Ad Hoc Networks 47 (2016) 26–40

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Improving sensor network immunity under worm attacks: A software diversity approach☆

Yi Yang[a,*], Sencun Zhu[b,c], Guohong Cao[b]

[a] Department of Mathematics and Computer Science, Fontbonne University, United States
[b] Department of Computer Science and Engineering, The Pennsylvania State University, United States
[c] College of Information Sciences and Technology, The Pennsylvania State University, United States

ARTICLE INFO

Article history:
Received 11 June 2015
Revised 15 October 2015
Accepted 26 April 2016
Available online 27 April 2016

Keywords:
Sensor worm defense
Software diversity
Wireless sensor networks
Deployment errors
Percolation theory

ABSTRACT

Because of cost and resource constraints, sensor nodes do not have a complicated hardware architecture or operating system to protect program safety. Hence, the notorious buffer-overflow vulnerability that has caused numerous Internet worm attacks could also be exploited to attack sensor networks. We call the malicious code that exploits a buffer-overflow vulnerability in a sensor program *sensor worm*. Clearly, sensor worm will be a serious threat when an attacker could simply send a single packet to compromise the entire sensor network. Despite its importance, so far little work has focused on sensor worms.

In this work, we first illustrate the feasibility of launching sensor worms through trial experiments on Mica2 motes. Inspired by the *survivability through heterogeneity* philosophy, we then explore the technique of software diversity to combat sensor worms. Given a limited number of software versions, we design an efficient algorithm to assign the appropriate version of software to each sensor, so that sensor worms are restrained from propagation. We also examine the impact of sensor node deployment errors on worm propagation, which directs the selection of our system parameters based on percolation theory. We then extend the above scheme by considering enhanced sensors that can load multiple program versions. We show that the existence of enhanced sensors could further improve the immunity and robustness of sensor networks under worm attacks. Finally, extensive analytical and simulation results confirm the effectiveness of our schemes in various settings.

1. Introduction

For sensor nodes operating in an unattended and hostile environment, they often face various security attacks. Due to their low manufacturing cost (e.g., less than one dollar for envisioned smartdust microsensors), it is unlikely that we will use expensive tamper-resistant hardware for these sensors. Therefore, malicious attacks are inevitable and they may be launched to obtain important information from sensor networks, to interfere with their normal operations, or even to destroy them. The question is how much we can do to increase the survivability of a sensor network that is under attacks.

In the literature, many security mechanisms have been proposed to defend against various kinds of attacks in sensor networks, for example, dodging communication channel jamming, thwarting MAC layer attacks, countering attacks against routing protocols, providing attack-resilient data aggregation [32], and node localization. However, we notice that a potentially more severe attack has not yet been studied. We name this attack *sensor worm attack*. More specifically, we define sensor worms as crafted messages that exploit software vulnerability of sensor nodes in a sensor network, causing sensor nodes to crash or taking control of sensor nodes. Clearly, sensor worm attack could be the most dangerous one if an attacker simply sends a single message to compromise the entire sensor network, defeating the mission of the sensor network.

Naturally, the first question that will arise is: *is it possible for worm attacks to occur in sensor networks*? On one hand, compared with regular computer systems, it is even easier for sensors to be compromised by worm attacks. This is because sensors do not have complicated hardware architecture or operating system to protect program safety due to cost and resource constraints. Moreover, sensors in the same network are homogeneous in both

☆ This work is supported by NSF CCF-1320605 and NSA 2015 CAE Cybersecurity Research Grant.

* Corresponding author.
E-mail address: yyang@fontbonne.edu (Y. Yang).

http://dx.doi.org/10.1016/j.adhoc.2016.04.011
1570-8705/

hardware and software. If one sensor is compromised because of a program vulnerability, all the other sensors are vulnerable to the same compromise attack. On the other hand, worm attack on sensors is not exactly the same as that on regular computers. The Harvard architecture of many sensor microcontrollers has separate memories for program and data. This prevents a piece of malicious code directly injected into the stack of the data space from being executed in such sensors. While this is generally true, as a trial study we conducted experiments on Mica2 motes and found that a buffer overflow vulnerability, the common trigger of worm attacks, could result in the transfer of program flow to a transmission component in the code space. Then, an exploited sensor may relay the attack packet it received before becoming irresponsive. Consequently, this leads to the propagation of the worm packet over the entire network and the failure/corruption of all the sensors.

The next question to be answered is: *what can we do to defend against such sensor worm attacks*? Although a huge body of literature exists on addressing the buffer overflow problem for protecting both clients and servers in the Internet, it is not immediately clear whether and how these solutions may be adapted to the sensor system, which features wireless communication, high connectivity, different hardware architecture, OS (e.g., TinyOS in Mica motes) and programming language (e.g., nesC for TinyOS). While this remains an interesting open research question, in this work we are more interested in improving the *survivability* of entire sensor networks under worm attacks.

In spirit of the *survivability through heterogeneity* philosophy, we explore the technique of *software diversity* to combat sensor worms. While the general idea of software diversity is not new and it has been applied to wired networks [9,16,26], its application to sensor networks faces unique challenges due to high node density and sensor deployment error. High node density implies that it is unrealistic to assign a different version to each node; deployment errors could lead to the potential danger that sensor nodes with the same version of code become neighbors (connected) after they are deployed into the field.

To address the above challenges, we first adopt a location-based version assignment strategy. Given a limited number of software versions of the same functionality, we load every sensor with a proper version of the software through an efficient graph color assignment algorithm, such that a sensor worm may be isolated in a small "island". Then we analyze based on percolation theory the impact of deployment errors on sensor worm propagation, which gives some practical guidelines for choosing appropriate system parameters to minimize the chance of worm propagation. We then extend the above scheme by considering enhanced sensors that can load multiple program versions. We show that the existence of enhanced sensors could further improve the immunity and robustness of sensor networks under worm attacks. Finally, our performance evaluation demonstrates the effectiveness of the proposed schemes in defending against sensor worm attacks. It also shows that our schemes greatly outperform two other version assignment algorithms proposed in [26].

The remainder of this paper is organized as follows. First, related work is discussed in Section 2. Then, Section 3 studies the feasibility of launching sensor worm attacks. After that, Section 4 describes our sensor worm defensive schemes. Performance evaluation is presented in Section 5. Finally, we summarize our work and discuss future work in Section 6.

2. Related work

Next, we present related work in worm attack defense, software diversity, graph coloring, and sensor worm defense.

2.1. Worm attack defense

Worm attacks in Internet [30] as well as buffer overflow vulnerabilities [27] have been extensively studied. Both proactive and reactive strategies are proposed to defend against worm attacks in Internet [11]. Also, Internet worm propagation is modeled and the impact of network topology on the size of the final infected population has been analyzed in [13,17]. Wang et al. [31] propose generic techniques for blocking buffer overflow attacks based on some inherent distinctions between exploit code and random data.

2.2. Software diversity

Inspired by diversity, an important source of robustness in biological systems, software diversity in computer systems [16] as well as computer networking [9,26] bears a lot of attention recently. A variety of randomization techniques [18] have been proposed to enhance the intrusion resistance of computer systems by increasing software complexity without degrading functionalities and performance. Diversification at the network level is achieved by applying different operating systems, critical software components [24] or communication protocols [25] on different machines of the network. How to apply software diversity in heterogeneous computer networks has been discussed in [44].

Similar work is conducted in preventing epidemics in the context of computer worms or viruses [10]. In [28], it is stated that selective immunization should be enforced according to the node's degree, i.e., nodes with high degree should be installed different softwares because they are more important in the network connectivity.

2.3. Graph coloring

Graph coloring (especially vertex coloring) [21], a famous problem in graph theory, ensures that there are no two adjacent nodes sharing the same color. So, its solution is a natural option for making globally optimal decision in software diversity. In the distributed coloring algorithms proposed in [26], the initial random assignment of nodes' colors will cause high communication overhead in the following color adjustment and negotiation with neighbors. Also, the algorithms may not converge to a few colors due to the high density of sensor networks, which in practice may result in a high cost for software implementations. Our sensor worm defense schemes have already considered all these factors as well as the sensor deployment errors, thus are well-tailored for sensor networks.

2.4. Sensor worm

De et al. [12] models node compromise spread in wireless sensor networks using epidemic theory and identifies key factors determining potential outbreaks. However, the sensor deployment error and sensor compromise containment strategies are not considered in this work. Other techniques [23,29] were designed for sensor memory protection. These works together with [8] improve the defensive capabilities of individual sensors. It is necessary for us to enhance sensor network immunity under worm attacks in a systematic way, since the defensive capability of individual sensors is limited.

In our previous work [45], we proposed a software diversity based approach to improve sensor networks' immunity under worm attacks. In this work, the feasibility of sensor worms is first investigated. Then, a graph coloring based software assignment algorithm is implemented to enhance the diversity of sensor networks against sensor worms. In the end, the impact of sensor deployment errors is examined and the percolation theory is applied