Secure and Privacy-Preserving Cross-Layer Advertising of Location-Based Social Network Services

Michael Dürr, Florian Gschwandtner,
Corina Kim Schindhelm, and Markus Duchon

Mobile and Distributed Systems Group, Department for Informatics,
Ludwig-Maximilians-University Munich, 80538 Munich, Germany
{michael.duerr,florian.gschwandtner,corina-kim.schindhelm,
markus.duchon}@ifi.lmu.de

Abstract. We present a novel cross-layer protocol design which integrates in our decentralized *Online Social Network Vegas* and allows for secure and privacy-preserving advertising and communication of *Location-based Social Network Services* over WiFi. Our proposal requires minimal modifications to the MAC Layer and could be easily integrated into upcoming standards like IEEE 802.11u.

Keywords: Security, Privacy, Mobile Social Networks, Cross-Layer, WiFi, Beaconing.

1 Introduction

Due to the skyrocketing participation in *Online Social Networks* (OSNs) like *Facebook Places* and *Foursquare*, a remarkable increase of *Location-based Social Network Services* (LB-SNSs) can be observed. Unfortunately, mobile users still suffer from limited 3G connectivity or restrictive pricing plans. In order to continue the ongoing LB-SNS, they attempt to switch to private WiFi access networks or public WiFi hotspots. As mobile users often have to choose between a plethora of WiFi access points (APs) (e.g. provided by coffee shops, shopping malls, or private persons), switching to one of them turns out to be complex and cumbersome. Present WiFi installations are subject to heterogeneous authentication schemes like WPA or WebAuth which results in a trial-and-error procedure when a mobile device does not automatically select the preferred AP. As a result, once associated with an AP, a mobile device cannot receive useful information (e.g. load statistics of neigboring APs) from other APs in radio range. Chandra et al. [1] recently identified this problem and proposed different approaches to code such information into IEEE 802.11 beacon frames. The upcoming standard IEEE 802.11u attempts to provide a general solution as it facilitates unauthorized access to an AP in case a mobile device has another authorizing relationship to an external network. In addition, new features like transparent Layer 2 support for authentication in combination with access to

J.Y. Zhang et al. (Eds.): MobiCASE 2011, LNICST 95, pp. 331–337, 2012.

external profile information from OSNs allow for novel location-based and personalized mobile services. Such services could comprise of consumer-selective product advertisements as well as personalized voucher and coupon distribution of nearby businesses, individual public transport schedule broadcasts from nearby bus- or suburban train stations, or community-restricted content sharing through private APs. Although these opportunities appear promising for the commercial domain, from a consumer perspective, they cause severe privacy and security concerns.

We present a secure and privacy-preserving cross-layer protocol that facilitates the advertisement and utilization of LB-SNSs. The protocol is tailored for our OSN architecture Vegas, a decentralized OSN that has been developed based on our previous work [2]. A Vegas-based LB-SNSs was recently published in [3].

2 Protocol Design

We identify three different situations that our protocol must be able to deal with: *a)* the mere recognition and authentication of an advertised service, *b)* the interpretation and processing of the advertised content, and *c)* the establishment of an optional reverse channel from a mobile device to the corresponding AP for advanced services. We decided to split a service advertisement into two parts. Service advertisement identification information is broadcasted within a Layer 2 MAC frame, whereas the advertisement itself is carried within a Layer 7 UDP packet. This helps to minimize data sent within a beacon frame necessary to identify the advertised type of service and still allows for simple authentication of the broadcasting AP.

2.1 Vegas Design

As we focus on security and privacy, we decided to integrate our solution into our decentralized, secure and privacy-preserving OSN Vegas. Vegas does not allow for communication between participants that are not directly connected by an edge of the underlying social graph. This restriction is motivated by a problem we termed *social network pollution* [2]. To give a few examples of social network pollution, present OSNs offer the possibility for search operations on their social graphs, provide unsolicited friendship recommendations, and offer support for non-authorized linkage of a friend's friends. This causes a multitude of unwanted friendship establishments, i.e., links in the social graph which not necessarily represent a real friendship. It should be stressed that Vegas does not prohibit friends that do not represent human identity. A Vegas friend can also map to the profile of a company or any organization from the civilian domain.

Figure 1 illustrates the communication model of Vegas. Each user interacts with the OSN through one or more mobile or stationary clients. Vegas applies an asynchronous message exchange scheme based on the concept presented in [4]. We rely on well known services like email, SMS, or instant messaging which can be exploited to implement the *exchanger* instance. An exchanger represents

the abstract concept of a message queue which is used to transmit messages or any other kind of content. Any two Vegas friends A and B are aware of one or more such exchanger addresses of each other. A datastore represents the abstract concept of a user-writable storage space with world-readable access (e.g. some web space). Each user provides one or more datastores to place individually encrypted and signed profile for each of his friends.

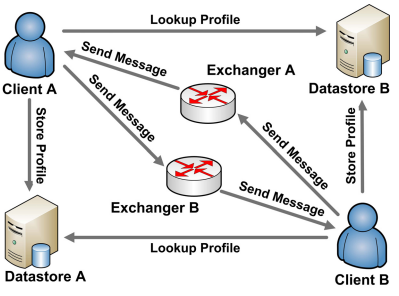

Fig. 1. In Vegas all exchanged information is encrypted and signed. Each user maintains one or more client instances and performs encrypted messaging over one or more exchanger instances. A user publishes individual profiles for each friend at one or more datastores.

2.2 Vegas Operation

Vegas communication and profile distribution works as follows: Two Vegas friends A and B generate a unique public key pair which must not be applied for messaging and profile generation except in the context of A and B. We term such a key pair a *link-specific* key pair. As user A holds a unique key pair $K^{-}_{A\to X_i}/K^{+}_{A\to X_i}(i \in 1,\ldots,n)$ for each of his n friends $X_1,...,X_n$, a key pair simply represents a directed edge in the overall social graph. The notion of a key $K^{-(+)}_{A\to X_i}$ means that this key is a private (public) key generated by A for exclusive communication with X_i. A utilizes X_i's public key $K^{+}_{X_i\to A}$ to encrypt messages as well as profile information intended for X_i. In order to allow X_i to map a received message to its originator A, a fingerprint of A's public key $K^{+}_{A\to X_i}$ is included into each message sent to X_i. In case A wants to send a message to X_i, A applies X_i's public key $K^{+}_{X_i\to A}$ to encrypt the message content. After signing the message with $K^{-}_{A\to X_i}$, A delivers this message via an exchanger to X_i. X_i identifies sender A through his attached public key fingerprint. Since X_i is the only user that knows about this fingerprint, he represents the only user that is able to map it to the identity of A. We apply the same Vegas operations for the placement and update of profile information which we use to send messages.

2.3 Cross-Layer Protocol

As we aim at individual advertising and the deployment of personal and commercial services on top of ubiquitous WiFi APs, we necessitate a novel

communication protocol which allows for secure services that smoothly integrate into Vegas. Figure 2 illustrates our protocol including all interactions between the involved parties. For simplification, we use the same identifiers C, AP, and S to either refer to a service (or a device) involved in the protocol or to the organization or person operating the correspondent service. At the beginning, a service provider S has to configure an access point AP that is envisaged to advertise a certain service (1). First, AP generates a public key pair K^-_{AP}/K^+_{AP} and presents S with the public part. In case S does not trust AP, AP can optionally provide K^+_{AP} encapsulated as a certificate signed by a trusted CA. S then generates a certificate $c_{(K^-_S)}(K^+_{AP})$ from K^+_{AP} (e.g. signed by his own certificate $c_{(Self)}(K^+_S)$) and sends $c_{(Self)}(K^+_S)$, $c_{(K^-_S)}(K^+_{AP})$, a service identifier ID_S, and the actual content M that will be advertised back to AP. To support mobile users in choosing relevant offers, ID_S also includes a semantical description of the service. A user C that wants to recognize and validate service advertisements from S has to establish a Vegas friendship with S in advance. To become Vegas friends, C and S rely on a (semi-) trusted out-of-band (OOB) channel to exchange their public keys ($K^+_{S\to C}$, $K^+_{C\to S}$), their exchanger addresses (Ex_S, Ex_C), and their datastore addresses (DS_S, DS_C) (2). In case C and S do not require detailed profile information of each other, exchanger and datastore addresses need not to be exchanged. Exchanging the public keys always suffices

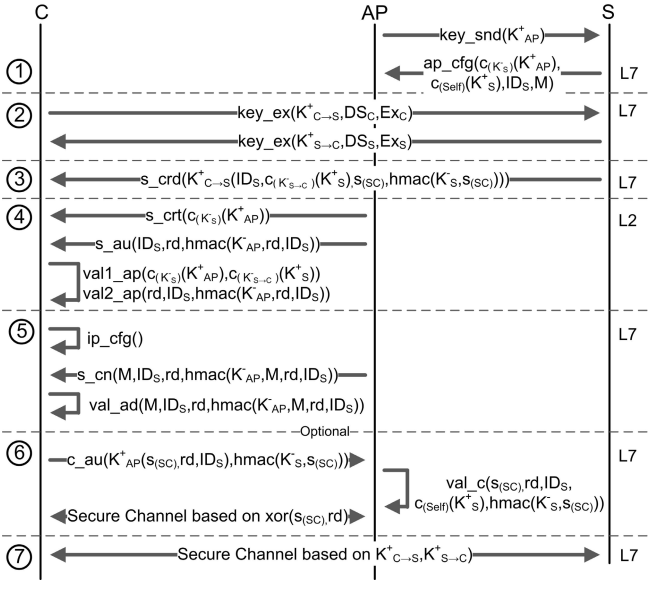

Fig. 2. The cross-layer protocol steps: *1)* Access point setup; *2)* Vegas key exchange; *3)* Provision of service credentials; *4)* Layer 2 service advertisement and authentication; *5)* Layer 7 service broadcast and integrity validation; *6)* Optional (advance services): User authorization; *7)* Optional (advance services): Service tunneling.

to facilitate service provider authentication and service consumer authorization. After C and S became Vegas friends, S sends an *s_cred* message to C including all information that is necessary to interact with AP (3). This always comprises ID_S and a certificate $c_{(K^-_{S\to C})}(K^+_S)$, which holds K^+_S and which is signed by the link-specific private key $K^-_{S\to C}$. Optionally, *s_crd* includes a shared secret $s_{(SC)}$ and a cryptographic hash $hmac(K^-_S, s_{(SC)})$ (generated based on K^-_S) which can be used as credentials to services beyond a simple advertisement service (see step 6). After AP has been configured it starts to broadcast service advertisements (4). We decided to overload IEEE 802.11 beacon frames with new IEs which carry all information necessary to authenticate the advertised service (see Section 2.4). Independent of the advertised service, AP repeatedly broadcast certificate message *s_crt* which includes $c_{(K^-_S)}(K^+_{AP})$. The service identifier and credentials are broadcasted in a separate message *s_au* which consists of the service identifier ID_S, a random value rd, and an cryptographic hash $hmac(K^-_{AP}, rd, ID_S)$ over both values based on K^-_{AP}. rd will change periodically and can be used to identify advertisements replayed by an attacker long after the advertised service shut down. AP always keeps a list of the most recent values of rd as this value can also be used by C during his authorization (see step 6). We separate the broadcast of AP's certificate $c_{(K^-_S)}(K^+_{AP})$ from S's service advertisements. This prevents redundancies as S may decide to broadcast advertisements for more than one service via AP. As soon as C comes into radio range of AP, C can receive the overloaded beacon frames. In case C recognizes a service description ID_S and already received a complete copy of $c_{(K^-_S)}(K^+_{AP})$, C can easily perform an authenticity and integrity check of the received advertisement: Procedure $val1\_ap$ validates the signature of $c_{(K^-_S)}(K^+_{AP})$ by applying $K^+_{C\to S}$ to $c_{(K^-_{S\to C})}(K^+_S)$ and K^+_S to $c_{(K^-_S)}(K^+_{AP})$ and procedure $val2\_ap$ validates the advertisement by recalculating the cryptographic hash of rd and ID_S and comparing the result to $hmac(K^-_{AP}, rd, ID_S)$. Only in case both validations succeed, a service description can be considered authenticated. In case validation succeeds, C applies procedure ip_cfg which configures an IP address. This is critical, since IP configuration, e.g. via DHCP, might demand for upstream Layer 2 authentication. However, the upcoming IEEE 802.11u standard will strongly simplify this task. By calling procedure *val_ad*, C can authenticate a service advertisement content M of any broadcast message *s_cn* that is related to ID_S. AP just has to ad ID_S, rd, and a cryptographic hash $hmac(K^-_{AP}, M, rd, ID_S)$ to *val_ad* in order to allow C to validate the hash. Due to the possibility to multiplex services with distinct service identifiers, steps (1) – (5) already suffice to facilitate simple services like community-centric and privacy-preserving advertisement, voucher, and coupon broadcasts. In case C also received a shared secret $s_{(SC)}$ and a cryptographic hash $hmac(K^-_S, s_{(SC)})$ (step 3), C optionally can access advanced services (6). To prove his service access authorization, C sends a message *c_au* which includes all credentials encrypted with K^+_{AP} necessitated by AP. AP then calls a procedure *val_c* which identifies the requested service by ID_S, recalculates the $hmac(K^-_S, s_{(SC)})$ to validate the originator of $s_{(SC)}$, and verifies that this

is no replay by proving the freshness of rd. Now C and AP can both calculate a shared key $xor(s_{(SC)}, rd)$ which can be used to establish a secure channel. As long as AP and S are not operated by one and the same identity, AP cannot infer further information about C. Assuming an Internet connection to S and the case where a service requires detailed profile information about C, it is even possible to establish a secure channel between C and S (7).

2.4 Information Element Structure

To facilitate Layer 2 service advertisement recognition and authentication, we decided to overload standard IEEE 802.11 beacon frames by introducing two new IEs. We utilize the *LB-SNS Certificate Fragment* (LCF) IE to broadcast AP certificates and the *LB-SNS Identity and Authentication* (LIA) IE to broadcast service identifiers and the corresponding authentication information (see Figure 3a). To indicate a custom ID, the field *Element ID* is set to `0xDD` and the field *User ID* to `0x123456`. The *Length* field indicates the width of the Value field, which has a maximum size of 252 exclusive the User ID field. Value fields of the LCF and LIA IEs are depicted in figures 3b) and 3c). LCF and LIA both include a *Type* field which is used to distinguish LCF IEs from LIA IEs. As a commonly applied public key already has a size of 2048 bit and a Value field is limited to 252 byte, we apply LCF IEs to broadcast only fragments or AP certificates $c_{(K_S^-)}(K_{AP}^+)$. To be able to reassemble a certificate, each LCF IE carries an identifier *Cert ID* to map the LCF IE to the corresponding certificate, a sequence number *Seq ID* to describe the ordering of the fragments, a fragmentation flag *Flag* which indicates the end of a certificate, and the certificate fragment *Cert Frag* itself. Service advertisements are carried in the LIA IE which holds an identifier *Cert ID* indicating the certificate $c_{(K_S^-)}(K_{AP}^+)$ that must be used to prove AP authenticity, a service identifier *Service ID* and a random value *Rand* field, and an *HMAC* field, which holds a cryptographic MD5 hash $hmac(K_{AP}^-, rd, ID_S)$ to prove authenticity of the Service ID and Rand fields.

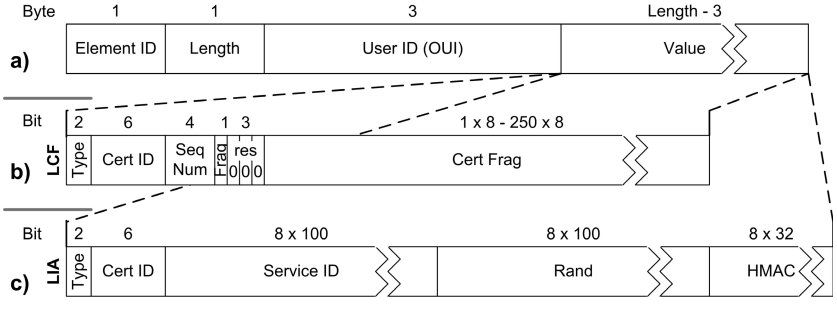

Fig. 3. New IEs: *a)* Common structure of the new information elements (IEs); *b)* Structure of the *LB-SNS Certificate Fragment* (LCF) IE; *c)* Structure of the *LB-SNS Identity and Authentication* (LIA) IE.

Due to space constraints, we omit a detailed explanation of the Service ID field which is necessary to recognize and multiplex advertised services.

References

1. Chandra, R., Padhye, J., Ravindranath, L., Wolman, A.: Beacon-Stuffing: Wi-Fi Without Associations. In: Hotmobile 2007 (2007)
2. Dürr, M., Werner, M., Maier, M.: Re-Socializing Online Social Networks. In: Proc. of GreenCom–CPSCom 2010. IEEE (December 2010)
3. Dürr, M., Marcus, P., Wiesner, K.: Secure, Privacy-Preserving, and Context-Restricted Information Sharing for Location-based Social Networks. In: ICWMC 2011, pp. 243–248. IARIA (June 2011)
4. Werner, M.: A Privacy-Enabled Architecture for Location-Based Services. In: Schmidt, A.U., Russello, G., Lioy, A., Prasad, N.R., Lian, S. (eds.) MobiSec 2010. LNICST, vol. 47, pp. 80–90. Springer, Heidelberg (2010)