ← extract_source_address (p);
  a ← extract_attestation (p);
  if (not verify_source_sig (p)) or
  (empty (a) and not is_neighbor (s)) or
  (not saowf_verify (a)) then
      return ;                                  /* drop (p) */
  foreach node in a do
      prevnode ← node;
      if (not are_neighbors (node, prevnode)) or
      (not making_progress (prevnode, node)) then
          return ;                              /* drop (p) */

  c ← closest_next_node (s);
  p′ ← saowf_append (p);
  if is_neighbor (c) then forward (p′, c);
  else forward (p′, next_hop_to_non_neighbor (c));
```

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

No-backtracking implies Vampire resistance. It is not immediately obvious why no-backtracking prevents Vampire attacks in the forwarding phase. Recall the Reason for the success of the stretch attack: middle nodes in a source route cannot check whether the source-defined route is optimal, or even that it makes progress toward the destination. When nodes make independent routing decisions such as in link-state, distance-vector, coordinat based, or beacon-based protocols, packets cannot contain maliciously composed routes. This already means the adversary cannot perform carousel or stretch attacks no node may unilaterally specify a suboptimal path through the network. However, a sufficiently clever adversary may still influence packet progress. We can prevent this interference by independently checking on packet progress: if nodes keep track of route "cost" or metric and, when forwarding a packet, communicate the local cost to the next hop, that next hop can verify that the remaining route cost is lower than before, and therefore the packet is making progress toward its destination. (Otherwise we suspect malicious interference and drop the packet.) If we can guarantee that a packet is closer to its destination with every hop, we can bound the potential damage from an attacker as a function of network size. (A more desirable property is to guarantee good progress, such as logarithmic path length, but both allow us to obtain an upper bound on attack success.) PLGP does not satisfy no-backtracking. In non source routing protocols, routes are vigorously composed of forwarding decisions made independently by each node. PLGP differs from other protocols in that packets paths are further bounded by a tree, forwarding packets along the shortest route through the tree that is allowed by the physical topology. In other words, packet paths are constrained both by physical neighbor relationships and the routing tree. Since the tree implicitly mirrors the topology (two nodes have the same parent if and only if they are physical neighbors, and two nodes sharing an ancestor have a network path to each other), and since every node holds an identical copy of the address tree, every node can verify the optimal next logical hop. However, this is not sufficient for no-backtracking to hold, since nodes cannot be certain of the path previously traverse by a packet. Communicating a local view of route cost is not as easy as it seems, since adversary can always lie about their local metric, and so PLGP is still vulnerable to directional antenna/wormhole attacks, which allow adversaries to divert packets to any part of the network. To preserve no-backtracking, we add a verifiable path history to every PLGP packet, similar to route authentications in Ariadne [29] and path-vector signatures in [70]. The resulting protocol, PLGP with attestations (PLGPa) uses this packet history together with PLGP's tree routing structure so every node can securely verify progress, preventing any significant adversarial influence on the path taken by any packet which traverses at least one honest node. Whenever node n forwards packet p, it this by attaching a non replayable attestation (signature). These signatures form a chain attached to every packet, allowing any node receiving it to validate its path. Every forwarding node verifies the attestation chain to ensure that the packet has never traveled away from its destination in the logical deal with space. See Function secure forward packet for the modified protocol.

PLGPa satisfies no-backtracking.
To demonstrate that our customized method maintains the no-backtracking residence, we determine a system as a selection of nodes, a topology, connection qualities, and node details, credit the design used by Poturalski et al. in [57]. Sincere nodes can transmitted and get information, while harmful nodes can also use online antennas to deliver to (or get from) any node in the system without being overheard by any other node. Sincere nodes can write, ahead, agree to, or fall information, and harmful nodes can also randomly convert them. Our attacker is believed to management m nodes in an N-node system (with their corresponding identification accreditations and other key cryptographic material) and has ideal information of the system topology. Lastly, the opposition cannot impact connection between any two honest nodes. Since all information are finalized by their founder, information from honest nodes cannot be randomly customized by harmful nodes wanting to stay unnoticed. Rather, the attacker can only modify bundle areas that are modified en direction (and so are not authenticated), so only the direction attestation area can be modified, reduced, or eliminated entirely. To avoid truncation, which would allow Skeletons to cover up the point that they are shifting a bundle away from its location, we use Saxena and Soh's one-way trademark sequence developing [64], which allow nodes to add hyperlinks to an current trademark sequence, but not eliminate hyperlinks, creating attestations add only. For the reasons of Creature of the night strikes, we are aren't bothered about packages with irrelavent hop matters that are never obtained by honest nodes but rather are directed between attacker only, so we determine the hop depend of a bundle as follows:

Definition 2. The hop depend of bundle p, obtained or submitted by a genuine node, is no higher than the variety of records in p's direction attestation area, plus 1. When any node gets a concept, it assessments that every node in the direction attestation 1) has a corresponding access in the trademark sequence, and 2) is rationally nearer to the location

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

than the past opinthe chain(see Operate protected ahead _packet). This way, sending nodes can implement the ahead improvement of a concept, protecting no-backtracking. If no attestation is existing, The node assessments to see if the founder of the concept is a actual other resident. Since information are finalized with the originator's key, harmful nodes cannot incorrectly declare to be the source of a concept, and therefore do not advantage by eliminating attestations.

Theorem 1. A PLGPa bundle p meets no-backtracking in the existence of an attacker managing m<N_ 3 nodes if p goes through at least one honest node.

Proof. Consider two irrelavent PLGPa method records H and M of the same N-node system, in which node S delivers bundle p to node D. Restrict H such that all nodes are sincere, and constrain M such that m<N_ 3 are harmful. Let p achieve an irrelavent sincere node I along the protocol-defined bundle direction in h trips in H, but in h trips for _>0 in M (no-backtracking is not pleased in the latter). Since PLGPa is deterministic, the distinction _ must be because of a harmful node. Further, since the hop depend of p when it comes at I is higher in M than in H, s direction statement sequence must be more time in M. Remember that every node has a exclusive exclusive cope with, and no bundle may be submitted between any two nodes without shifting either in reverse or ahead through the exclusive cope with area, so p must have shifted in reverse in the organize area by at least one hop.9 Consider the following three scenarios: 1) I is a next door neighbor of S and the next hop of p;2)I is a next door neighbor of D and the last hop of p before the destination; and 3) I is a sending node of the bundle, but is neither a next door neighbor of S nor D.IfI delivers a bundle with h, trips in its direction attestation, the attacker must have been successful in at least one of the following: Consider the following three scenarios: 1) I is a next door neighbor of S and the next hop of p;2)I is a next door neighbor of D and the last hop of p before the destination; and 3) I is a ahead node of the bundle, but is neither a next door neighbor of S nor D.IfI delivers a bundle with h trips in its direction attestation, the attacker must have been successful in at least one of the following: resulting in sincere node I to ahead p with non zero attestation, over a direction that backtracked, breaking the rumours that sincere nodes properly adhere to PLGPa;10 . resulting in sincere node I to ahead p with a nonnull attestation, from resource S who is I's immediate next door neighbor, breach the rumours that sincere nodes properly adhere to PLGPa . truncating the direction attestation, breaking the security of sequence signatures. Lastly, if I delivers p with a zero attestation, it is either a next door neighbor of S or the attacker has damaged the trademark plan used by the emailer to verify the packet's invariant fields—an sincere would not ahead a bundle with no attestation if the bundle resource is not a next door neighbor. Since each possible adversarial activity which outcomes in backtracking goes against an rumours, the evidence is finish. Since no-backtracking assures bundle improvement, and PLGPa maintains no-backtracking, it is the only method mentioned so far that provably range the rate of power used in the adversarial condition to that used with only sincere nodes to 1, and by the meaning of no-backtracking PLGPa avoids Creature of the night strikes. This is obtained because bundle improvement is safely proven. Observe that we cannot assurance that a bundle will achieve its location, since it can always be decreased. Assured distribution is beyond the opportunity of this document. In totally required no-backtracking, topology changes that may remove all protocol-level routes to a node that do not need backtracking, even though network-level routes still are available (e.g., the GPSR "dead end" scenario). To cope with such circumstances we can allow for limited backtracking (_-backtracking, in contrast to our exclusive 0-backtracking scheme), which provides some flexibility in the way no-backtracking is confirmed, enabling a certain quantity of finish backtracking per bundle within the protection parameter. The extensive protection evidence by introduction onis simple.

V. SECURING THE DISCOVERY PHASE

Without fully solving the problem of malicious topology discovery, we can still mitigate it by forcing synchronous discovery and ignoring discovery messages during the intervening periods. This can lead to some nodes being separated from the network for a period of time, and is essentially a form of rate limiting. Although we rejected rate limiting before, it is acceptable here since discovery should consume a small fraction of running time compared to packet forwarding. We can enforce rate limits in a number of ways, such as neighbor throttling [35] or one-way hash chains [14]. We can also optimize discovery algorithms [32] to minimize our window of vulnerability. If a network survives the highrisk discovery period, it is improbable to suffer serious damage from Vampires during normal packet forwarding. While PLGPa is not vulnerable to Vampire attacks during the forwarding phase, we cannot make the same claim about discovery. However, we can give some intuition as to how to further modify PLGPa to bound damage from malicious discovery. (The value of that bound in practice remains an open problem.) The major issue is that malicious nodes can use directional antennas to masquerade neighbors to any or all nodes in the network, and therefore look like a group of size one, with which other groups will try to preferentially merge. Merge requests are composed of the requested group's ID as well as all the group members' IDs, and the receiving node will flood this request to other group members. Even assuming groups generate signed tokensthat cost no energy to verify, a Vampire would be able to flood its group with every group descriptor it knows, and use its directional antenna to snoop on broadcasts outside their neighbor range, relaying merge requests from entirely

honest groups. Since each Vampire will start as a group of one, other groups will issue merge requests, which the Vampire can deny. In PLGP, denials are only allowed if another merge is in progress, so if we modify the decline message to include the ID of the group with which the merge is in progress (and a signature for non repudiation), these messages can be kept and replayed at the end of the topology discovery period, detecting and removing nodes who incorrectly deny merge requests. Therefore, Vampires reject legitimate merge requests at their own peril. Any group containing a Vampire can be made to serially join with a "group" composed only of each Vampire in the network (all of them would have to advertise themselves as neighbors of each group). Even wholly honest groups can be fooled using directional antennas: Vampires could maintain the illusion that it is a neighbor of a given group. Since join events require multiparty calculation and are flooded throughout the group, this makes for a fairly effective attack. PLGP already provides for the discovery of such subterfuge upon termination of topology discovery: a node who is a member of multiple groups will be detected once those groups join (and all groups are guaranteed to merge by the end of the protocol). Since PLGP offers the chance to detect active Vampires once the network converges, successive rediscovery periods become safer. This is more than can be said of other protocols, where malicious behavior during discovery may go undetected, or at least scot-free. However, the bound we can place on malicious discovery damage in PLGPa is still unknown. Moreover, if we can conclude that a single malicious node causes a factor of k energy increase during discovery (and is then expelled), it is not clear how that value scales under collusion among multiple mal- icious nodes.

VI. CONCLUSION

In this paper, we described Creature of the night strikes, a new class of resource consumption strikes that use redirecting methods to lastingly turn off ad hoc wireless indicator systems by burning nodes' battery pack. These strikes do not rely on particular methods or implementations, but rather reveal weaknesses in a variety of popular method sessions. We revealed a variety of proof-of-concept strikes against commissioner illustrations of current redirecting methods using some poor opponents, and calculated their attack success on a arbitrarily generated topology of 30 nodes. replica results show that based on the location of the attacker, system power expenses during the sending stage improves from between 50 to 1,000 percent. theoretical most severe power utilization can increase by as much as a factor of per attacker per bundle, where N is the system size. We recommended protection against some of the forwarding-phase strikes and described PLGPa, the first indicator system redirecting method that provably range harm from Creature of the night strikes by confirming that packages consistently make progress toward their locations. We have not offered a fully acceptable solution for Creature of the night strikes during the topology finding stage, but recommended some instinct about harm restrictions possible with further variations to PLGPa. main of harm range and protection for topology finding, as well as managing mobile systems, is left for future work.

REFERENCES

[1] "The Network Simulator - ns-2," http://www.isi.edu/nsnam/ns, 2012.
[2] I. Aad, J.-P. Hubaux, and E.W. Knightly, "Denial of Service
Resilience in Ad Hoc Networks," Proc. ACM MobiCom, 2004.
[3] G. Acs, L. Buttyan, and I. Vajda, "Provably Secure On-Demand Source Routing in Mobile Ad Hoc Networks," IEEE Trans. Mobile Computing, vol. 5, no. 11, pp. 1533-1546, Nov. 2006.
[4] T. Aura, "Dos-Resistant Authentication with Client Puzzles," Proc. Int'l Workshop Security Protocols, 2001.
[5] J. Bellardo and S. Savage, "802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions," Proc. 12th Conf. USENIXSecurity, 2003.
[6] D. Bernstein and P. Schwabe, "New AES Software Speed Records," Proc. Ninth Int'l Conf. Cryptology in India: Progress in Cryptology (INDOCRYPT), 2008.
[7] D.J. Bernstein, "Syn Cookies," http://cr.yp.to/syncookies.html,1996.

[8] I.F. Blaked, G. Seroussi, and N.P. Smart, Elliptic Curves in Cryptography, vol. 265. Cambridge Univ. , 1999.
[9] J.W. Bos, D.A. Osvik, and D. Stefan, “Fast Implementations of AES on Various Platforms,” Cryptology ePrint Archive, Report 2009/ 501, http://eprint.iacr.org, 2009.
[10] H. Chan and A. Perrig, “Security and Privacy in Sensor Networks,” Computer, vol. 36, no. 10, pp. 103-105, Oct. 2003.
[11] J.-H. Chang and L. Tassiulas, “Maximum Lifetime Routing in Wireless Sensor Networks,” IEEE/ACM Trans. Networking, vol. 12, no. 4, pp. 609-619, Aug. 2004.
[12] T.H. Clausen and P. Jacquet, Optimized Link State Routing Protocol (OLSR), IETF RFC 3626, 2003.
[13] J. Deng, R. Han, and S. Mishra, “Defending against Path-Based DoS Attacks in Wireless Sensor Networks,” Proc. ACM Workshop Security of Ad Hoc and Sensor Networks, 2005.
[14] J. Deng, R. Han, and S. Mishra, “INSENS: Intrusion-Tolerant
Routing for Wireless Sensor Networks,” Computer Comm., vol. 29, no. 2, pp. 216-230, 2006.
[15] S. Doshi, S. Bhandare, and T.X. Brown, “An On-Demand Minimum Energy Routing Protocol for a Wireless Ad Hoc Network,” ACM SIGMOBILE Mobile Computing and Comm. Rev., vol. 6, no. 3, pp. 50-66, 2002.
[16] J.R. Douceur, “The Sybil Attack,” Proc. Int’l Workshop Peer-to-Peer
Systems, 2002.
[17] H. Eberle, A. Wander, N. Gura, C.-S. Sheueling, and V. Gupta, “Architectural Extensions for Elliptic Curve Cryptography over GF(2m) on 8-bit Microprocessors,” Proc. IEEE Int’l Conf’ Application-Specificb Systems, Architecture Processors (ASAP), 2005.
[18] T. English, M. Keller, K.L. Man, E. Popovici, M. Schellekens, and W. Marnane, “A Low-Power Pairing-Based Cryptographic Accelerator for Embedded Security Applications,” Proc. IEEE Int’l SOC Conf. , 2009.
[19] L.M. Feeney, “An Energy Consumption Model for Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks,” Mobile Networks and Applications, vol. 6, no. 3, pp. 239-249, 2001.
[20] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, “Strong Authentication for RFID Systems Using the AES Algorithm,”
Proc. Int’l Workshop Cryptographic Hardware and Embedded Systems
(CHES), 2004.
[21] R. Fonseca, S. Ratnasamy, J. Zhao, C.T. Ee, D. Culler, S. Shenker, and I. Stoica, “Beacon Vector Routing: Scalable Point-to-Point Routing in Wireless Sensornets,” Proc. Second Conf. Symp. Networked Systems Design & Implementation (NSDI), 2005.
[22] S. Galbraith, K. Harrison, and D. Soldera, “Implementing the Tate Pairing,” Proc. Int’l Symp. Algorithmic Number Theory, 2002.
[23] S. Goldberg, D. Xiao, E. Tromer, B. Barak, and J. Rexford, “PathQuality Monitoring in the Presence of Adversaries,” Proc. ACM SIGMETRICS Int’l Conf. Measurement and Modeling of Computer Systems, 2008.
[24] A.J. Goldsmith and S.B. Wicker, “Design Challenges for EnergyConstrained
Ad Hoc Wireless Networks,” IEEE Wireless Comm., vol. 9, no. 4, pp. 8-27, Aug. 2002.
[25] R. Govindan and A. Reddy, “An Analysis of Internet InterDomain Topology and Route Stability,” Proc. IEEE INFOCOM, 1997.
[26] M. Guirguis, A. Bestavros, I. Matta, and Y. Zhang, “Reduction of Quality (RoQ) Attacks on Internet End-Systems,” Proc. IEEE INFOCOM, 2005.
[27] J.L. Hill and D.E. Culler, “Mica: A Wireless Platform for Deeply Embedded Networks,” IEEE Micro, vol. 22, no. 6, pp. 12-24, Nov./ Dec. 2002.
[28] Y.-C. Hu, D.B. Johnson, and A. Perrig, “SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks,” Proc. IEEE Workshop Mobile Computing Systems and Applications,
2002.
[29] Y.-C. Hu, D.B. Johnson, and A. Perrig, “Ariadne: A Secure OnDemand Routing Protocol for Ad Hoc Networks,” Proc. MobiCom, 2002.
[30] Y.-C. Hu, D.B. Johnson, and A. Perrig, “Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks,” Proc. IEEE INFOCOM, 2003.
[31] Y.-C. Hu, D.B. Johnson, and A. Perrig, “Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols,” Proc.
Second ACM Workshop Wireless Security (WiSE), 2003.
[32] Y. Huang and S. Bhatti, “Fast-Converging Distance Vector Routing for Wireless Mesh Networks,” Proc. 28th Int’l Conf.
Distributed Computing Systems Workshops (ICDCSW), 2008.
[33] D. Hwang, B.-C. Lai, P. Schaumont, K. Sakiyama, Y. Fan, S. Yang, A. Hodjat, and I. Verbauwhede, “Design Flow for HW/SW
Acceleration Transparency in the Thumbpod Secure Embedded System,” Proc. Design Automation Conf., 2003.
[34] L. Iannone, R. Khalili, K. Salamatian, and S. Fdida, “Cross-Layer Routing in Wireless Mesh Networks,” Proc. Int’l Symp. Wireless Comm. Systems, 2004.
[35] D.B. Johnson, D.A. Maltz, and J. Broch, “DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks,” Ad Hoc Networking, Addison-Wesley, 2001.
[36] C. Karlof and D. Wagner, “Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures,” Proc. IEEE Int’l Workshop Sensor Network Protocols and Applications, 2003.
[37] B. Karp and H.T. Kung, “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks,” Proc. ACM MobiCom, 2000.
[38] Y. Kawahara, T. Takagi, and E. Okamoto, “Efficient Implementation of Tate Pairing on a Mobile Phone Using Java,” Proc. Int’l Conf. Computational Intelligence and Security, 2006.
[39] M. Koschuch, J. Lechner, A. Weitzer, J. Groschdl, A. Szekely, S. Tillich, and J. Wolkerstorfer, “Hardware/Software Co-Design of Elliptic Curve Cryptography on an 8051 Microcontroller,” Proc. Eighth Int’l Conf. Cryptographic Hardware and Embedded Systems (CHES), 2006.
[40] A. Kroller, S.P. Fekete, D. Pfisterer, and S. Fischer, “Deterministic Boundary Recognition and Topology Extraction for Large Sensor Networks,” Proc. Ann. ACM-SIAM Symp. Discrete Algorithms, 2006.
[41] A. Kuzmanovic and E.W. Knightly, “Low-Rate TCP-Targeted
Denial of Service Attacks: The Shrew vs. the Mice and Elephants,” Proc. SIGCOMM, 2003.
[42] Y.-K. Kwok, R. Tripathi, Y. Chen, and K. Hwang, “HAWK: Halting Anomalies with Weighted Choking to Rescue WellBehaved TCP Sessions from Shrew DDoS Attacks,” Proc. Int’l Conf. Networking and Mobile Computing, 2005.
[43] L. Xiaojun, N.B. Shroff, and R. Srikant, “A Tutorial on Cross-Layer Optimization in Wireless Networks,” IEEE J. Selected Areas in Comm., vol. 24, no. 8, pp. 1452-1463, Aug. 2006.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

[44] X. Luo and R.K.C. Chang, "On a New Class of Pulsing Denial-ofService Attacks and the Defense," Proc. Network and Distributed System Security Symp. (NDSS), 2005.
[45] M. Maleki, K. Dantu, and M. Pedram, "Power-Aware Source Routing Protocol for Mobile Ad Hoc Networks," Proc. Int'l Symp. Low Power Electronics and Design (ISLPED), 2002.
[46] Y. Matsuoka, P. Schaumont, K. Tiri, and I. Verbauwhede, "Java Cryptography on KVM and Its Performance and Security Optimization Using HW/SW Co-Design Techniques," Proc. Int'l Conf. Compilers, Architecture, and Synthesis for Embedded Systems (CASES), 2004.
[47] M. McLoone and M. Robshaw, "Public Key Cryptography and RFID Tags," Proc. RSA Conf. Cryptography (CT-RSA), 2006.
[48] T.J. McNevin, J.-M. Park, and R. Marchany, "pTCP: A Client Puzzle Protocol for Defending Against Resource Exhaustion
Denial of Service Attacks," Technical Report TR-ECE-04-10, Dept. of Electrical and Computer Eng., Virginia Tech, 2004.
[49] V.P. Nambiar, M. Khalil-Hani, and M.M.A. Zabidi, "Accelerating the AES Encryption Function in OpenSSL for Embedded Systems," Proc. Int'l Conf Electrical Design (ICED), 2008.
[50] A. Nasipuri and S.R. Das, "On-Demand Multipath Routing for Mobile Ad Hoc Networks," Proc. Int'l Conf. Computer Comm. And Networks, 1999.
[51] L.B. Oliveira, D.F. Aranha, E. Morais, F. Daguano, J. Lopez, and R. Dahab, "TinyTate: Computing the Tate Pairing in ResourceConstrained Sensor Nodes," Proc. IEEE Sixth Int'l Symp. Network Computing and Applications (NCA), 2007.
[52] K. Park and H. Lee, "On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack," Proc. IEEE INFOCOM, 2001.
[53] B. Parno, M. Luk, E. Gaustad, and A. Perrig, "Secure Sensor Network Routing: A Clean-Slate Approach," CoNEXT: Proc. ACM CoNEXT Conf., 2006.
[54] V. Paxson, "An Analysis of Using Reflectors for Distributed Denial-of-Service Attacks," SIGCOMM Computing Comm. Rev., vol. 31, no. 3, pp. 38-47, 2001.
[55] C.E. Perkins and P. Bhagwat, "Highly Dynamic Destination Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," Proc. Conf. Comm. Architectures, Protocols and Applications, 1994.
[56] R. Potlapally, S. Ravi, A. Raghunathan, R.B. Lee, and N.K. Jha, "Impact of Configurability and Extensibility on IPSec Protocol Execution on Embedded Processors," Proc. Int'l Conf. VLSI Design, 2006.
[57] M. Poturalski, P. Papadimitratos, and J.-P. Hubaux, "Secure Neighbor Discovery in Wireless Networks: Formal Investigation
of Possibility," Proc. ACM Symp. Information, Computer and Comm. Security (ASIACCS), 2008.
[58] D. Raffo, C. Adjih, T. Clausen, and P. Mu
¨hlethaler, "An Advanced Signature System for OLSR," Proc. Second ACM Workshop Security of Ad Hoc and Sensor Networks (SASN), 2004. 332 IEEE TRANSACTIONS ON MOBILE COMPUTING, VOL. 12, NO. 2, FEBRUARY 2013
[59] D.R. Raymond, R.C. Marchany, M.I. Brownfield, and S.F. Midkiff, "Effects of Denial-of-Sleep Attacks on Wireless Sensor Network MAC Protocols," IEEE Trans. Vehicular Technology, vol. 58, no. 1, pp. 367-380, Jan. 2009.
[60] D.R. Raymond and S.F. Midkiff, "Denial-of-Service in Wireless Sensor Networks: Attacks and Defenses," IEEE Pervasive Computing, vol. 7, no. 1, pp. 74-81, Jan.-Mar. 2008.
[61] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang, "BGP Routing Stability of Popular Destinations," Proc. Second ACM SIGCOMM Workshop Internet Measurement (IMW), 2002.
[62] R.L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," Comm. ACM, vol. 21, no. 2, pp. 120-126, 1978.
[63] V. Rodoplu and T.H. Meng, "Minimum Energy Mobile Wireless Networks," IEEE J. Selected Areas in Comm., vol. 17, no. 8, pp. 13331344, Aug. 1999.
[64] A. Saxena and B. Soh, "One-Way Signature Chaining: A New Paradigm for Group Cryptosystems," Int'l J. Information and Computer Security, vol. 2, no. 3, pp. 268-296, 2008.
[65] M. Scott, N. Costigan, and W. Abdulwahab, "Implementing Cryptographic Pairings on Smartcards," Proc. Eighth Int'l Conf. Cryptographic Hardware and Embedded Systems (CHES), 2006.
[66] R.C. Shah and J.M. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks," Proc. IEEE Wireless Comm. and
Network Conf. (WCNC), 2002.
[67] S. Singh, M. Woo, and C.S. Raghavendra, "Power-Aware Routing in Mobile Ad Hoc Networks," Proc. ACM MobiCom, 1998.
[68] F. Stajano and R. Anderson, "The Resurrecting Duckling: Security Issues for Ad-Hoc Wireless Networks," Proc. Int'l Workshop Security Protocols, 1999.
[69] I. Stojmenovic and X. Lin, "Power-Aware Localized Routing in Wireless Networks," IEEE Trans. Parallel and Distributed Systems, vol. 12, no. 11, pp. 1122-1133, Nov. 2001.
[70] L. Subramanian, R.H. Katz, V. Roth, S. Shenker, and I. Stoica, "Reliable Broadcast in Unknown Fixed-Identity Networks," Proc. Ann. ACM SIGACT-SIGOPS Symp. Principles of Distributed Computing, 2005.
[71] H. Sun, J.C.S. Lui, and D.K.Y. Yau, "Defending against Low-Rate TCP Attacks: Dynamic Detection and Protection," Proc. IEEE 12th Int'l Conf. Network Protocols (ICNP), 2004.
[72] C. Villamizar, R. Chandra, and R. Govindan, BGP Route Flap Damping, IETF RFC 2439, 1998.
[73] L. von Ahn, M. Blum, N.J. Hopper, and J. Langford, "CAPTCHA: Using Hard AI Problems for Security," Proc. 22nd Int'l Conf. Theory and Applications of Cryptographic Techniques (Eurocrypt), 2003.
[74] Y. Wang, J. Gao, and J.S.B. Mitchell, "Boundary Recognition in Sensor Networks by Topological Methods," Proc. ACM MobiCom, 2006.
[75] A.D. Wood and J.A. Stankovic, "Denial of Service in Sensor Networks," Computer, vol. 35, no. 10, pp. 54-62, Oct. 2002.
[76] G. Yang, M. Gerla, and M.Y. Sanadidi, "Defense Against Low-Rate TCP-Targeted Denial-of-Service Attacks," Proc. Ninth Int'l Symp. Computers and Comm. (ISCC),
2004.
[77] J. Yuan, Z. Li, W. Yu, and B. Li, "A Cross-Layer Optimization Framework for Multihop Multicast in Wireless Mesh Networks," IEEE J. Selected Areas in Comm., vol. 24, no. 11, pp. 2092-2103, Nov.2006.
[78] M.G. Zapata and N. Asokan, "Securing Ad Hoc Routing Protocols," Proc. First ACM Workshop Wireless Security (WiSE), 2002.