ISSN:0975-9646

Wolf Routing to Detect Vampire Attacks in Wireless Sensor Networks

Prof. Besty Haris
Dept of Computer Science, MVJCE

Abstract: Ad-hoc low-power wireless networks are a high price research direction in sensing and pervasive computing. Prior security work in this area has focused primarily on denial of communication at the routing or medium access control levels. This paper proposes a scheme to detect resource depletion attacks, called Vampire Attacks at the routing protocol layer, which permanently disable networks by quickly draining nodes' battery power. The scheme is based on the preying behaviour of wolves. These "Vampire" attacks are not specific to any specific protocol, but rather rely on the properties of many popular classes of routing protocols. Most of the general protocols are susceptible to Vampire attacks, which are devastating, difficult to detect, and are easy to carry out using as few as one malicious insider sending only protocol compliant messages. In the worst case, a single Vampire can increase network-wide energy usage by a factor of O(N), where N in the number of network nodes. In this paper the author discusses about a bio-inspired Vampire attack detection method in Wireless Sensor Network using Wolf-Routing Algorithm.

1. INTRODUCTION:

Wireless ad hoc sensor network is a configuration for area surveillance that affords rapid, flexible deployment in arbitrary threat environments, e.g., battlefield spaces or enterprise premises. Such a network is depicted in Fig. 1. With no infrastructure support, sensor nodes communicate with each other only when they are within wireless transmission range.

The nodes are typically unattended and severely resource restricted, with limited processing, memory, and power capacities. They operate cooperatively to process and fuse sensor data into information to fulfil the surveillance mission. In a wireless sensor network (WSN), each sensor at a node observes physical phenomena in its sensing range. Node processing quantizes and combines, or fuses, the observations to produce aggregate information; processing occurs along an intermediate sequence of wirelessly linked nodes that ultimately reaches the sink (destination) node

Wireless sensor networks (WSN) are networks of small resource constrained devices which sense the environment and report the results via wireless networks. They allow spatial or temporal measurements of phenomenon previously difficult to analyse [3]. One of the current challenges in the WSN field is the development of management systems which allow WSN to be easily deployed in various application domains as different WSN application domains often have different management requirements. However given that WSN are very restricted in terms of resources and usually battery powered, overheads involving communication are to be avoided as much as possible. Such a network is highly vulnerable to Vampire Attack, which are not protocol specific and which does not use the loop holes in the routing protocols. Therefore lightweight decentralised management solutions are favoured.

Heuristic optimization methods have an edge over their classical counterparts because they can incrementally induce a globally optimum solution by using heuristics to efficiently search a large space. A special kind of heuristic optimization known as nature-inspired optimization or metaheuristics is gaining substantial popularity in the research community due to its advantages, which are applicable in computational intelligence, data-mining [10] and their applications. For instance, clustering integrated with nature-inspired optimization produces improved performance [4]. Borrowed from the wonders of nature, such algorithms computationally optimize complex search problems with superior performance and search efficiency compared to earlier optimization techniques. The Wolf Search Algorithm (WSA), imitates the preying behaviour of wolves and has displayed unique advantages in efficiency because each searching agent simultaneously performs autonomous solution searching and merging. Local optima are overcome when the searching agents leap far away upon being triggered by the random emergence of an enemy. WSA is tested against classical algorithms such as GA, PSO, ACO, and it outperformed GA and PSO in most of the testing cases, and beats ACO in the convergence test.

Wolf Search Algorithm (WSA), which is based on wolf preying behaviour can be used for searching for the vampire node(s). WSA is different from the aforementioned bio-inspired metaheuristics because it simultaneously possesses both individual local searching ability and autonomous flocking movement. In other words, each wolf in WSA hunts independently by remembering its own trait and only merges with its peer when the peer is in a better position. In this way, long-range inter-communication among the wolves that represent the searching points for candidate solutions is eliminated because wolves are known to stalk their prey in silence. Assembly depends on visual range. Therefore, the swarming behaviour of WSA, unlike most bio-inspired algorithms, is delegated to each individual wolf rather than to a single leader, as in PSO [4], Fish [6] and Firefly [2].

Effectively, WSA functions as if there are multiple leaders swarming from multiple directions to the best solution, rather than a single flock that searches for an optimum in one direction at a time. The appearance of a hunter that corresponds to each wolf is added at random and on meeting its hunter, each wolf jumps far out of its hunter's visual range to avoid being trapped in local optima by the algorithm's design.

2. Related Work

Traditional intrusion detection systems require comprehensive, up-to-date knowledge bases, limiting their response to the continuous evolution of attacks against WSNs. While encryption and authentication mechanisms may prevent intrusions in non-traditional intrusion detection systems, they cannot totally eliminate them, especially for intrusions initiated within the network. The latter two mechanisms require a great deal of communication overhead that becomes an implementation issue in a resource-limited WSN. Y. Huang, et al., presented an anomaly detection approach using cross feature analysis for wireless ad hoc networks, based on strong inter-feature correlations that exist in normal traffic [1]. This observation motivates much of the cross-layer design for performance enhancements in WSNs. Published studies have applied one bio-inspired/ evolutionary computational method to the functions of a single protocol layer of the OSI stack to detect/identify compromised nodes in a wireless network [3] – [6]. These singular applications motivate use in cross-layer design, wherein a different evolutionary method is injected at each of the physical (PHY), medium access control (MAC), network, and application layers to identify and purge false data caused by the malicious behaviour of compromised nodes.

3. Description of Vampire Attacks:

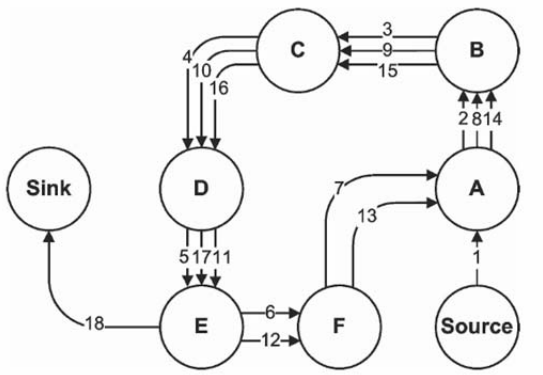
Vampire attack happens in the network in the sense, any of the nodes in the network which is affected or infected and this nodes behaviour is abruptly changing for the network behaviour, this kind of nodes are called
"Malicious node". If malicious nodes present in the network energy that have been using by each and every nodes will increases drastically. The malicious nodes has been place in the network uniquely. First In between the routing nodes, and the second placed in the Source node itself. The chance of placing a malicious node in the routing path this makes causing damage in network. Source node identifying the particular packets and selected packets are identified for the routing to the destination.

The routing path is discovered by source node by using shortest path routing algorithm and the path shouldn't be changeable by the intermediate nodes. In this type of occasion there is a chance to happening attack. The adversary composes packets with purposely introduced routing loops. This is one of the major problem of the network where the consuming energy of each and every nodes in the network will be increasing. Since it sends packets in circle.it targets source routing protocols by exploiting the limited verification of message heads at forwarding nodes, allowing single packets to repeatedly traverse the same set of nodes. This process continues for the particular period of time, transmitting the process in the loop and wasting every nodes power which is presently in the routing path. The main problem these kind of attackers are it is not easily identified if it attacked or affected the network.it will take some long time to identify and make ensure that it presented in the network.

Even in non-power-constrained systems, depletion of resources such as memory, CPU time, and bandwidth may easily cause problems. A popular example is the SYN flood attack, wherein adversaries make multiple connection requests to a server, which will allocate resources for each connection request, eventually running out of resources, while the adversary, who allocates minimal resources, remains operational (since he does not intend to ever complete the connection handshake). Such attacks can be defeated or attenuated by putting greater burden on the connecting entity (e.g. SYN cookies, which offload the initial connection state onto the client, or cryptographic puzzles). These solutions place minimal load on legitimate clients who only initiate a small number of connections, but deter malicious entities who will attempt a large number. Note that this is actually a form of rate limiting, and not always desirable as it punishes nodes who produce bursty traffic but may not send much total data over the lifetime of the network. Since Vampire attacks rely on amplification, such solutions may not be sufficiently effective to justify the excess load on legitimate nodes.

3.1 Attacks on Stateless Protocols

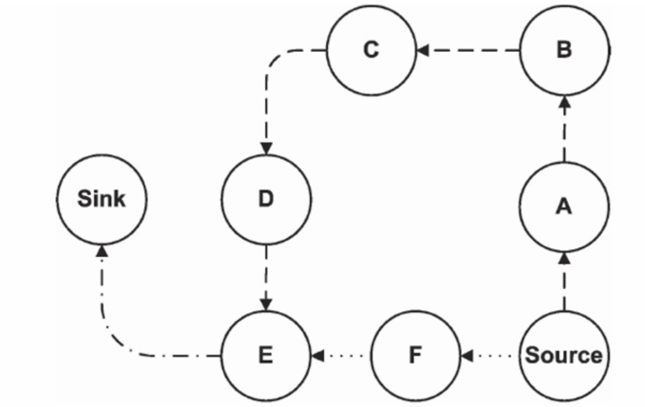
3.1.1 Carousel attack. In this attack, an adversary sends a packet with a route composed as a series of loops, such that the same node appears in the route many times. This strategy can be used to increase the route length beyond the number of nodes in the network, only limited by the number of allowed entries in the source route.



(a) An honest route would exit the loop immediately from node E to Sink, but a malicious packet makes its way around the loop twice more before exiting.

An example of this type of route is in Figure 1(a). malicious node carries out a carousel attack, sending a single message to node 19 (which does not have to be malicious). Note the drastic increase in energy usage along the original path.3 Assuming the adversary limits the transmission rate to avoid saturating the network, the theoretical limit of this attack is an energy usage increase factor of O(λ), where λ is the maximum route length

3.1.2 Stretch Attack: Another attack in the same vein is the stretch attack, where a malicious node constructs artificially long source routes, causing packets to traverse a larger than optimal number of nodes. An honest source would select the route Source → F → E → Sink, affecting four nodes including itself, but the malicious node selects a longer route, affecting all nodes in the network. These routes cause nodes that do not lie along the honest route to consume energy by forwarding packets they would not receive in honest scenarios. An example of this type of route is in Figure 1(b). The outcome becomes clearer when we examine Figure 3(c) and compare to the carousel attack. While the latter uses energy at the nodes who were already in the honest path, the former extends the consumed energy "equivalence lines" to a wider section of the network.



(b) Honest route is dotted while malicious route is dashed. The last link to the sink is shared.

Energy usage is less localized around the original path, but more total energy is consumed. The theoretical limit of the stretch attack is a packet that traverses every network node, causing an energy usage increase of factor O(min(N, λ)), where N is the number of nodes in the network and λ is the maximum path length allowed. This attack is potentially less damaging per packet than the carousel attack, as the number of hops per packet is bounded by the number of network nodes. However, adversaries can combine carousel and stretch attacks to keep the packet in the network longer: the resulting "stretched cycle" could be traversed repeatedly in a loop

3.2 Attacks on stateful protocols

Routes in link-state and distance-vector networks are built dynamically from many independent forwarding decisions, so adversaries have limited power to affect packet forwarding, making these protocols immune to carousel and stretch attacks. In fact, any time adversaries cannot specify the full path, the potential for Vampire attack is reduced. However, malicious nodes can still mis-forward packets, forcing packet forwarding by nodes who would not normally be along packet paths.

3.2.1 Directional antenna attack.

Vampires have little control over packet progress when forwarding decisions are made independently by each node, but they can still waste energy by restarting a packet in various parts of the network. Using directional antenna adversaries can deposit a packet in arbitrary parts of the network, while also forwarding the packet locally. This consumes the energy of nodes that would not have had to process the original packet, with the expected additional honest energy expenditure of O(d), where d is the network diameter, making d/2 the expected length of the path to an arbitrary destination from the furthest point in the network. This attack can be considered a half-wormhole attack

3.2.2. Malicious discovery attack. Another attack on all previously-mentioned routing protocols (including stateful and stateless) is spurious route discovery. In most protocols, every node will forward route discovery packets (and sometimes route responses as well), meaning it is possible to initiate a flood by sending a single message. Systems that perform as-needed route discovery, such as AODV and DSR, are particularly vulnerable, since nodes may legitimately initiate discovery at any time, not just during a topology change. A malicious node has a number of ways to induce a perceived topology change: it may simply falsely claim that a link is down, or claim a new link to a non-existent node

4. Description of Wolf Search Algorithm:

Wolves are social predators that hunt in packs. Wolves typically commute as a nuclear family, which is different from PSO and Fish Swarm, which usually move in relatively large groups. Wolves remain silent and use stealth when hunting prey together. Unlike ants, which use pheromones to communicate with their peers about food traits, WSA forgoes this kind of communication, which shortens the run time of the search. Wolves have developed unique, semi-cooperative characteristics; that is, they move in a group in a loosely coupled formation, but tend to take down prey individually. This detail is important because some optimization algorithms, such as those that are swarm-based, focus on group coordination whereas algorithms that emphasize individual movements fall on the other end of the spectrum. As a synonym in computing, WSA naturally balances scouting the problem space in random groups (breadth) and searching for the solution individually (depth). When hunting, wolves will attempt to conceal themselves as they approach their prey. This characteristic prompts the searching agents in WSA to always look for and move to a better position in the same way that wolves continuously change their positions for better ones with more shelter, fewer terrain obstacles or less vulnerability. When hunting, wolves simultaneously search for prey and watch out for threats such as human hunters or tigers. Each wolf in the pack chooses its own position, continuously moving to a better spot and watching for potential threats. WSA is equipped with a threat probability that simulates incidents of wolves bumping into their enemies. When this happens, the wolf dashes a great distance away from its current position, which helps break the deadlock of getting stuck in local optima. The direction and distance they travel when moving away from a threat are random, which is similar to mutation and crossover in GA when changing current solutions while evolving into a better generation.

Wolves have an excellent sense of smell and often locate prey by scent. Similarly, each wolf in the WSA has a sensing distance that creates a sensing radius or coverage area –generally referred to as visual distance. This visual distance is applied to the search for food (the global optimum), an awareness of their peers (in the hope of moving into a better position) and signs that enemies might be nearby (for jumping out of visual range). Once they sense that prey is near, they approach quickly, quietly and very cautiously because they do not wish to reveal their presence. In search mode, when none of the abovementioned items are detected within visual range, the wolves move in Brownian motion (BM), which mimics the random drifting of particles suspended in fluid.

Based on wolves' hunting behavior, as described above, the three rules that govern the logics of the Wolf Search Algorithm (WSA) are presented as follow.

1. Each wolf has a fixed visual area with a radius defined by v for X as a set of continuous possible solutions. The coverage would simply be the area of a circle by the radius v.In hyper-plane, where multiple attributes dominate, the distance would be estimated by Minkowski distance. Each wolf can only sense companions who appear within its visual circle and the step distance by which the wolf moves at a time is usually smaller than its visual distance.
2. The result or the fitness of the objective function represents the quality of the wolf's current position. The wolf always tries to move to better terrain but rather than choose the best terrain it opts to move to better terrain that already houses a companion. If there is more than one better position occupied by its peers, the wolf will chose the best terrain inhabited by another wolf from the given options. Otherwise, the wolf will continue to move randomly in BM.
3. At some point, it is possible that the wolf will sense an enemy. The wolf will then escape to a random position far from the threat and beyond its visual range.

Merging with Other Wolves

In the implementation of WSA the result/fitness of the objective function reflects the quality of a terrain position that will eventually lead to food. This quality can be defined as either secludicity from predators, higher ground from which it is easier to hunt, or another similar benefit. The intention behind a wolf's decision to change location is to simultaneously secure an increased chance of finding food and a decreased chance of being hunted. Wolves are expected to trust other wolves, because they never prey on each other, therefore a wolf will only move into terrain inhabited by another wolf when that terrain is better. If the new position is better, the incentive is stronger provided that it is already inhabited by a companion wolf. There is another factor that must be considered, specifically the distance between the current wolf's location and its companion's location. The greater this distance, the less attractive the new location becomes, despite the fact that it might be better. This decrease in the wolf's willingness to move obeys the inverse square law

CONCLUSION

Heuristic optimization methods have an edge over their classical counterparts because they can incrementally induce a globally optimum solution by using heuristics to efficiently search a large space. This paper presents a solution to detect vampire attacks using an optimization algorithm, the Wolf Search Algorithm (WSA), which imitates the preying behaviour of wolves and has displayed unique advantages in efficiency because each searching agent simultaneously performs autonomous solution searching and merging.

REFERENCES

[1] M. Gilli, P. Winker, "A Review of Heuristic Optimization Methods in Econometrics", Swiss Finance Institute Research Paper No. 08-12. June 2008, Available at SSRN: http://ssrn.com/abstract=1140655.

[2] X.-S. Yang, "Firefly algorithms for multimodal optimization". Stochastic Algorithms: Foundations and Applications, SAGA 2009. Lecture Notes in Computer Sciences. 5792. pp. 169–178.

[3] X.-S. Yang, S. Deb, "Cuckoo search via Levy flights", in: World Congress on Nature and Biologically Inspired Computing (NaBIC 2009). IEEE Publication, USA. 2009, pp. 210–214.

[4] X.-S. Yang, S. Deb, S. Fong, "Accelerated Particle Swarm Optimization and Support Vector Machine for Business Optimization and Applications", The Third International Conference on Networked Digital Technologies (NDT 2011), Springer CCIS 136, 11-13 July 2011, Macau, pp.53–66.

[5] X.-S. Yang, "A New Metaheuristic Bat-Inspired Algorithm", in: Nature Inspired Cooperative Strategies for Optimization (NISCO 2010), Eds. J.R. Gonzalez et al., Studies in Computational Intelligence, Springer Berlin, 284, Springer, pp.

[6] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energyefficient communication protocol for wireless microsensor networks," In *Proceedings of the 33rd Annual Hawaii International Conference on Systems Sciences,* (HICSS 00), Maui, Hawaii, USA pp. 3005-3014, January, 2000.

[7] M. Younis, M. Youssef, and K. Arisha, "Energy-aware routing in clusterbased sensor networks," In *Proceedings of the 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems,* (MASCOTS 02), Fort Worth, Texas, USA,pp. 129-136, October, 2002.

[8] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," In *Proceedings of the 5th Symposium on Operating System Design and Implementation,* (OSDI 02), Boston, Massachusetts, pp. 147-163, December, 2002.

[9] J. Jang, "A study on a sequenced directed diffusion algorithm for sensor networks," In *Proceedings of the 9th International Conference on Advanced Communication Technology,* (ICACT 07), Gangwon-Do, Korea, vol. 1, pp. 679-683, February, 2007.

[10] S. Hao and T. Wang, "Sensor networks routing via bayesian exploration," In *Proceedings of the 31st IEEE Conference on Local Computer Networks,* (LCN 06), Tampa, Florida, USA, pp. 954-955, November,2006.