Computer Communications 54 (2014) 67–83

Contents lists available at ScienceDirect

Computer Communications

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

A specification-based intrusion detection engine for infrastructure-less networks

Christoforos Panos [a], Christos Xenakis [b,*], Platon Kotzias [b], Ioannis Stavrakakis [a]

[a] Department of Informatics & Telecommunications, University of Athens, Panepistimioupolis, Ilisia, 15784, Athens, Greece
[b] Department of Digital Systems, University of Piraeus, 80 Karaoli & Dimitriou Street, 18534 Piraeus, Greece

ARTICLE INFO

Article history:
Received 20 March 2014
Received in revised form 22 July 2014
Accepted 1 August 2014
Available online 11 August 2014

Keywords:
MANET
IDS
AODV
Detection engine
Attestation

ABSTRACT

The proliferation of mobile computing devices has enabled the utilization of infrastructure-less networking as commercial solutions. However, the distributed and cooperative nature of routing in such networks makes them vulnerable to a variety of attacks. This paper proposes a host-based monitoring mechanism, called SIDE that safeguards the operation of the AODV routing protocol. SIDE encompasses two complementary functionalities: (i) a specification-based detection engine for the AODV routing protocol, and (ii) a remote attestation procedure that ensures the integrity of a running SIDE instance. The proposed mechanism operates on a trusted computing platform that provides hardware-based root of trust and cryptographic acceleration, used by the remote attestation procedure, as well as protection against runtime attacks. A key advantage of the proposed mechanism is its ability to effectively detect both known and unknown attacks, in real time. Performance analysis shows that attacks are resolved with high detection accuracy, even under conditions of high network volatility. Moreover, SIDE induces the least amount of control packet overhead in comparison with a number of other proposed IDS schemes.

1. Introduction

Infrastructure-less networks comprise a wide range of networking paradigms such as mesh networks, mobile ad hoc networks (MANETs), vehicular ad hoc networks (VANETs), delay tolerant networks (DTNs), opportunistic and sensor networks, as well as various overlay networks. A common characteristic of these networks is the absence of any fixed architectural component such as routers and access points, supporting and serving dynamic topologies and behaviors. These unique properties, empowered by the proliferation of mobile devices (i.e., smartphones, tablets, etc.) and the advent of ad-hoc networking standards, such as Wi-Fi direct [1], enable the materialization of infrastructure-less networks for providing communication and cooperation solutions, such as the extension of networking environments (i.e., cellular networks, personal or corporate wireless networks, etc.) in areas where network coverage is limited [3] (i.e., metropolitan areas, indoor environments, etc.).

A widely accepted implementation of an infrastructure-less network is based on a dynamic and adaptive routing protocol, named ad hoc on demand distance vector (AODV) [2], which, initially, was designed for MANETs and later has been adopted by DTN [4], opportunistic [5], mesh [6], and sensor [7] networks. AODV operates with the assumption that all participating nodes are well-behaved, and thus, it does not include any security mechanism. Considering also the deployment characteristics of infrastructure-less networks (i.e., wireless shared access, dynamic topologies, cooperative routing, etc.), it can be realized that AODV faces a wide set of security threats [11]. More specifically, any malicious network node may easily exploit critical protocol fields such as *hop count*, *sequence numbers*, and *source and destination address*, causing a variety of attacks, such as route disruption, resource consumption and denial of services [9].

Since the protection of the protocol's fields and functionality is not possible by default, an effective way to address these inherent vulnerabilities is through the deployment of a detection mechanism. However, the design of an intrusion detection system (IDS) for AODV has been proven a challenging task, considering the limitations of the existing IDS [8,12,23] (i.e., analyzed in Section 2.2 of this paper). The majority of them capture, store, and, subsequently, process the whole traffic (i.e., control and payload) within the radio range of a monitoring node, in order to collect as much audit data as possible and then assess the behavior of the neighboring nodes. Consequently, monitoring nodes bear additional computational

* Corresponding author.
E-mail addresses: cpanos@di.uoa.gr (C. Panos), xenakis@unipi.gr (C. Xenakis), platon@unipi.gr (P. Kotzias), ioannis@di.uoa.gr (I. Stavrakakis).

http://dx.doi.org/10.1016/j.comcom.2014.08.002
0140-3664/

and storage burdens, while energy consumption is increased. In addition, during the collection of audit data, malicious activities are not detected. Finally, in cases of high nodes' mobility or continuous changes in network topology, the collected audit data might lead to inconclusive or erroneous assessments, resulting in false positives/negatives.

The limitations and weaknesses of current IDSs may be addressed by a host-based IDS that monitors the behavior of its own host node. A host-based IDS alleviates the need for collecting audit data that may be malicious, incomplete, or outdated, providing an accurate and real time view of the host node's protocol operations. Thus, malicious behaviors can be detected immediately, with low false positives/negatives, and without the associated overheads of audit data collection. However, such an approach has been unfeasible in the past, mainly, because of the fact that a host-based IDS operating on a malicious node, could not be considered as trusted. The emergence of trusted computing [20] may address this uncertainty and make host-based IDS a viable security solution for infrastructure-less networks. Trusted computing provides hardware-based root of trust, accompanied by a set of primitive functions that propagates trust from hardware to the application software. At the core of this technology resides the process of remote attestation with which a computer can prove the integrity of a platform (e.g., hardware and software) to a remote party [38].

This paper proposes a novel host-based monitoring mechanism, called SIDE (i.e., Specification-based Intrusion Detection), which relies on trusted computing in order to provide a resilient, specification-based IDS. More specifically, each network node implements an instance of SIDE, which unlike existing IDSs, is responsible for monitoring its own host node. This approach enables SIDE's detection engine to monitor local information and ascertain an accurate view of protocol operations, in real time. SIDE's detection engine is based on a comprehensive *set of specifications* that defines the legitimate functionality of the AODV protocol. As a result, any malicious activity (i.e., known or unknown) that violates the legitimate functionality of AODV can be identified. To defend against malicious host nodes that may attempt to modify or even disable SIDE, the proposed mechanism encompasses a *remote attestation procedure* that verifies the integrity of running SIDE instances in the network. Moreover, SIDE operates on a *trusted computing platform* that provides hardware-based root of trust and cryptographic acceleration, used by the remote attestation procedure, as well as protection against runtime attacks. The proposed mechanism utilizes a TrustZone [42] enabled ARM processor, which constitutes a trusted computing platform included in the vast majority of mobile and embedded devices. The performance of SIDE is evaluated through an extensive set of simulations. The numerical results show that SIDE resolves attacks in real time with high detection accuracy, while imposing limited overheads in the operation of AODV.

The rest of this paper is organized as follows. Section 2 analyzes the functionality of the AODV routing protocol; briefly evaluates existing security schemes that have been proposed for AODV; and provides an analysis of remote attestation techniques. In Section 3 the proposed mechanism is introduced and its functionality is elaborated. In Section 4, we perform an in-depth evaluation of SIDE, which includes: (i) an outline of its advantages over previously proposed detection engines; (ii) a security evaluation of its robustness against a variety of attacks; (iii) the computational cost and memory requirements, and, (iv) a comparative evaluation of its performance based on simulations. Finally, Section 5 contains the conclusions.

2. Background

In this section, we first provide an overview of the AODV protocol's functionality. This overview covers only the most critical aspects of the protocol's operations, since a more throughout analysis of AODV exists in [2]. In Section 2.2, we provide an evaluation of several security solutions that have been proposed for AODV. A comprehensive analysis of all the related literature requires an extensive review, which is outside the scope of this paper. Instead, we have selected a representative set of security solutions that covers the majority of utilized security mechanisms and encompasses: (i) extensions to the AODV protocol that incorporate cryptography and (ii) intrusion detection mechanisms that use either anomaly-based or specification-based detection. Finally, in Section 2.3, we evaluate existing remote attestation procedures.

2.1. Overview of the AODV routing protocol

AODV is an on demand routing protocol, which maintains routes as long they are needed by source nodes. It is scalable and offers low processing, memory, and communication overheads to the underlying network. It utilizes three control messages to achieve route discovery: route request (RREQ), route reply (RREP), and route error (RERR). It also provides an optional fourth control message (i.e., Hello message), which is used for preserving connectivity between neighboring nodes. When a node wishes the establishment of a route, it initiates a route discovery process by broadcasting a RREQ message that includes the: *source IP address*, *source sequence number*, *destination IP address*, *destination sequence number*, *RREQ id* (i.e., an incremented identifier), and *hop count field*. Each RREQ message is, uniquely, identified by the pair of *source IP address* and *RREQ id*. The intermediate nodes that receive the RREQ may either reply to it (i.e., possess an updated route to the destination) or forward it (i.e., do not possess a route to the destination and the time to live (TTL) field is greater than one). In case that multiple copies of the same RREQ are received by an intermediate node, the duplicates are discarded. The destination node or an intermediate node that has a fresh route to the destination replies to a RREQ, by generating an RREP message that contains the: *source IP address*, *source sequence number*, *destination IP address*, *destination sequence number* (i.e., an increasing counter denoting the most recent route), *lifetime field* (i.e., indicates the time for which the route is considered valid), and *hop count field* (i.e., denotes the distance in hops from the source to the destination). Intermediate nodes receiving the RREP update their routing tables, only, if the *destination sequence number* in the message is higher from the stored value in their routing tables, or the *destination sequence numbers* are equal, but the *hop count field* in the RREP is smaller than the stored value. If a link breaks, an intermediate node initiates a local repair mechanism attempting to discover a new route to the destination by transmitting a RREQ message. If the repair mechanism fails to discover a route, the node generates a RERR message that includes the *IP addresses* and the last known *destination sequence numbers* of the unreachable destinations, informing the receiving nodes that they should restart the routing discovery process, if they want to communicate with them.

A node offers connectivity information by broadcasting local Hello messages, if this feature is enabled. Every time-period of *hello interval*, the node broadcasts a Hello message, which contains the: *destination IP address*, *destination sequence number*, *lifetime field*, and *hop count field*. The *lifetime field* is assigned the value *allowed hello loss* * *hello interval*, while the *hop count* is set equal to zero. The *allowed hello loss* parameter is used by network administrators to determine the time frame (i.e., in multiples of the *hello interval*), where the routes are considered valid. Nodes perceive connectivity by listening to the packets transmitted by their neighbors. If a node does not receive any packet from a neighbor for a time period greater than *allowed hello loss* * *hello interval*, it assumes that the link to this node is currently lost.