Implementation of a Security Framework for Wireless Multi-hop Networks

Stefano Paris
Department of Electronics and Information
Politecnico di Milano
paris@elet.polimi.it

Antonio Capone
Department of Electronics and Information
Politecnico di Milano
capone@elet.polimi.it

ABSTRACT

Wireless Multi-hop Networks represents an emerging technology for next-generation wireless networking. Although very little attention has been devoted so far to the security issues of this technology, they represent a critical feature that can limits the employment of this technology as a convenient alternative to others networking solutions.

In this demo we present an implementation of a security framework for Wireless Multi-hop Networks that provides authentication and access control of wireless nodes as well as a key distribution scheme.

We compare our security framework both with a static key approach and with an end-to-end solution that consists in establishing an encrypted IPSec tunnel.

The experimental analysis shows that our solution enhances the network security with a negligible impact on the network performance, thus representing an effective solution for wireless multi-hop networking.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms

Security, Performance

Keywords

Wireless Multi-hop Networks, Experimental Test-bed, Security

1. INTRODUCTION

Wireless Multi-hop Networks, such as wireless mesh and ad-hoc networks, have emerged as a technology for next-generation wireless networking [1].

Q2SWinet'09, October 28–29, 2009, Tenerife, Canary Islands, Spain.

The use of wireless medium makes these networks more susceptible to attacks. Passive attacks like eavesdropping let malicious violate the confidentiality of the information exchanged over the network, whereas active attacks involve actions performed by rivals to gain the control of the network. For instance, the topology information that devices exchange among each others are in this case replicated, modified or forged, in order to deny access to users, steal the identity of legitimate nodes or to get sensible positions inside the network.

In this paper we show MobiSEC[1], a complete security framework originally proposed in [5] for both the access and backbone areas of the Wireless Mesh Network (WMN). It provides authentication and access control of network nodes as well as security and integrity of all data communications that occur in the WMN. This is achieved with layer-2 encryption that uses a shared key whose delivery is assured by two key distribution protocols.

MobiSEC extends the IEEE 802.11i standard to the wireless multi-hop scenario, exploiting the routing capabilities of the wireless nodes. A two-step approach is adopted: in the first step new nodes perform the authentication process with the nearest neighbor, according to the 802.11i protocol, like generic wireless stations. In the second step, these nodes can upgrade their role in the network, by further authenticating to a central server, obtaining a temporary key with which all traffic is encrypted.

The proposed solution has been implemented and integrated in MobiMESH [6], a experimental platform that provides a complete framework for analyzing, studying and testing the behavior of a multi-hop network in a real-life environment.

We measured the performance of MobiSEC in several realistic network scenarios and we compared it both with a static approach that consists in using a fixed key to protect all the wireless links, as well as with an end-to-end solution that consists in establishing an encrypted IPSec tunnel.

Experimental results show that our solution exhibits very high robustness against cryptanalytic attacks, and it further obtains a very high throughput, which is almost the same as that achieved by the static key approach. Therefore, it represents a very effective solution to provide security in wireless multi-hop environments without impairing the network performance.

The paper is structured as follows: Sections 2 describes

[1]A Network Simulator (ns2) implementation of MobiSEC is available online at http://home.dei.polimi.it/paris/mobisec.html

the security architecture and the key distribution protocol. Section 3 discusses the experimental results which show the effectiveness of our solution, whereas Section 4 presents the demonstration.

2. DESCRIPTION OF THE SECURITY ARCHITECTURE

In this Section we provide a general description of MobiSEC, the architecture we originally proposed in [5] to authenticate the network nodes and secure the traffic exchanged over the wireless multi-hop backbone of a infrastructured mesh network.

Figure 1 shows in detail the message exchanges that occur between a generic wireless node and the Key Server during the execution of the key distribution protocols defined by MobiSEC. These message exchanges enable all network nodes to obtain the same cryptographic material (denoted as *session secret*) used in a session to generate the keys that secure the communications of the wireless network.

According to the protocol used by all nodes, the *session secret* represents a key list (Server Driven Protocol) or a seed (Client Driven Protocol) that is used to compute the sequence of keys with a scheme that resembles a hash-chain method.

A generic node, after entering in the radio range of a wireless node already connected to the multi-hop network, authenticates itself to a central server (namely the *Key Server*) and sends its first request to obtain the secret S used in the current session by the other nodes that form the wireless network, and the time when it was generated, t_v, which represents the starting validity time. Let us define a *session* as the maximum validity time of the secret currently shared by all nodes.

The node, based on the instant at which it joins the backbone (t_n in Figure 1), can compute an identifier of the key currently used by its peers (k_{id}), and its remaining validity time (T_1), according to the following expression:

$$\begin{aligned} k_{id} &= \left\lfloor \frac{t_n - t_v}{timeout} \right\rfloor + 1 \\ T_1 &= key_{id} \cdot timeout - (t_n - t_v) \end{aligned} \tag{1}$$

According to the distribution protocol, the key identifier is used by the node to extract the current key from the provided session secret (Server Driven Protocol), i.e. $key(k_{id}, S) = S[k_{id}]$, or it represents the number of times the node has to apply a hash function to the session secret (Client Driven Protocol) to derive the current key, as indicated in Equation 2.

$$\begin{cases} key(k_{id}, S) = hash(S) & \text{if } k_{id} = 1 \\ key(k_{id}, S) = hash(key(k_id - 1, S)) & \text{if } k_{id} > 1 \end{cases} \tag{2}$$

Only authorized nodes that have the necessary credentials can authenticate to the Key Server and obtain the cryptographic material needed to derive the key sequence used to protect the wireless network.

In order to minimize the risks of using the same key for a long time and improve the overall security, the Key Server generates proactively a different secret for the successive session.

It is important that each node obtains the secret that will be used in the next session before the current session expires. This is especially true for nodes that take a long time to receive the response from the server (due, for example, to slow links or high number of hops from the server). In fact, if the request is sent when the current session is about to expire, the first nodes that receive the response will cut off the others when they enable the new key.

The key identifier that triggers the proactive request to the server is set equal to the difference between the maximum number of keys related to a specific session and a correction factor (c), which is computed based on the time necessary to receive the response from the Key Server (Δt), according to Equation (3). In this equation, t_s is the time when the first or proactive key request was sent, and t_r is the time when the corresponding key response was received from the Key Server. Hence, if a node takes a time (Δt in Equation (3)) greater than *timeout* to receive the response from the Key Server, it must perform the next proactive request before setting the last key.

$$\begin{aligned} &\Delta t = t_r - t_s \\ &\begin{cases} c = \lceil \frac{\Delta t - timeout}{timeout} \rceil & \text{if } \Delta t \geq timeout \\ c = 0 & \text{if } \Delta t < timeout \end{cases} \end{aligned} \tag{3}$$

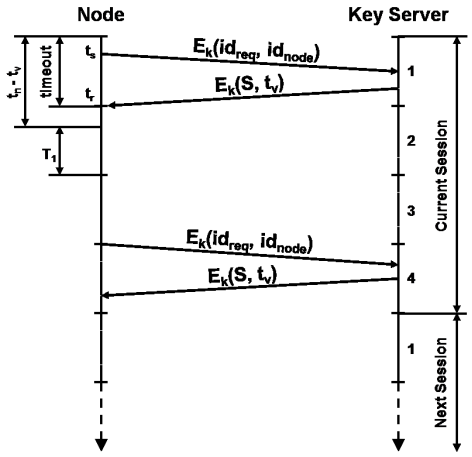


Figure 1: Key Distribution Protocol. $E_k(\bullet)$ represents the symmetric cryptographic function used to protect the security of the messages, whereas id_{req} and id_{node} represent the identifier of the request and of the node, respectively.

Figure 2 shows the high-level architecture of the security framework. We implement the key distribution protocols as a client/server application using the OpenSSL library to authenticate and protect the connection that is established between a wireless node and the Key Server to exchange the cryptographic material.

The Key Switcher module implements the tasks defined by our protocols to obtain and install the currently used key from the *session secret*. This component operates as a

kernel module to improve its responsiveness, since it will be scheduled with a higher priority than user space processes. Therefore, concurrent user space processes cannot cause a scheduling delay greater than the key validity time.

On the other hand, such delay has a negligible effect on the client-side application (the Client Daemon module in Figure 2), since the correction factor that is used to trigger the proactive request takes into account also this contribution.

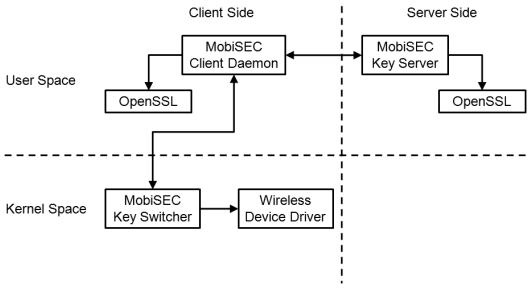


Figure 2: MobiSEC Architecture. The client-side application is installed on all network nodes, whereas the server-side application is installed exclusively on the node that acts as Key Server.

3. EXPERIMENTAL STUDY

In this Section we present the numerical results obtained testing the proposed security framework within the MobiMESH test-bed [6].

Figure 3 shows the two devices that we use to implement the proposed security framework. The node illustrated in Figure 3(a) is an embedded system based on a VIA Epia Board, equipped with a PCI-to-MiniPCI expander that permits to install up to four MiniPCI wireless cards. Therefore, this node can act simultaneously both as router and access point. The single-radio device (Figure 3(b)) is a Linksys Broadband Wireless Router WRT54G whose architecture is based on the MIPSEL platforms which hosts a Broadcom 4712 processor overclocked to 216MHz.

Both the devices run a customized version of the open-source firmware OpenWRT, which is an embedded version of the Linux operating system.

(a) Multi-Radio Node (b) Single-Radio Node

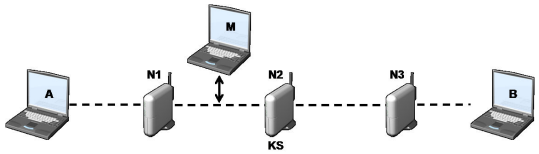
Figure 3: Embedded devices used to implement the wireless multi-hop tesdbed.

We first measured the throughput of a long-lived TCP connection, whose segments are routed on a multi hop wireless path protected either by the Server Driven or the Client Driven Protocol; then, we compared the obtained results with those achieved using a static key and an end-to-end approach like IPSec.

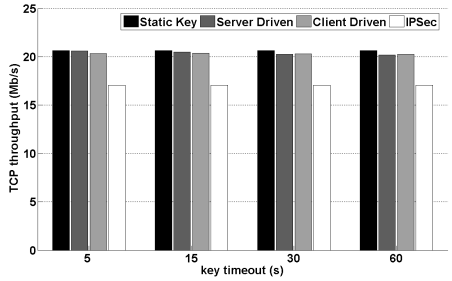
To prove the robustness of MobiSEC, we used a weak cryptographic system, i.e. WEP with a key length of 128 bit, and we tried to crack the key from the packets sniffed with the aircrack-ng tool, which implements the attack designed by Fluhrer, Mantin and Shamir (FMS attack) [7] with the KoreK improvements [8]. In all the tests we set the key timeout to 60 seconds and the session duration to 240 seconds. Such value is obtained by setting the maximum number of keys related to a specific session to 4.

We considered the multi-hop network scenario illustrated in Figure 4(a), where each node (N_1, N_2 and N_3) is connected only with the previous and the successive node.

Experimental results have been obtained generating TCP traffic between the clients A and B with the *iperf* traffic generator. All embedded nodes run the client side application of the MobiSEC framework. Node N_2 also acted as Key Server, to evaluate the effect of the network load on our protocols, since in this configuration both N_1 and N_3 send their requests to N_2. Figure 4(b) shows the TCP throughput of the connection established between clients A and B, secured by the considered protection schemes, as a function of the test duration. The results confirm that our solution, unlike IPSec, does not reduce the network performance with respect to a static key approach.



(a) Topology



(b) TCP throughput

Figure 4: Multi-Hop Topology. A multi-hop data transfer between nodes A and C is performed to measure the network performance.

Strength analysis has been carried out in the same network scenario to evaluate how much our solution increases the overall security. Such analysis was performed sniffing the traffic transmitted between nodes N_1 and N_2 and then applying a cryptanalytic attack with the aircrack-ng tool. Table 1 reports the outcome of the cryptanalytic attack as a function of the time spent to gather the packets on which the attack is performed: only the static WEP key was broken, but the number of packets needed to get the key was significantly larger than the theoretical number indicated in [8]. In this works, the authors suggest that the number of pack-

ets necessary to crack a 128 bit WEP key is approximately $5 \cdot 10^5$ - 10^6, that is equivalent to 110-220 seconds considering an Ethernet packet and the theoretical throughput of an 802.11a/g wireless link. Therefore, setting the maximum key validity time to 60 s, as we did in MobiSEC, turns out to be quite a conservative choice.

Table 1: Outcome of the cryptanalytic attack.

	Packet-Gathering Time (s)		
Protocol	60	240	600
Static Key	Failed	Failed	Cracked (50 s)
Server Driven	Failed	Failed	Failed
Client Driven	Failed	Failed	Failed

4. DEMO HIGHLIGHTS

During this demonstration, we use two standard wireless routers (Linksys WRT54G) as relaying nodes and three laptops. Figure 5 shows the network topology that will be employed during the demonstration to show the security and performance improvements provided by our framework.

In the first phase of the demonstration, laptop A and B exchange data traffic and a video stream while the adversary laptop M performs a cryptanalytic attack on the traffic sniffed on the link N_1-N_2, which is protected by a static key. After breaking the key, M will show the video stream exchanged by the other two laptops.

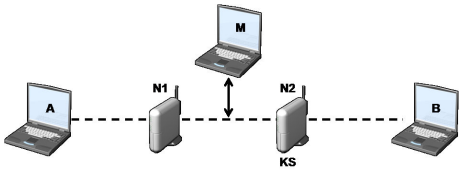


Figure 5: Network topology of the demonstration.

In the second phase, the two relaying nodes start executing the key distribution protocols in order to show the validity of the security architecture, in terms of improved security and network performance. Laptop B will keep receiving the video stream sent by notebook A, whereas M will not able to steal the information and the video transmitted through the multi-hop network.

Acknowledgments

This work was partially supported by MIUR in the framework of the PRIN SESAME project.

5. REFERENCES

[1] I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: a survey. *Computer Networks*, 47(4):445–487, 2005.

[2] N. Ben Salem and J.-P. Hubaux. Securing wireless mesh networks. *IEEE Wireless Communications*, 13(2):50–55, 2006.

[3] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang. Self-securing ad hoc wireless networks. *ISCC '02*, pages 567–574, 2002.

[4] N. Milanovic, M. Malek, A. Davidson, and V. Milutinovic. Routing and security in mobile ad hoc networks. *IEEE Computer*, 2004.

[5] F. Martignon, S. Paris, and A. Capone. Design and Implementation of MobiSEC: a Complete Security Architecture for Wireless Mesh Networks. *Computer Networks*, 53(12):2192–2207, August 2009.

[6] A. Capone, S. Napoli, and A. Pollastro. MobiMESH: An experimental platform for wireless mesh networks with mobility supports. *WiMESHNets '06*. ACM, 2006.

[7] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the key scheduling algorithm of RC4. *Lecture Notes in Computer Science*, 2259:1–24, 2001.

[8] W. A. Arbaugh, N. Shankar, Y. C. J. Wan, and K. Zhang. Your 802.11 wireless network has no clothes. *IEEE Wireless Communications*, 9(6):44–51, 2002.