ENACT: Encounter-based Architecture for Contact Tracing

Aarathi Prasad*
Amherst College

David Kotz
Dartmouth College

ABSTRACT

Location-based sharing services allow people to connect with others who are near them, or with whom they shared a past encounter. Suppose it were also possible to connect with people who were at the same location but at a different time – we define this scenario as a **close encounter**, i.e., an incident of spatial and temporal proximity. By detecting close encounters, a person infected with a contagious disease could alert others to whom they may have spread the virus. We designed a smartphone-based system that allows people infected with a contagious virus to send alerts to other users who may have been exposed to the same virus due to a close encounter. We address three challenges: finding devices in close encounters with minimal changes to existing infrastructure, ensuring authenticity of alerts, and protecting privacy of all users. Finally, we also consider the challenges of a real-world deployment.

1. INTRODUCTION

Location-based sharing services allow people to connect with others who are near them, or with whom they shared a past encounter. An **encounter** is defined as an incident of co-location, i.e., two people who were at the same location at the same time. Detecting encounters allow people to get in touch with others they met in the past, for example at professional events, without having to exchange contact information [13, 19]. Suppose it were also possible to connect with people who were at the same location but a different time – we define this scenario as a **close encounter**, i.e., an incident of spatial and temporal proximity. By detecting close encounters, a person infected with a contagious disease could alert others to whom they may have spread the virus, a person who lost their belongings could reach out to others who may have discovered them, or people who were at the same event or location can share photos or videos with each other; in this paper, we will focus specifically on the problem of contact tracing, i.e., finding people who may have been exposed to a contagious virus.

*This work was done primarily when author was a graduate student at Dartmouth College

WPA'17, June 19, 2017, Niagara Falls, NY, USA

DOI: http://dx.doi.org/10.1145/3092305.3092310

Contagious viruses spread rapidly on college campuses, given the crowded and communal lifestyles of college students [4] – the most recent was a 2017 outbreak of 700 cases of norovirus in Southern California [8]. Contagious diseases that spread on college campuses include norovirus (stomach flu) [7], conjunctivitis [3], mumps [6], and measles [9]. When infected a person coughs, sneezes or breathes, he releases viruses in particle droplets that land on nearby objects and stay on surfaces for hours even after the patient leaves; the measles virus, for example, can survive on a surface for up to two hours [1, 9]. Touching an infected object transmits the viruses to a person's hand and then they become infected when they touch their nose or mouth with their unwashed hand. Handwashing is the key to prevent the spread of the virus – however, people remain unaware of their infected and contagious state because of the delay in symptom onset and so they cannot take precautions.

Typically when one patient is diagnosed with a contagious illness, institutional health organizations resort to campus-wide emails to alert the students, faculty and staff about the disease and encourage them to take precautionary measures [2, 5, 11]. However, most people do not understand their personal risk of infection and ignore the generic emails. Others panic and flood the health office with false reports. Most institutional health offices are not equipped (or staffed) to handle the many false cases reported as a consequence of the campus-wide alert [10]. To reduce the number of false reports and to help people better understand their own risk of getting the infection, the health office could also provide personalized information about an individual's exposure to the virus, in addition to the campus-wide alert.

We expect a smartphone-based system can retrieve the locations an infected person visited and notify those individuals whom they encountered. In this paper, we propose the ENACT (Encounter-based Architecture for Contact Tracing) system that leverages existing wireless infrastructure to detect close encounters, allowing the infected person to also notify the individuals who were at the same location as the infected person but at a different time.

To receive alerts about diseases and healthy lifestyle management, the students, faculty and staff on the college campus install a mobile application that registers their smartphones with the college health services. An alert includes the name of the disease, the symptoms the user should watch for, the duration during which the user should look for symptoms, the measures the user should take to stay healthy and avoid onset of the illness, and steps to take if they observe any of the listed symptoms. Typically, a similar alert is also sent as an email to all students, faculty and staff; those people who are not using our application will be informed about the spread of the virus via this email. The application facilitates contact tracing, by providing additional exposure information to those recipients who may have

been exposed to a contagious virus because of their spatio-temporal proximity to the infected user.

One approach for personalization is for the system to provide exposure information to anyone who may have visited the same location as the diagnosed patient. Location-specific alerts, however, may cause people visiting the location to panic and be concerned about visiting the location; in some cases, the location may be unavoidable (such as a classroom). Spatio-temporal personalization is better, as only those users who may have been exposed to the virus will receive personalized information; every other user using the application will receive a generic alert, with contents similar to the campus-wide email.

Although the broader effects of this idea on campus health will need further study via experimental deployment, this paper focuses on the technological foundations of the detection of close encounters. We make four contributions:

- We introduce a new problem: detecting close encounters. Close-encounter detection allows location-based services to also connect people in temporal proximity, in addition to those who are co-located.
- We present the design of a smartphone-based system called ENACT (Encounter-based Architecture for Contact Tracing) to detect close encounters, while providing strong privacy properties.
- We describe protocols for the ENACT architecture that send alerts to the users who were in spatial and temporal proximity of the infected user. We further extend the protocol to reduce the risk of disclosure of user identity or location history.
- We consider the challenges of a real-world deployment of the ENACT protocol and suggest ways to handle the challenges.

2. THE ENACT SYSTEM

Even though all users receive the alert, the system must, with high accuracy, identify those users (recipients) who were near the infected surface during the time the virus was active, and alert them about their exposure to the virus. We begin with a simple design that addresses personalization accuracy, but later, we extend the design to also provide user privacy and alert authenticity.

2.1 System model

A simple approach to identify the users who were in a close encounter with the patient could be to compare the location traces of the users with those of the patient. Instead of storing users' actual location traces, ENACT relies on "event tags" – the tag encapsulates both the spatial and temporal location of the user, i.e., where the user was located and when.

The ENACT system comprises five main components: users, smartphones, transmitters such as Wi-Fi access points, health providers, and the ENACT server. We make the following assumptions about each type of component.

Users. Users are people who have installed the ENACT mobile application on their smartphones and carry their smartphones at nearly all times; a study conducted in 2013 with 7,446 participants showed that 79% of smartphone users have their phone on or near them for all but two hours of their waking day and 63% keep it with them for all but one hour [12].

Smartphones. Smartphones have wireless capabilities to receive packets from nearby transmitters and connect to the Internet. On receiving packets, the application stores a tag that contains both the identity of the transmitter and the time when transmitter was observed by the smartphone. The application also stores, in the tag, the RSSI value, which is a number representing the strength of the transmitter signal.

Health Providers. Only an authorized health provider can diagnose a patient to be sick and contagious, collect tags from a patient's smartphone, and send alerts about the disease; we discuss later how the ENACT can verify authenticity of alerts.

Wi-Fi access points (AP). In the simple protocol design, the tags contain the MAC addresses of APs that the user's smartphone encounters (instead of the actual locations). If two users are at the same location, their smartphones will observe at least a threshold number of common APs; this fact is the basis of the tag matching for detecting close encounters.

ENACT Server. The server receives alerts and forwards them to all users.

2.2 Event tags

The smartphone application stores event tags, which contain the identity of the Wi-Fi access points (APs) it observes as well as the time duration during which they were observed. How can we use this information to determine if two users shared a close encounter?

Suppose the users shared an encounter. One way of determining if two devices are spatially close is by finding their individual locations and then computing the distance between them [14].

Proximity-based localization techniques rely on the assumption that a smartphone is approximately co-located with the wireless device from which it receives the strongest signal. Prior work has relied on the correlation of the wireless (RF) signals obtained by two devices in close proximity to each other to determine whether they were spatially co-located [14, 15, 20]. ENACT relies on a similar hypothesis that co-located devices observe a threshold number of common APs with similar RSSI values, ignoring the vagaries of wireless networks. People's durations of stay may vary, but they will have both observed the common access points during the time when they both were at the same location, with similar RSSI values [14, 15]. At least one pair of event tags will match as they observed at least one common AP with similar RSSI values, and within the same minute, since they shared an encounter.

Suppose the users shared a close encounter. Since the patient and recipient were at the same location, they would have observed at least a threshold number of common APs with similar RSSI values. But they observe the APs at different times, and these times are also recorded in the tags. The event tags will match only if the common APs were observed within the temporal boundary of the close encounter. If a close encounter was defined by "all users who visited a location within two hours", then two users' tags match only if they observed a threshold number of common APs with similar signal strengths within a span of two hours.

For example, at time T, A observes APs a, b and c, and at time $T + 60$, i.e., an hour later, B observes a, b and d. A stores the tags (T, a, R_a), (T, b, R_b), and (T, c, R_c) and B stores (T', a, R'_a), (T', b, R'_b), and (T', d, R'_d), where $T' = T + 60$ and R_a and R'_a indicate a's signal strength as observed by A and B respectively. If a close encounter is defined by a temporal boundary of two hours, $R_a \sim R'_a$ and the threshold number of common APs observed at that location is at least two, A and B will have one set of matching tags, stored at times T and T'. If B continues to observe a and b with similar signal strengths until $T' + n$, then there will be n matching tags, indicating that B was exposed to the virus for n minutes.

The ENACT system relies on the smartphones to store the tags until a disease outbreak occurs. Since the incubation period for most contagious diseases is less than a month, it may be sufficient to store tags for a month. However, a user may choose to delete tags at any time, at the cost of losing the opportunity to obtain personalized alerts about virus exposures.

2.3 Communication steps

Now, we present the protocol and explain how the different components of the ENACT system interact with each other.

1. All smartphones store the MAC addresses of the APs they observe, along with the RSSI and the time when the smartphone observes the APs.

2. When a user is diagnosed by the health provider, the health provider generates an alert using the ENACT application using disease and patient-specific parameters and sends it to the patient, via the ENACT app. The patient-specific parameters include how long the patient has been contagious and the time a virus survives on a surface.

3. On obtaining the patient-specific parameters, the patient's smartphone retrieves all the tags it stored during the time duration when the patient was contagious, attaches it to the alert and sends it to the server.

4. The server forwards the message to all users. On obtaining the alert, the recipient's application retrieves the relevant tags stored on the user's smartphone, during the time when the patient was contagious. The application tries to match the tags it retrieved with the tags included in the alert message. If there are n matches, the application generates and provides the recipient a personalized alert that the user may have been exposed to the virus for n minutes. If there are no matches, the application presents the user a general alert from the health provider.

3. EXTENDING ENACT

Since tags contain the MAC addresses of APs, on obtaining the tags, a recipient application can determine which APs a patient saw and when. If a recipient with malicious intent gains access to this information, they may be able to determine the identity of the patient, or reconstruct the location history of the patient. Other ways malicious users could misuse the system is to create panic on campus by sending fake alerts. We extend our simple design to provide better privacy to users, and ensure authenticity of alerts.

3.1 User Privacy

To prevent tampering and unnecessary exposure of location history, the system needs a different location representation that varies across time and space. This representation should not statically refer to the location, or to a proxy for location like the MAC address of an AP, so any compromise of the location history will not disclose actual locations. However, this representation needs to be generated by the AP itself since both the patient and recipient should be able to obtain it. Furthermore, the representation, which we call an 'event tag', should vary over time so no adversary can "map" the entire campus and be able to determine users' past locations. Instead, these tags should be designed solely for determining close encounters. The patient's smartphone must be able to generate event tags for the duration when the virus is active, for each location they were at when contagious, because the patient may not stay in or near one location for the entire period when she was contagious and may receive only one or a few tags from the same access point when she was at that location.

Table 1: Notations used in the ENACT system model.

Notation	Description
p	Patient
r	Recipient
u	Other users
HP	Health provider
B	Wireless transmitter
n	Number of wireless transmitters at any location
b	Broadcasting interval
t	Hash chain value
T	Timestamp
C	Contagious period
τ	Virus survival period
R	RSSI value
$g(t, T, R)$	Set of generated tags based on tag t for location patient was at time T when receiving signal of RSSI R
$h(X)$	Message digest (hash) of X
f	Position in hash-chain
$d(T)$	Database record for time T
$sig(D, X)$	D's signature on text X
$E(k, X)$	Encrypt X using key k

In the ENACT system, an access point uses hash chaining to generate and broadcast a series of related numbers; it broadcasts a new number every b minutes. For simplicity, assume for now that the hash chain grows infinitely in that it never gets reset. In later sections, we discuss how the ENACT system handles hash-chain reset.

3.2 Alert authenticity

Next, to prevent fake alerts, the system must allow only the smartphone of a patient who is sick and contagious, as diagnosed by an authorized health provider, to send alerts about the disease. We achieve this goal by adding one more component to the system, a PKI server to issue certificates for health providers. When the health providers install the ENACT application on their smartphones, they generate private-public key pairs and register with the PKI server to obtain a certificate for their public key.

When a user is diagnosed by the health provider, the health provider generates an alert, signs the message and sends it to the patient, via the ENACT app. After attaching the tags, the patient's smartphone sends the new message back to the health provider. The health provider signs the final message that includes the signed alert and list of tags from the patient, and sends the signed message to the ENACT server. The server forwards the signed alert to all the users. On receiving the alert, the user's smartphone verifies the signature on the alert. If the signature is not valid, the message is discarded. Otherwise the application attempts to find and match tags in the user's smartphone with the tags in the alert it received.

The updated protocol with the hash-chaining technique and digital signatures works as follows; the protocol steps are shown in Figure 1, using notation from Table 1.

1. Beacon broadcast: The wireless transmitter, e.g., a Wi-Fi access point, sends a number representing location that we refer to as tag t. The smartphone obtains it with signal strength R at time T, and adds to its database a tuple (t, T, R); for simplicity, we exclude R in the diagram and subsequent descriptions. If it receives multiple tags from different access points during the minute T, it stores them as separate tuples,

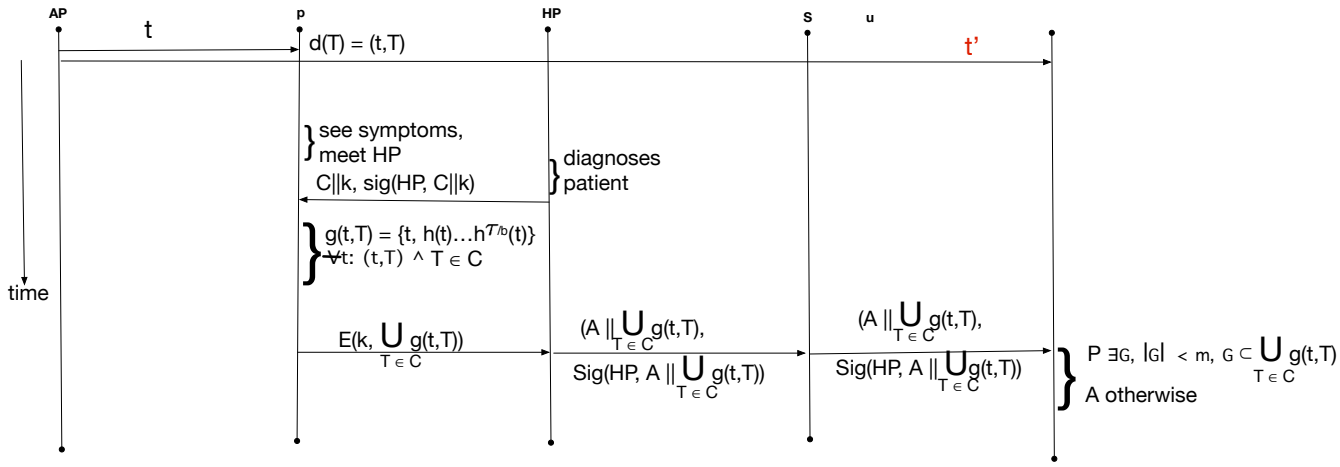

Figure 1: ENACT updated protocol

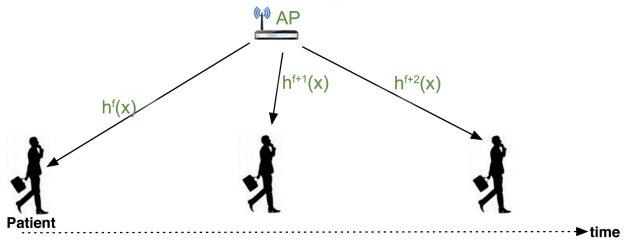

Figure 2: fb minutes after the transmitter sends a, user receives $h^f(a)$ from B. User receives $h^{f+1}(a)$ after another b minutes and continues to receive numbers from B when still within B's range.

i.e., $(t_0, T), (t_1, T) \ldots (t_n, T)$.

Suppose t is the f^{th} number in the hash-chain whose first number is a, i.e., $h^0(a)$=a. The f^{th} number is sent kb minutes after a. The access point first transmits the base tag a and at the end of the broadcasting interval of b minutes, it sends $h^1(a)$, where $h^1(a)$=$h(a)$. After another b minutes, it sends $h^2(a)$=$h(h^1(a))$, and so on. The f^{th} number is $h^f(a)$=$h(h^{f-1}(a))$. Figure 2 shows a user receiving three different numbers when within range of one transmitter. Assume for simplicity that all transmitters B_i started transmitting the base number a_i at the same time, and they all transmit the f^{th} tag, $t_i = h^f(a_i)$, in the hash chain at the same time, at time T, when the user observed the tags.

2. Sending alert: When a patient p observes symptoms, she visits the health provider HP and may be diagnosed with a certain disease. The health provider generates a message that contains information pertaining to the patient and her contagious state, including the time period during which she was contagious, C. The health provider's message also contains a cryptographic key k, and her signature to ensure the message's authenticity.

On receiving the message, the patient's smartphone retrieves all the tags it obtained during the time period C, so for all $T \in C$, it finds all the tuples of the form (t, T) and retrieves the tags t. For every tag it obtains, the smartphone computes a set of tags that may have been received by people who were at the different locations the patient visited during the virus survival time τ at those locations. This set $g(t, T)$ depends on the tags it obtained at time T and differs for each tag it obtained from different access points; so if the smartphone obtained n tags of the form $t_1, t_2, \ldots t_n$ at time T, it would generate n sets of $g(t, T)$.

Suppose the patient's smartphone obtains $t_i = h^f(a_i)$ from B_i at time T. The smartphone generates the set of tags $g(t_i, T)$ by generating $\frac{\tau}{b}$ numbers forward in the hash chain, starting from the most recent number t_i; $\frac{\tau}{b}$ denotes the number of tags the access point will send during the virus survival period of τ minutes. Then $g(t_i, T) = \{h^j(t_i) | 0 \leq j < \frac{\tau}{b}\}$. The application eliminates duplicate tags among the generated sets; duplication may happen in tag generation by the application if the user stayed at the location for longer than one broadcasting interval and received multiple tags from the same access point.

Consider an example scenario to better understand the tag-generation process: suppose the patient is at a location for an extended period of two hours, the broadcasting interval is one minute, and the contagious period is three hours; it uses the tag it received when it reached the location as the base number to generate numbers for the remaining duration of the hash-chain for up to three hours. So the application on the patient's phone adds to the alert message the 120 tags it received during the two hours, and generates the rest of the 60 tags by applying the hash function repeatedly, starting with the last tag it received when at the location. Of course, a patient may be contagious for days, and will be at different locations for varying durations at different times during the contagious period. For each tag obtained during the contagious period, the application creates a hash chain of 180 numbers, and removes any duplicates.

The smartphone collects all the tags in a message and encrypts the set with the key sent by the health provider, and sends the encrypted message back to the health provider. The health provider decrypts the message, generates an alert message A and attaches the tags to it. The health provider then signs the alert message and sends it to the server. The server forwards it to all users.

3. **Viewing alert:** On receiving the alert message, the smartphone first verifies the signature of the health provider. If the signature is invalid, it discards the message. Otherwise, it retrieves all the tags and attempts to match a threshold number of tags for one pair of time values within the temporal boundary of the close encounter.

   Considering only one AP for simplicity, tag matching is based on the following concept: Suppose a user arrived near the access point at a time later than when the patient first observed the same access point. The user receives $h^y(t_i)$ from the same access point. If $y < \frac{\tau}{b}$, the application will be able to find a pair of values generated from the same hash chain, and hence one matching tag. As shown in Figure 1, if there is at least one such matching tag, the application presents a personalized alert P that informs the user that they were exposed to the virus for at least one minute; i.e., the time of exposure may be computed based on the number of matching tags. If no such match is found, the application shows the user the generic alert generated by the health provider, A.

By using the extended protocol, the patient achieves better privacy since their location history is encapsulated in a list of hashes whereas in the basic protocol the MAC addresses of all APs observed by the patient are included in the tags stored on their phone and sent to all users. Our basic protocol can work without modification to any existing network infrastructure since the tags contain MAC addreses; the only caveat to the basic protocol is the exposure of the patient's location history in the form of list of MAC addresses and time when the MAC addresses were observed. (The non-patient's location history data is never revealed.)

An adversary could potentially set up fake APs to send hash values, set up listening devices to obtain hash values from nearby APs, or install malware on all phones on campus to obtain hash values from all APs. However, we do not expect an adversary to expend the cost and effort required for the above attacks just to obtain the information collected by our system. If such attacks were to occur, more sensitive personal information about the individuals will be at stake, and hence, we consider these attacks and possible solutions out of scope.

4. IMPLEMENTATION CONSIDERATIONS

Health provider-patient interaction: The health provider can send the message containing the key and disease parameters to a patient by any near-field communication technique or a visual pairing method such as a QR code, since the patient and health provider will be co-located at the time of diagnosis.

Clock sync: On receiving and verifying the authenticity of an alert, a recipient's smartphone retrieves the tuples based on the time fields, and compares the tags: MAC addresses for the basic protocol and hash values for the extended protocol. By default, we consider the time granularity as one minute; we assume that the clocks of all the smartphones are synced to the accuracy of one minute, which is easily achievable with Internet time protocols. Typically, the smartphone would have more than one MAC address or hash value associated with a time value, in which case the smartphone would record them as multiple tuples. After a match occurs for a certain time value, the smartphone then looks for the number of location tags that matched, and checks whether the number is higher than the threshold.

Hash-chain reset: In the protocol, until now, we assumed that the hash chains are never reset. If the hash chains are never reset, a malicious user can obtain one hash value from an AP and easily generate all the future values to be sent by the AP. However, if the APs occasionally reset their hash chains by picking a new random number a as the seed for a new hash chain, how can we support patients and recipients when the virus survival period near the AP spans a reset moment? A recipient who collected hash values from an AP after the reset moment will not be able to learn about virus exposure from a patient who left the location before the reset occurred, as the patient will only be able to generate hash values in the previous hash chain before the reset and will not have any hash values in common with the recipient. One option is to reset hash chains daily at a time when we expect few close encounters to occur, for example, at 4am.

5. RELATED WORK

Several smartphone applications have been developed to assist contact tracers and responders for monitoring Ebola outbreak through location traces [24], screening for tuberculosis and malaria in Botswana [18], and using text messages [22]. Researchers have developed models to predict the spread of diseases using diagnosis [21] and social network data [26]. ENACT can be used to alert actual contacts, including strangers, based on real encounters and close encounters, without any involvement of contact tracers.

Researchers have used smartphones to help connect strangers who shared an encounter [16, 19], however, ENACT protects the identities of the patient and recipient from each other and the server. Finally, our work is complementary to techniques used for proximity testing [17, 23, 25].

6. SUMMARY

We introduced a new problem: detecting close encounters. We designed the ENACT system that allows people infected with a contagious virus to send alerts to other users who may have been exposed to the same virus due to a close encounter. We presented two designs – a simpler version that focused on personalization accuracy and an extended version for ensuring authenticity of alerts and providing better privacy to the users. Finally, we explored different considerations when implementing the system.

Acknowledgements

This research results from a research program at the Institute for Security, Technology, and Society at Dartmouth College, supported by the National Science Foundation under award number CNS-1329686. We thank Xiaohui Liang, Timothy Lahey, Ann Bracken, Virginia Brack, our colleagues in the THaW project and the anonymous reviewers for their valuable feedback.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

7. REFERENCES

[1] 9 things everybody should know about measles. Online at http://www.vox.com/2015/1/26/7907707/measles-symptoms-vaccine Last accessed April 15, 2017.

[2] Communication to Students Regarding Meningococcal Disease and New Vaccine Recommendation. Online at https://studenthealth.sa.ucsb.edu/docs/default-source/default-document-library/menb-dlist-email-8-11-15.pdf?sfvrsn=0 Last accessed April 15, 2017.

[3] Conjunctivitis/pink eye on Purdue campus. Online at http://www.purdueexponent.org/campus/article_23882082-0e65-11e7-8e2b-f7926bf6c174.html Last accessed April 15, 2017.

[4] Disease prevention on college campuses. Online at http://dujs.dartmouth.edu/wp-content/uploads/2011/03/28_pdfsam_11w_final.pdf Last accessed April 15, 2017.

[5] Meningitis confirmed on campus. Online at http://yaledailynews.com/blog/2016/02/18/meningitis-confirmed-on-campus/ Last accessed April 15, 2017.

[6] Mumps outbreak at Harvard threatens graduation. Online at http://www.nbcnews.com/feature/college-game-plan/mumps-outbreak-harvard-threatens-graduation-n564406 Last accessed April 15, 2017.

[7] Norostat data. Online at https://www.cdc.gov/norovirus/reporting/norostat/data.html Last accessed April 15, 2017.

[8] Norovirus tally reaches 700 kids; new cases declining. Online at http://www.vcstar.com/story/news/local/2017/03/30/norovirus-tally-reaches-700-new-cases-declining/99606526/ Last accessed April 15, 2017.

[9] Responding to measles outbreak. Online at https://www.insidehighered.com/news/2015/02/02/measles-outbreak-raises-issues-colleges Last accessed April 15, 2017.

[10] TB case spurs panic. Online at http://www.nydailynews.com/news/world/tb-case-spurs-panic-article-1.254526 Last accessed April 15, 2017.

[11] Update on meningococcal disease at Providence College. Online at https://www.brown.edu/about/administration/vp-campus-life/email-021115 Last accessed April 15, 2017.

[12] 79% of people 18-44 have their smartphones with them 22 hours a day [study]. Online at http://www.adweek.com/socialtimes/smartphones/480485 Last accessed April 15, 2017.

[13] P. Aditya, V. Erdélyi, M. Lentz, E. Shi, B. Bhattacharjee, and P. Druschel. Encore: Private, context-based communication for mobile social apps. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 135–148. ACM, 2014. DOI 10.1145/2594368.2594374.

[14] I. Bilogrevic, K. Huguenin, M. Jadliwala, F. Lopez, J.-P. Hubaux, P. Ginzboorg, and V. Niemi. Inferring social ties in pervasive networks: an on-campus comparative study. In *Proceedings of the ACM Conference on Pervasive and Ubiquitous Computing*, pages 123–126. ACM, 2013.

[15] I. Carreras, A. Matic, P. Saar, and V. Osmani. Comm2sense: Detecting proximity through smartphones. In *2012 IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 253–258, March 2012. DOI 10.1109/PerComW.2012.6197489.

[16] L. P. Cox, A. Dalton, and V. Marupadi. SmokeScreen: Flexible privacy controls for presence-sharing. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*. ACM Press, June 2007. DOI 10.1145/1247660.1247688.

[17] L. Ertaul, A. Balluru, and A. Perumalsamy. Private proximity testing for location based services. In *Proceedings of the International Conference on Security and Management (SAM)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2013.

[18] R. Littman-Quinn, A. Chandra, A. Schwartz, F. M. Fadlelmola, S. Ghose, A. A. Luberti, A. Tatarsky, S. Chihanga, D. Ramogola-Masire, A. Steenhoff, and C. Kovarik. mHealth applications for telemedicine and public health intervention in Botswana. In *IST-Africa Conference Proceedings, 2011*, pages 1–11, May 2011.

[19] J. Manweiler, R. Scudellari, and L. P. Cox. SMILE: Encounter-based Trust for Mobile Social Services. In *Proceedings of the Conference on Computer and Communications Security (CCS)*, pages 246–255. ACM, 2009. DOI 10.1145/1653662.1653692.

[20] S. Mathur, R. Miller, A. Varshavsky, W. Trappe, and N. Mandayam. Proximate: Proximity-based secure pairing using ambient wireless signals. In *Proceedings of the Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 211–224. ACM, 2011. DOI 10.1145/1999995.2000016.

[21] C. Milling, C. Caramanis, S. Mannor, and S. Shakkottai. Detecting epidemics using highly noisy data. In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 177–186. ACM, 2013. DOI 10.1145/2491288.2491294.

[22] G. Mosweunyane, T. Seipone, T. Z. Nkgau, and O. J. Makhura. Design of a USSD System for TB Contact Tracing. In *Proceedings of the International Conference Health Informatics (AfricaHI)*, pages 1–3, 2014.

[23] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh. Location privacy via private proximity testing. In *Network and Distributed System Security Symposium (NDSS)*, 2011.

[24] J. A. Sacks, E. Zehe, C. Redick, A. Bah, K. Cowger, M. Camara, A. Diallo, A. N. I. Gigo, R. S. Dhillon, and A. Liu. Introduction of Mobile Health Tools to Support Ebola Surveillance and Contact Tracing in Guinea. *Global Health: Science and Practice*, 3(4):646–659, 2015.

[25] G. Saldamli, R. Chow, H. Jin, and B. Knijnenburg. Private proximity testing with an untrusted server. In *Proceedings of the Conference on Security and Privacy in Wireless and Mobile Networks*, pages 113–118. ACM, 2013. DOI 10.1145/2462096.2462115.

[26] D. Welch, S. Bansal, and D. R. Hunter. Statistical inference to advance network models in epidemiology. *Epidemics*, 3(1):38–45, 2011.