2012 ASE/IEEE International Conference on Social Computing and 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust

Vegas - A Secure and Privacy-Preserving Peer-to-Peer Online Social Network

Michael Dürr, Marco Maier and Florian Dorfmeister
Ludwig-Maximilians-University Munich
80538 Munich, Germany
Email: {michael.duerr,marco.maier,florian.dorfmeister}@ifi.lmu.de

***Abstract*—Although Social Network Service (SNS) providers like Facebook and Google attempt to mitigate security and privacy-related concerns of their users, abuses and misuses of personal data still make the headlines. As centralized storage of personal data is a decisive factor for unintended information disclosure, several architectures for decentralized Online Social Networks (OSNs) have been proposed. System designs range from solutions based on a decentralized client server architecture like Diaspora to P2P systems like PeerSoN. Despite all efforts to accomplish strong decentralization, most proposals cannot achieve sufficient informational self-determination, i.e., users do not have full control over storage and dissemination of their personal data and published content.**

In this paper we follow a contrary approach and present Vegas, a secure and privacy-preserving P2P OSN which restricts the possibility to browse the social graph to the ego network. We show how Vegas achieves a maximum degree of security and privacy through encryption and decentralization. We present our mobile Vegas prototype and its context-dependent communication channel decision model. Finally we show how Vegas can be extended to support services like social-search and directory services in a secure and privacy-preserving way.

***Keywords*-Online Social Networks; Privacy; Security; Trust;**

I. Introduction

Centralized *Online Social Networks* (OSNs) such as *Facebook*[1] and *Google+*[2] have become the predominant communication channel in both the personal and the business domain. However, a plethora of recent examples has shown that intense interactions within centralized OSNs lead to unintended social information dissemination and offer opportunities for social data abuse [1]–[3]. Due to these observations, several attempts have been made to develop a decentralized OSN which complies with certain security and privacy demands. Recent examples comprise Peer-to-Peer (P2P) OSNs like *PeerSoN* [4] and *Safebook* [5] as well as ambitious web projects like *Diaspora*[3]. Unfortunately non of these systems can be considered a secure and privacy-preserving alternative for OSNs like Facebook and Goolge+. Most of these systems attempt to achieve the same user experience as known from centralized OSNs, whereas certain functionality, e.g, seeking for friends, only works at the cost of less privacy.

As we believe that it is not possible to provide for a decentralized OSN with the same functionality as known from their centralized counterparts, we take a contrary approach. We develop a highly restrictive P2P OSN which provides for a maximum degree of privacy but, in its core specification, does not attempt to achieve the functionality of Facebook and Co. Instead we allow for optional relaxations of our design in order to provide for enhanced features like social search and social directory services.

In this paper we present Vegas, a highly restrictive, secure, and privacy-preserving OSN which allows for domain specific relaxations dependent of the desired degree of security and pricacy.

The remainder is structured as follows. In Section II we address preliminaries that represent the foundation for our design choice. Related work is presented in Section III. We give a detailed picture of the Vegas architecture in Section IV before we present our mobile prototype and its channel decision model in Section V. We sketch the extendability of Vegas to facilitate enhanced services by introducing further relaxations in Section VI before Section VII concludes this work.

II. Preliminaries

To identify architectural peculiarities of a secure and privacy-preserving OSN, we already identified inevitable preliminaries in our previous work [6]. It should be stressed that these preliminaries stem from our stringent requirements for security and privacy. To better understand our design decision, we give a brief summary of these preliminaries.

A. Informational self-determination

We consider the requirement of informational self-determination to be fulfilled in case a user has full control over access, storage, and distribution of his personal profile information and content published within an OSN. Although it is not possible to remove all copies of once published content from the Internet, we demand that a user is able to modify or remove personal data from the OSN at any time. The reason for this requirement is that social network service (SNS) providers cannot be expected to always comply with their guarantees on a privacy-preserving treatment (e.g. the deletion) of once published personal data [7].

B. Strong trust relationships

Recently it was shown that the static structure of a social graph cannot be used to infer an accurate picture of the actual

[1]http://www.facebook.com/
[2]https://plus.google.com/
[3]https://joindiaspora.com/

social relationships of its nodes [8]. In general a large fraction of links map to relationships that, in case you only refer to the recent emergence of communication between the connected nodes, cannot be considered an existent relationship any more. The phenomenon can be observed in *real social networks* (RSNs): it simply takes too much time and effort to keep up an unlimited amount of friendships over time [9]. A social graph may also comprise links which map to relationships that have been created mistakenly due to the thoughtless acceptance of unsolicited friendship request. We termed this phenomenon *social network pollution* [6]. As security and privacy within an OSN heavily depend on the degree of trust between friends, we demand strong trust relationships, i.e., we want our OSN design to better map social relationships we are used to maintain in reality to their digital counterparts. For that reason we prevent from friending with other participants that have not met each other or do not know any kind of identifier such as an email address or a telephone number before.

C. Profile availability

Due to our demand for informational self-determination each user has to administer his personal profile and content on his own. Since we focus on a decentralized solution, users must be expected to go on- and offline at any time. As an OSN is worth nothing if personal profile information is not permanently accessible to friends, we demand a 24 hours a day profile availability even in case the corresponding user is offline.

D. Mobility support

The utilization of OSNs on mobile devices increases steadily and social, context-aware applications such as Facebook Places or Goolge Latitude become very popular. A secure and privacy preserving OSN should therefore account for mobility support in terms of mobile communication and extendability.

III. Related Work

Although several architectures for decentralized Online Social Networks (OSNs) have been proposed, an architecture which complies with our preliminaries is still missing. In the following we review decentralized OSN designs which focus on security and privacy.

Security in Safebook [5] is based on the principles of decentralization and trust among friends. The architecture uses *matryoshkas*, concentric rings of nodes, that enable trustful data storage and retrieval as well as communication obfuscation for their shared center node. A DHT substrate offers a lookup service to find entry points for other users' matryoshkas. A trusted identification service (TIS) ensures trustful communication and user privacy within the DHT. Besides its limited guarantees on availability, a DHT implicates additional management overhead as it complicates the OSN protocol, causes additional signaling traffic, necessitates caching, and suffers from a weaker trust model. To guarantee an uninterrupted chain of trust, each request forward requires message decryption, signature re-calculation, and message re-encryption. The TIS prevents impersonation and sibyl attacks but represents a centralized third party which may not be acceptable for users of a decentralized OSN. The authors state that this may be implemented offline but do not explain how. Safebook nodes can seek and request access to any profile and therefore facilitate unsolicited friendship requests. This fact represents the foundation for OSN pollution.

PeerSoN [4] is a P2P OSN that focuses on secure authentication, encryption, and the prevention of impersonation attacks. Peers need not necessarily be connected to the Internet to make use of their social network. However, this is restricted to insight communication with other PeerSoN enabled devices. Profiles of other users cannot be accessed in this case. Since PeerSoN offers a DHT-based lookup service, problems similar to that of Safebook exist. To identify users, PeerSoN applies global unique identifiers (GUID) built from the hash of an email address or a public key. Unfortunately hashing does not prevent from successful traffic flow analysis and therefore cannot assure informational self-determination. A node that issues recurring requests for one and the same hash could derive certain information about a user.

Vis à Vis [10] also follows a DHT approach where users provide their data through virtual individual servers (VISs) operated in the cloud. As administrative tasks are shifted from a centralized OSN to a centralized VIS provider this system still does not comply with our demand for informational self-determination.

Lockr [11] improves privacy for centralized and decentralized online content sharing services by reducing the chances for mismanagement or accidental disclosure of social networking information. Although Lockr strictly separates between the management of social relationships and shared content, it is still possible to map content that is shared among ordinary OSNs like Facebook to anonymous OSN identities. The content itself is also stored in centralized OSNs, i.e., informational self-determination cannot be achieved.

Diaspora [12] is a web-based decentralized OSN which is run on a network of connected servers called pods. Users can either join an existing pod of a friend, a pod of a non-profit organization like a university, or an official pod run by the project's founders. As users can also operate their personal pod, Diaspora provides the basis that is necessary to support informational self-determination. However, Diaspora still facilitates searching for unknown members, supports unsolicited friendship requests, and therefore, partially allows for the disclosure of the social graph.

IV. Vegas design

To accomplish our requirements we maximize the degree of security and privacy at the cost of functionality. Figure 1 illustrates an example view of the Vegas social graph. The view of user n_i is limited to his neighborhood $\Gamma(n_i) = \{k_1, k_2, k_3\}$, i.e., n_i does not know anything about the relationship between two friends $k_i, k_j \in \Gamma(n_i)$ or between a friend $k_i \in \Gamma(n_i)$ and any other user $v_i \in V \setminus (\{n_i\} \cup \Gamma(n_i))$. In the following we present

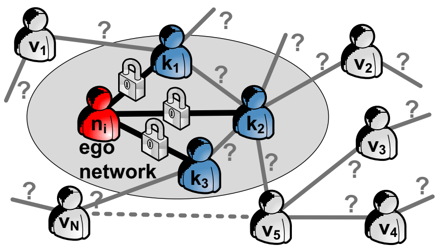


Fig. 1: Vegas social graph from the user perspective.

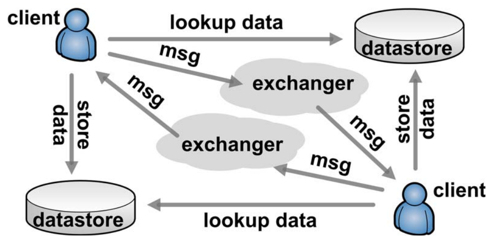


Fig. 2: Architectural components of Vegas.

the details that are necessary to accomplish this restricted design.

A. Overview

Figure 2 depicts a high level representation of the architectural components necessary for data access and information exchange in Vegas. A user interacts with the OSN through one or more mobile or stationary *client* devices. Clients communicate through *exchangers*, secure asynchronous communication channels that allow for delay-tolerant information exchange similar to a mail box. As Vegas suffers from P2P anomalies like churn, we introduce *datastores* for device synchronization as well as for permanent user profile and data availability.

B. Message security

To comply with our demand for strong trust relationships and informational self-determination, we apply an asynchronous message exchange scheme that is based on the concept of *locagrams* [13]. In contrast to ordinary public key cryptography, in Vegas, each user A holds a unique public key pair $K^-_{A\rightarrow X_i}/K^+_{A\rightarrow X_i}(i \in 1,\dots,n)$ for each of his friends $X_1,...,X_n$. Since A exclusively provides X_i with a unique public key, we denote the corresponding key pair a *link-specific* key pair. Due to its uniqueness, the public key can be considered a directed edge in the underlying social graph.

In case A wants to send a message to X_i, A applies X_i's public key $K^+_{X_i\rightarrow A}$ to encrypt[4] the message content, signs the message with $K^-_{A\rightarrow X_i}$, adds the fingerprint of $K^+_{A\rightarrow X_i}$, and sends the message to an exchanger accessible by X_i. X_i can fetch this messages and identify sender A through the attached public key fingerprint. Since X_i is the only user that knows about the mapping of the included fingerprint, nobody else can map this fingerprint to the identity of A.

It is obvious that this approach necessitates the management of three times more keys than a traditional PKI. Assuming n friends, each user has to manage $2n$ public and n private keys instead of $n+1$ public and 1 private key. For two reasons we consider this overhead acceptable. *a)* We expect the average Vegas user to maintain far less friendships compared to OSNs like Facebook as there exist less opportunities to establish unsolicited friendships. *b)* The complexity of public key revocation does not exist as this task reduces to a simple key delete operation. In case A considers the mapping of a public key to X_i's identity to be compromised, A can trigger a key refresh operation in order to replace the two key pairs utilized in conjunction with X_i.

C. Message exchange

We introduce exchangers which represent the abstract concept of a message queue that is used to transmit messages. If two users decide to become friends, they not only exchange their link-specific public keys but also one or more exchanger addresses. In case user A wants to send a message to one of his friends $X_1,...,X_n$ he selects an appropriate exchanger to send his message. The selection process depends on certain demands such as short transmission delay, high transmission bandwidth, high degree of anonymity, or small monetary costs. Although we expect an exchanger to behave like a mailbox, any kind of push- or pull-based messaging technology can be used. Our present implementation supports emailing, short messaging (SMS), instant messaging (XMPP) and even micro blogging (*Twitter*[5]).

D. Data provision

In order to comply with our requirement for profile availability we introduce the concept of datastores. A datastore represents the abstract concept of a user-writable storage space with world-readable access. In order to increase the degree of decentralization, each user can operate an arbitrary number of datastores to permanently provide his friends with profile information and other shared content. Datastores also serve for the purpose of synchronization between multiple client devices. Each piece of data (profile attributes, sent and received messages, simple URLs, ...) is stored as a single encrypted file with a randomly generated name and within a randomly generated directory. As only the user and his friends know about the necessary datastore addresses and the directory structure to content mapping, Vegas can significantly decrease opportunities for deanonymization attacks. Datastores can be implemented as a web resource like FTP, WebDAV, or be deployed at a cloud storage service like *Amazon S3*[6], *Google Drive*[7], or *Dropbox*[8].

[4]When we refer to public key encryption, we always consider a hybrid approach. Data is encrypted based on a symmetric key which in turn is encrypted in an asymmetric way.

[5]http://twitter.com/
[6]http://aws.amazon.com/s3/
[7]https://drive.google.com/
[8]http://www.dropbox.com/

E. *Profile synchronization*

As mentioned before datastores also serve the purpose of profile synchronization, as a user must be expected to access an OSN from more than a single client device. We apply the same Vegas operations for the placement and update of profile information which we use to send messages. In case user A wants to update his profile attribute $attr_i(A)$, he generates a new symmetric key K_s, re-encrypts $attr_i(A)$ with K_s and asymmetrically encrypts K_s for each of his friends $X_1, ..., X_n$. Finally, A updates the file representing the profile attribute with the content $K_s(attr_i(A)), K^+_{X_1 \to A}(K_s), \ldots, K^+_{X_n \to A}(K_s)$ at the corresponding datastore(s). As A utilizes the same key material to encrypt messages as well as profile information for X_i, X_i can simply perform an interval-based read operation at A's datastore to receive pending updates. As an alternative, A can trigger such an update by sending each of his friends a corresponding notification via the friends' exchangers.

F. *Key refresh*

Similar to PKIs, in Vegas, it can happen that a link-specific public key pair must be revoked. As (the fingerprint of) a link-specific public key also serves as an anonymous identifier for the corresponding friend, it must already be considered compromised if the mapping between the public part and the issuer of the key was disclosed. As a link-specific public key is utilized for exclusive communication with one friend, the complexity of key revocation reduces to key deletion. To agree on new link-specific keys, we implemented a simple key refresh protocol, which allows for an in-band key negotiation and exchange.

G. *Friendship establishment*

As we demand strong trust relationships, in its core specification, Vegas does not facilitate to browse the social graph to seek for other users. The visibility of the social graph is restricted to the ego network. Two options to establish a new friendship exist.

1) Out-of-band invitation: The easiest (and most secure) way to become friends is a face-to-face exchange of the link-specific keys and exchanger addresses. Data can be exchanged ad-hoc based on an encrypted NFC or Bluetooth connection. An optical out-of-band (OOB) solution (which is implemented in our Android prototype) is given by a face-to-face exchange of QR-codes. In case Vegas users A and B want to become friends, user A triggers the generation of a *friendship request* QR-code including a temporary public key $K^+_{A \to B}$ and an exchanger address $Ex(A)$ on his device. User B takes a photo of the QR-code, decodes the information and generates a *friendship response* QR-code including a public key $K^+_{B \to A}$ and an exchanger address $Ex(B)$. A in turn takes a photo of that QR-code and also decodes the included information. To prevent from replay attacks, A and B execute an in-band key refresh operation and negotiate new public keys.

Another less trustful possibility is our email-based friendship establishment mechanism which has already been described in our previous work [6].

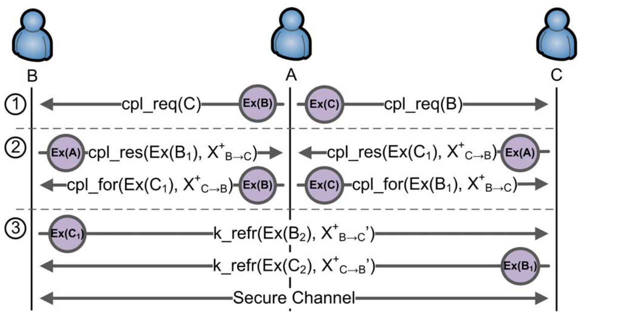


Fig. 3: User A initiates the coupling of his friends B and C.

2) Coupling: It is a quite common situation that, after person A has introduced two of his friends B and C to each other, A wants B and C to establish a Vegas friendship as well. To map this procedure to Vegas, we implemented *coupling*, a simple protocol which complies with the demand for strong trust relationships. Figure 3 illustrates the basic protocol. The notation $Ex(X)$ within a circle means that a message is sent via an exchanger provided by X. Such a message is also encrypted by a link-specific key.

To initiate coupling between B and C, A sends a coupling request to both of them. The request contains parts of the profiles of B and C that they have marked as public (1). In case B and C accept A's offer, both create a link-specific public key and an exchanger address to be used during coupling. After A received both responses, A forwards the re-encrypted messages (2). To comply with our demand for strong trust relationships, B and C have to perform a key refresh (3) as A knows about the link-specific public keys utilized during the coupling procedure. Now it depends on the trust between A and B as well as B and C whether a key refresh on the insecure channel can be trusted without another OOB information, ensuring that A cannot act as a man-in-the-middle.

H. *Discussion*

It is obvious that, dependent on the data and the number of affected friends, the application of strong cryptography can cause considerable overhead. Nevertheless we consider this overhead acceptable. On the one hand profile attributes represent rather static information, i.e., they are not subject to frequent changes. On the other hand Vegas does not dictate the way shared content becomes published within the OSN. In many cases shared content is less sensitive, i.e., it already suffices to provide friends with an encrypted link instead of encrypted content. The fact that a user cannot be linked to some referenced content may already significantly increase his anonymity. Finally we assume that, although confidential content has to be encrypted, the number of friends, i.e., the overhead for encryption, decreases dependent on the degree of required confidentiality. The more confidential the information is the less friends will be elected for provision of that information.

Synchronization tasks for multiple clients can be mapped to profile updates for friends. In case user A has modified certain attributes of his profile, *A* marks these attributes as updated. Each client device that is used by *A* to access Vegas performs an interval-based read operation in order to check for recent modifications.

V. Channel selection

As the utilization of OSNs more and more shifts to the mobile domain [14], mobility support is an inevitable demand for a decentralized OSN. In context of Vegas, there are two aspects that have to be considered. On the one hand Vegas attempts to enhance security and privacy by supporting different exchanger implementations based on email, SMS, micro blogging, or instant messaging. On the other hand mobility support comes at the cost of limited bandwidth and in general additional monetary expenses. As it depends on individual preferences which factor should dominate, we demand a decision model which allows for semi-automated channel selection dependent on the individual context.

A. Decision criteria

We present four criteria to classify communication channels for our exchanger decision model.

1) Messaging overhead: In context of this work messaging overhead refers to any kind of overhead that occurs due to the specific characteristics of the considered communication channel. This not only comprises the necessity for encryption, the generation of additional header information, or the fragmentation of messages but also additional measures that have to be taken due to the pull- or push semantics of the considered channel.

2) Reliability: Although messaging in Vegas is asynchronous and provides for application layer acknowledgements (ALAs), the considered communication channels may rely on a synchronous protocol. We consider the reliability of a channel an important criteria as, due to our privacy demands, ALAs may be sent via distinct exchangers. In case a user requires instant feedback on successful message delivery, he might choose an SMS instead of an email in order to deliver the message.

3) Monetary costs: In case a user has no WLAN coverage but relies on a mobile connection, the utilization of Vegas can cause additional monetary costs. Therefore a user must be able to influence the selection of an exchanger dependent on the utilized channel.

4) Traceability: Traceability refers to the ability of a third party (e.g. the mobile network provider) to link an observed message to the sender or the receiver of a message. As Vegas focuses on privacy it must be possible to influence the exchanger selection process based on the traceability of a channel.

B. Channel classification

In its present implementation, Vegas utilizes email, SMS, XMPP and Twitter as communication channels for message exchange. In the following we discuss these channels with regard to our decision criteria before we detail our decision model.

1) Email: Email communication relies on a decentralized structure and gives its users the freedom to choose from a large set of service providers. Its application for exchangers restricts the ability to trace Vegas communication to the service provider of the corresponding sender or receiver address domain. Except from encryption and signing, a Vegas message necessitates no additional processing, as most providers allow emails with several megabytes of size. Email can be regarded a reliable channel as long as Vegas messages are not filtered due to a misconfigured spam filter. Due to its pull semantics, email should be considered for delay-tolerant messages.

2) XMPP: The *Extensible Messaging and Presence Protocol* (XMPP) was designed to work within a decentralized network structure. In contrast to email, XMPP facilitates small delivery times, as it was designed with push semantics to support near-real-time instant messaging. This however necessitates a persistent TCP connection which may be not desired in case of a mobile Internet connection. As long as Vegas messages do not exceed server-sided XMPP message size limits, additional processing like message splitting is not necessary.

3) Twitter: The micro blogging service Twitter significantly differs from the previous examples as it was designed for unidirectional message exchange. As messages are publicly accessible via the Twitter platform itself, this channel allows for a completely anonymous reception of Vegas messages. Anybody can read the content of the Twitter pinwall. However, only a Vegas user aware of the identity of a post can identify the sender and decrypt the message content. Twitter messages are limited to 140 characters, i.e., a Vegas message has to be split into several Twitter tweets in order to be transmitted. In addition, due to the restriction of a limited number of posts, Twitter must be considered an unreliable channel.

4) SMS: As SMSs cannot include more than 160 characters, Vegas messages have to be split into several SMS to get transmitted. Although SMSs cause additional costs they represent an appealing channel for message exchange. On the one hand SMS is the most widely used data application in the world [15]. Especially within countries that only provide for limited or no Internet access at all, SMS represents the predominant means for communication. On the other hand SMS is a real OOB channel. Some fragments of a Vegas message may be sent via Twitter, others via SMS which can increase sender privacy and security.

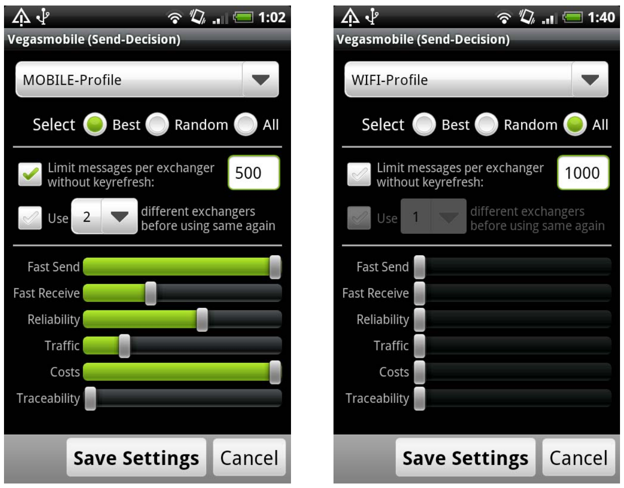
C. Decision model

Although there exist sophisticated methods based on, e.g., fuzzy logics [16] or decision trees [17], we decided not to model our channel decision on an approach from such a domain. On the one hand we want to reduce the complexity of our decision model in order to account for the limited resources of a mobile device. On the other hand there is no simple measure which allows to automatically weight the importance of, e.g., monetary costs or non-traceability, in order to determine certain fuzzy rules or to infer a decision tree.

We apply a simple weighted arithmetic mean as it facilitates a user to influence the channel decision by directly increasing or decreasing the impact of the aforementioned channel characteristics. Be K the set of all channels k and C the set of characteristics c of a channel k. Be $f_c(k)$ a function which determines a metric value for characteristic $c \in C$ of channel k. Our weighted function then calculates as

$$W(k) = \frac{\sum_{c \in C}(w(c) * f_c(k))}{\sum_{c \in C} w(c)}$$

where function $w(c)$ determines the weight of channel c. To provide for a reasonable mapping $f_c(k)$, further information about certain channel characteristics are required. For instance, monetary costs heavily depend on the data plan of a certain user. As it is not easy to determine a dedicated mapping of the considered channel characteristics in advance, we based our function $f_c(k)$ on the example weighting depicted in table I. Two example configurations for a send decision of our prototype are illustrated in figure 4. The prototype further separates messaging overhead into *fast send*, *fast receive* and *traffic*. This distinction is driven by the push and pull semantics of the different channels.



Vegasmobile (Send-Decision)

MOBILE-Profile

Select [✓ Best] Random All

✓ Limit messages per exchanger without keyrefresh: 500

✓ Use 2 different exchangers before using same again

Fast Send
Fast Receive
Reliability
Traffic
Costs
Traceability

Save Settings Cancel

Vegasmobile (Send-Decision)

WIFI-Profile

Select Best Random [✓ All]

Limit messages per exchanger without keyrefresh: 1000

Use 1 different exchangers before using same again

Fast Send
Fast Receive
Reliability
Traffic
Costs
Traceability

Save Settings Cancel

Fig. 4: Screenshots from the Vegas send decision configurator for a 3G (left) and a WLAN (right) connection.

VI. Relaxations

As Vegas focuses on a maximum degree of security and privacy, the core specification attempts to prevent the unsolicited distribution of personal information outside of the ego network. However, there exist several scenarios where users can benefit from relaxations to our design. In the following we sketch an example of a relaxation to illustrate its advantage.

TABLE I: Example classification of communication channels applied in Vegas ($++ :=$ high, $-- :=$ low).

Channel	*Email*	*SMS*	*XMPP*	*Twitter*
Messaging overhead	$--$	$++$	$-$	$+$
Reliability	$++$	$-$	$+$	$--$
Monetary costs	$+$	$++$	$+$	$+$
Non-Traceability	$-$	$+$	$-$	$++$

A major driver for the utilization of OSNs is the ability to perform *social search*. By social search we mean the ability to formulate the query for any kind of information which may be found more easily by restricting the request to a certain group of people that are, from a social viewpoint, somehow related to the query. It is obvious that Vegas only allows to perform social search within the ego network.

In our previous work [18] we presented a simple query-based forwarding mechanism which facilitates secure and privacy preserving forwarding of queries outside the boundaries of the ego network. In a nutshell a modified version of that mechanism works as follows: The originator A of a query determines a subset of his friends $X_1, \ldots, X_n$ whom he considers candidates that may be able to answer his question. A friend $X_i \in \{X_1, \ldots, X_n\}$ that is able to give an answer to the query can directly respond to A. In this case messaging is secure as it is based on the Vegas messaging mechanism. X_i can also decide to forward the query to one or more of his friends $Y_1, \ldots, Y_m$. In order to preserve A's anonymity X_i re-encrypts the query and forwards it to his friends. In case a friend $Y_i \in \{Y_1, \ldots, Y_n\}$ can answer the query, Y_i can reply to X_i. Again this communication is secure as it solely depends on $X_i's$ and $Y_i's$ link-specific key pairs. Privacy is preserved as Y_i cannot recognize the originator of the query. It should be mentioned that we do not make any assumption regarding the forwarding metric. This can be based on a simple social metric like the number of friends or on a more sophisticated approach which is based on a matching of common semantics of profile attributes and the query content.

This example shows that a simple relaxation of the core concept can already facilitate a higher degree of functionality. Another example for a relaxation represent *Directory Buddies* which can be used to provide for services like privacy preserving and secure exchange of business cards [19].

VII. Conclusion

Vegas represents a peer-to-peer Online Social Network (OSN) which focuses on security and privacy. Although the restrictive design has strong impact on the functionality of Vegas, we showed that there exist several reasons for such an approach. Our exchanger model enables for high anonymity and can be configured with an easy to use and simple to understand channel decision model. Due to its trust model certain relaxations allow for an easy integration of extended functions.

As Vegas attempts to provide a completely decentralized and secure OSN, it is not possible to examine its social graph structure and its users' interaction behavior. Nevertheless it is essential to gather such information to figure out reasonable settings for our decision model, to choose the best social query routing metric, and to deploy the most efficient social search algorithms. Therefore, in our future work, we plan to extend our prototype with a logging framework which allows for the anonymous collection of this information.

References

[1] D. Irani, S. Webb, K. Li, and C. Pu, "Large online social footprints–an emerging threat," in *2009 International Conference on Computational Science and Engineering*. IEEE, 2009, pp. 271–276.

[2] G. Wondracek, T. Holz, E. Kirda, and C. Kruegel, "A practical attack to de-anonymize social network users," in *2010 IEEE Symposium on Security and Privacy*. IEEE, 2010, pp. 223–238.

[3] B. Krishnamurthy and C. E. Wills, "On the leakage of personally identifiable information via online social networks," *SIGCOMM Comput. Commun. Rev.*, vol. 40, pp. 112–117, January 2010.

[4] S. Buchegger and A. Datta, "A case for P2P infrastructure for social networks - opportunities & challenges," in *WONS'09*. IEEE, Feb 2009, pp. 161–168.

[5] L. A. Cutillo, R. Molva, and T. Strufe, "Safebook: A privacy-preserving online social network leveraging on real-life trust," *Communications Magazine, IEEE*, vol. 47, no. 12, pp. 94–101, Dec. 2009.

[6] M. Dürr, M. Werner, and M. Maier, "Re-Socializing Online Social Networks," in *Proc. of GreenCom–CPSCom'10*. IEEE, Dec 2010.

[7] M. Aspan, "How Sticky Is Membership on Facebook? Just Try Breaking Free," New York Times, Feb 2008. [Online]. Available: http://www.nytimes.com/2008/02/11/technology/11facebook.html

[8] C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy, and B. Y. Zhao, "User interactions in social networks and their implications," in *EuroSys '09*. ACM, 2009, pp. 205–218.

[9] E. M. Jin, M. Girvan, and M. E. J. Newman, "Structure of growing social networks," *Phys. Rev. E*, vol. 64, p. 046132, Sep 2001.

[10] A. Shakimov, H. Lim, L. P. Cox, and R. Caceres, "Vis-à-vis:online social networking via virtual individual servers," Duke University, Tech. Rep., May 2008.

[11] A. Tootoonchian, S. Saroiu, Y. Ganjali, and A. Wolman, "Lockr: better privacy for social networks," in *CoNEXT '09*. ACM, 2009, pp. 169–180.

[12] Diaspora, "Share what you want, with whom you want." September 2010. [Online]. Available: https://joindiaspora.com/

[13] M. Werner, "A Privacy-Enabled Architecture for Location-Based Services," in *MobiSec '10*, 2010.

[14] Socialbakers, "Facebook hits 488 million mobile users," Mai 2012. [Online]. Available: http://www.socialbakers.com/blog/554-facebook-hits-488-million-mobile-users-infographic/

[15] L. Whitney, "Cell phone subscriptions to hit 5 billion globally," February 2010. [Online]. Available: http://reviews.cnet.com/8301-13970_7-10454065-78.html

[16] L. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338 – 353, 1965.

[17] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986.

[18] M. Dürr and K. Wiesner, "A Privacy-Preserving Social P2P Infrastructure for People-Centric Sensing," in *KiVS'11*. OASIcs, 2011.

[19] M. Dürr, P. Marcus, and K. Wiesner, "Secure, Privacy-Preserving, and Context-Restricted Information Sharing for Location-based Social Networks," in *ICWMC'11*, 2011.