IOSR Journal of Engineering (IOSRJEN)
e-ISSN: 2250-3021, **p**-ISSN: 2278-8719
Vol. 3, Issue 7 (July. 2013), ||V3 || PP 47-50

Host to Host Communication in Wireless Sensor Networks Without any Attacks

V. Chandra Sekhar[1], D. Raman[2]
[1](Master of Technology, Computer Science and Engineering,Vardhaman College of Engineering, Hyderabad,India)
[2](Associate Professor, Computer Science and Engineering,Vardhaman College of Engineering, Hyderabad,India)

Abstract: - In a wireless sensor networks, the communication of the network will be affected by the attacks. The most common attacks are denial of service, packet droppers and packet modifiers .In a wireless sensor networks the communication between the hosts is static (the hosts addresses are predefined).If any attacks disrupt the communication then the network performance degrades. To address this problem, a simple yet effective scheme is proposed i.e., the communication between the hosts will be dynamically perform. By this scheme, the performance of the network will be the effective and efficient.

Index Terms: - Approximation algorithm, Caching algorithm, Wireless sensor networks**.**

I. INTRODUCTION

The world is becoming more interconnected with the advent of the Internet and new networking technology. There is a large amount of personal, commercial, military, and government information on networking infrastructures worldwide. Network security is becoming of great importance because of intellectual property that can be easily acquired through the internet. There are currently two fundamentally different networks, data networks and synchronous network comprised of switches. The internet is considered a data network. Since the current data network consists of computer based routers, information can be obtained by special programs, such as "Trojan horses," planted in the routers. The synchronous network that consists of switches does not buffer data and therefore are not threatened by attackers. That is why security is emphasized in data networks, such as the internet, and other networks that link to the internet. Without proper protection, any part of any network can be susceptible to attacks or unauthorized activity. Routers, switches, and hosts can all be violated by professional hackers, company competitors, or even internal employees. In fact, according to several studies, more than half of all network attacks are waged internally. The Computer Security Institute (CSI) in San Francisco estimates that between 60 and 80 percent of network misuse comes from inside the enterprises where the misuse has taken place. To determine the best ways to protect against attacks, IT managers should understand the many types of attacks that can be instigated and the damage that these attacks can cause to e-business infrastructures.

1.1 Common Attacks

Common attacks are broken down into categories. Some attacks gain system knowledge or personal information, such as eavesdropping and phishing. Attacks can also interfere with the system's intended function, the other form of attack is when the system's resources are consumes uselessly, these can be caused by denial of service (DoS) attack. Other forms of network intrusions also exist, such as land attacks, smurf attacks, and teardrop attacks. These attacks are not as well known as DoS attacks, but by name. The frequently occurring attacks are

1. Eavesdropping
2. Viruses
3. Worms
4. IP Spoofing Attacks
5. Denial of Service
6. Packet droppers and modifiers

Now we will discuss about one of the attack i.e., packet droppers and modifiers. Actually the meaning of Packet dropping: A node that drops all or some of the packets that is supposed to forward. Packet modification: A node that modifies all or some of the packets that is supposed to forward. The proposed system is the communication between the hosts i.e., the number hosts will change dynamically according to the requirements. For this purpose, we used approximation algorithm and caching algorithm.

II. SYSTEM MODEL

In this proposed system we use two algorithms. They are Approximation algorithm and caching algorithm.

2.1 Approximation Algorithm

An approximate algorithm is a way of dealing with NP-completeness for optimization problem. This technique does not guarantee the best solution. The goal of an approximation algorithm is to come as close as possible to the optimum value in a reasonable amount of time which is at most polynomial time. Suppose we have some optimization problem instance i, which has a large number of feasible solutions. Also let c(i) be the cost of solution produced by approximate algorithm and c*(i) be the cost of optimal solution. For minimization problem, we are interested in finding a solution of a given instance i in the set of feasible solutions, such that c(i)/c*(i) be as small as possible. On the other hand, for maximization problem, we are interested in finding a solution in the feasible solution set such that c*(i)/c(i) be as small as possible. We say that an approximation algorithm for the given problem instance i, has a ratio bound of p(n) if for any input of sign n, the cost c of the solution produced by the approximation algorithm is within a factor of p(n) of the cost c* of an optimal solution. That is

$\max(c(i)/c^*(i), c^*(i)/c(i)) \leq p(n)$. (1)

This definition applies for both minimization and maximization problems.

Note that p(n) is always greater than or equal to 1. If solution produced by approximation algorithm is true optimal solution then clearly we have p(n) = 1.

For a minimization problem,

$0 < c^*(i) \leq c(i)$. (2)

and the ratio c(i)/c*(i) gives the factor by which the cost of the approximate solution is larger than the cost of an optimal solution.

Similarly, for a maximization problem,

$0 < c(i) \leq c^*(i)$ (3)

and the ratio c*(i)/c(i) gives the factor by which the cost of an optimal solution is larger than the cost of the approximate solution.

This algorithm will be proved in following the cases that are

- Vertex Cover
- The Traveling Salesman Problem

2.2 Caching algorithm

Caching, a fundamental metaphor in modern computing, finds wide application in storage systems, databases, Web servers, middleware, processors, file systems, disk drives, redundant array of independent disks controllers, operating systems and other applications such as data compression and list updating. In a two-level memory hierarchy, a cache performs faster than auxiliary storage, but is more expensive. Cost concerns thus usually limit cache size to a fraction of the auxiliary memory's size. Actually, cache algorithms (also frequently called replacement algorithms or replacement policies) are optimizing instructions algorithms that a computer program or a hardware-maintained structure can follow to manage a cache of information stored on the computer. When the cache is full, the algorithm must choose which items to discard to make room for the new ones.

The average memory reference time is

$T=m*T_m+T_h+E$ (4)

Where

T = average memory reference time

m = miss ratio = 1 - (hit ratio)

T_m = time to make a main memory access when there is a miss (or, with multi-level cache, average memory reference time for the next-lower cache)

T_h = the latency: the time to reference the cache when there is a hit

E = various secondary effects, such as queuing effects in multiprocessor systems

There are two important issues in cache algorithm they are hit ratio and latency. The "hit ratio" of a cache describes how often a searched-for item is actually found in the cache. More efficient replacement policies keep track of more usage information in order to improve the hit rate (for a given cache size).The "latency" of a cache describes how long after requesting a desired item the cache can return that item (when there is a hit). Faster replacement strategies typically keep track of less usage information—or, in the case of

direct-mapped cache, no information—to reduce the amount of time required to update that information. Each replacement strategy is a compromise between hit rate and latency.

In the cache algorithm, the techniques we used are Least Recently Used (LRU): This is the default and is a variation on Least Frequently Used. The oldest element is the Less Recently Used (LRU) element. The last used timestamp is updated when an element is put into the cache or an element is retrieved from the cache with a get call. Least Frequently Used (LFU): For each get call on the element the number of hits is updated. When a put call is made for a new element (and assuming that the max limit is reached) the element with least number of hits, the Least Frequently Used element, is get call. First In First Out (FIFO): Elements are placed in the same order as they come in. When a put call is made for a new element (and assuming that the max limit is reached for the memory store) the element that was placed first (First-In) in the store is the candidate forget call (First-Out).

III. THE PROPOSED SCHEME

In a wireless sensor network the communication between the host is dynamic i.e., the host address is defined dynamically. This can be done by using two algorithms namely approximation algorithm and caching algorithm. The caching algorithm will change the IP address according to the instructions given dynamically. The simulations over a wide range of network and application parameters show that the performance of the caching algorithms.

A distributed implementation based on an approximation algorithm for the problem of cache placement of multiple data items under memory constraint. Information that needs to be transmitted across the network needs to come in packets. A packet is the basic unit of information known by networks and devices, just like words in the English language. The process of packaging the information is called encapsulation. In this process wraps the appropriate protocol and control information to data so that it is understood consistently on the other end, on the receiving end.

The packets can be forward from one host to another host by using IP addresses. The packets can be forward in a network in the form of tree structure. For this purpose it uses Node Categorization algorithm and Heuristic ranking algorithm. All sensor nodes form a directed acyclic graph (DAG) and extract a routing tree from the DAG. The sink knows the DAG and the routing tree, and shares a unique key with each node. When a node wants to send out a packet, it attaches to the packet a sequence number, encrypts the packet only with the key shared with the sink, and then forwards the packet to its parent on the routing tree.

When an innocent intermediate node receives a packet, it attaches a few bits to the packet to mark the forwarding path of the packet, encrypts the packet, and then forwards the packet to its parent. On the contrary, a misbehaving intermediate node may drop a packet it receives. On receiving a packet, the sink decrypts it, and thus finds out the original sender and the packet sequence number. The sink tracks the sequence numbers of received packets for every node, and for every certain time interval, which we call a round.

Each packet is encrypted and padded so as to hide the source of the packet. The packet mark, a small number of extra bits, is added in each packet such that the sink can recover the source of the packet and then figure out the dropping ratio associated with every sensor node. The routing tree structure dynamically changes in each round so that behaviors of sensor nodes can be observed in a large variety of scenarios.

IV. CONCLUSION

In this scheme, the caching algorithms in different ad hoc network scenarios and compare them with other caching schemes, showing that our solution succeeds in creating the desired content diversity, thus leading to a resource efficient information access. The communication between the hosts will change dynamically by this scheme the performance of the network will increase and the communication between the hosts will be done without any disturbances i.e., without any attacks.

REFERENCE

[1]. Chuang Wang, Taiming Feng, Jinsook Kim, Guiling Wang, Wensheng Zhang, "Catching Packet Droppers and Modifiers in Wireless Sensor Networks," IEEE transactions on parallel and distributed systems, vol. 23, no. 5, ,May.2012, pp.835-843.

[2]. H. Chan and A. Perrig, "Security and Privacy in Sensor Networks," Computer, vol. 36, no. 10, , Oct. 2003, pp. 103-105.

[3]. Bhuse, A. Gupta, and L. Lilien, "DPDSN: Detection of Packet- Dropping Attacks for Wireless Sensor Networks," Proc. Fourth Trusted Internet Workshop, 2005.

[4]. 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.

[5]. M. Kefayati, H.R. Rabiee, S.G. Miremadi, and A. Khonsari, "Misbehavior Resilient Multi-Path Data Transmission in Mobile Ad-Hoc Networks," Proc. Fourth ACM Workshop Security of Ad Hoc and Sensor Networks (SASN '06), 2006.

[6]. R. Mavropodi, P. Kotzanikolaou, and C. Douligeris, "Secmr—A Secure Multipath Routing Protocol for Ad Hoc Networks," Ad Hoc Networks, vol. 5, no. 1, 2007, pp. 87-99.

[7] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical En-Route Filtering of Injected False Data in Sensor Networks," Proc. IEEE INFOCOM, 2004.

[8] S. Zhu, S. Setia, S. Jajodia, and P. Ning, "An Interleaved Hop-by- Hop Authentication Scheme for Filtering False Data in Sensor Networks," Proc. IEEE Symp. Security and Privacy, 2004.

[9] H. Yang, F. Ye, Y. Yuan, S. Lu, and W. Arbaugh, "Toward Resilient Security in Wireless Sensor Networks," Proc. Sixth ACM Int'l Symp. Mobile Ad Hoc Networking and Computing (MobiHoc '05), 2005.

[10] Z. Yu and Y. Guan, "A Dynamic En-route Scheme for Filtering False Data in Wireless Sensor Networks," Proc. IEEE INFOCOM, 2006.