Co-ordinating Developers and High-Risk Users of Privacy-Enhanced Secure Messaging Protocols

Harry Halpin, Ksenia Ermoshina, Francesca Musiani

Co-ordinating Developers and High-Risk Users of Privacy-Enhanced Secure Messaging Protocols

Harry Halpin[1], Ksenia Ermoshina[2], and Francesca Musiani[2]

[1] Inria
2 Rue Simone Iff
harry.halpin@inria.fr
[2] Institute for Communication Sciences, CNRS
20 rue Berbier-du-Mets 75013 Paris, France

Abstract. Due to the increased deployment of secure messaging protocols, differences between what developers "believe" are the needs of their users and their actual needs can have real consequences. Based on 90 interviews with both high and low-risk users, as well as the developers of popular secure messaging applications, we mapped the design choices of the protocols made by developers to the relevance of these features to threat models of both high-risk and low-risk users. Client device seizures are considered more dangerous than compromised servers by high-risk users. Key verification was important to high-risk users, but they often did not engage in cryptographic key verification, instead using other "out of band" means for key verification. High-risk users, unlike low-risk users, needed pseudonyms and were heavily concerned over metadata collection. Developers tended to value open standards, open-source, and decentralization, but high-risk users found these aspects less urgent given their more pressing concerns.

1 Introduction

The last few years have seen the spread of wide variety of secure messaging applications, and increased interest in these applications due to generalized concerns around privacy and security, caused in part due to the 2013 Snowden revelations. As put by previous work [2], currently developers imagine what properties users likely need, and these properties may or may not actually satisfy the needs of end-users. The first large-scale user study of secure messaging has shown that users are fundamentally confused around secure messaging in particular and how encryption works in general [1]. However, one trenchant criticism of this study is that it does not deal with *high-risk* users who are motivated to learn about encryption. High-risk users may care about very different properties than low-risk users in terms of their threat models. For example, most secure messaging properties do not hide the user identity, with applications like Signal exposing valuable information via associating users with their phone number despite concerns from high-risk users. If developers themselves are relatively low-risk users and building tools aimed at high-risk users, then the tools may or may not match

the needs of these high-risk users. Furthermore, standards may be biased against high-risk users, as many high-risk users cannot easily participate in standardization discussions, and their interests tend to be represented by developers who may or may not have an accurate understanding of their threat model. Although the vast majority of applications are converging to the "de-facto" Signal Protocol, developers still are still in a state of flux over what the security and privacy properties of secure messaging *should* be due to the lack of a single widespread standard (in contrast to the TLS link layer protocol for client-server encryption) [8]. Also, the newly started IETF Messaging Layer Security (MLS) effort provides an opportunity to finally produce a coherent security standard for secure messaging.[3] We hypothesize that there can be a disconnect when existing applications are developed primarily with the threat models of the developers themselves in mind rather than those of high-risk users, which is understandable given how difficult it is for developers to contact high-risk users. Also, there are also many properties outside of security and privacy that are important to users, such as decentralization, standardization, and licensing. In this study, the properties we investigate are classified into six broad categories: (1) Security Properties, (2) Group Support, (3) Privacy Properties, (4) Decentralization, (5) Standardization, and (6) Licensing.

Although many usable security studies call for end-users to be helped by developers [9], it should not be forgotten that developers can be helped in building new tools by input from user studies [3]. If the lesson from usable security research on secure messaging is that users are fundamentally confused and do not even believe that encryption can defend the confidentiality of their messages, the entire endeavor of creating secure messaging applications may seem futile [1]. Yet one problem with the results of this previous work and other usable security studies is their sample: Low-risk users will likely not have a clear threat model and invest time into understanding how their threat model maps to the properties of applications. Our research contribution is that we demonstrate high-risk users actually do care about the properties of secure messaging applications. By detailing what properties concern these users, we provide valuable feedback to the developer community. This work is the first to do such a study that balances high-risk users, low-risk users, and developers themselves. Our previous qualitative study focused on less than 50 users [2]. The results presented here extend and deepen those results, and they directly contrast and even contradict the results given in the largest quantitative study of 80 low-risk users [1]. We first state our theses, generated after our first study [2] but before the additional interviews given in this study, in Section 2. Our qualitative methodology is explained in Section 3, with the interviews being delved into in Section 4. Finally, we explore the results and future work in Section 5.

[3] https://datatracker.ietf.org/wg/mls/about/

2 Problem Statement

Our first thesis is the ***Developer-User Disconnect:*** We hypothesize that *the properties desired by developers are not necessarily desired, or even understood, by users.* The core of the problem is the methodology currently used in the developer community to design protocols, where developers of secure messaging applications hypothesize what properties a protocol should have based on their beliefs about users. These properties may or may not line up with the expectations of users, and therefore the goal of our project is to determine the properties developers believe are important and see if these properties match the properties wanted by users. Though some attempts are made to gather and analyze user experience via online feedback forms and rare offline workshops (observed at international events such as CCC, IFF or RightsCon), contact between high-risk users and developers seems minimal. Such feedback can be produced by tech-savvy high-risk users willing to contribute in co-developing free and open-source projects, although high-risk users are of course often engaged in more pressing issues at hand than dealing with bug reports. Users and developers need to converge in order to harmonize the needs of users with the concrete design decisions made by protocol designers.

Our second thesis is the ***High-Risk User Problem:*** We hypothesize that *high-risk users have different needs and behavior than low-risk users.* Although seemingly obvious, in most studies of end-to-end encrypted messaging, the features and problems users encounter may not be representative of actual end-users as the user base that is often studied in usability experiments in the United States and Western Europe are usually a rather homogeneous selection of students from a low-risk background [1]. Although it can be claimed that all users are to some extent "high-risk" potentially, we would argue that it is sensible to divide users between those *high-risk users* who can in the short-term suffer concrete physical harms such as long-term imprisonment and torture as a result of information security, and those *low-risk users* who do not have immediate consequences due to failures in information security. As put by one user, "We did not have any concerns about security, but then people started disappearing" (H13). High-risk users tend to be located in geopolitical areas where information security is important due to political instability, although well-known activists and persecuted minorities in "stable" countries would count as high-risk users. We hypothesize that high-risk users have different threat models and so different requirements for privacy and secure messaging. Note that this thesis dovetails with the *Developer-User Disconnect Problem*: as most developers are not high-risk users themselves, they imagine the threat model of high-risk users as well as the feature set they may desire and what trade-offs are reasonable, but these projections of the needs of high-risk users could easily be inaccurate.

3 Methodology

Our study combines quantitative methods from usability studies with a qualitative approach based on Science and Technology Studies (STS), trying to follow

how developers create a technical artifact based on their motivations and how those – with varying degree of success – attempt to map to the needs of real users [5]. Ethnographic work consists of interviews, typically done in the organic environment of those interviewed, which test particular scientific hypotheses and assumptions. These questions should attempt to elucidate a phenomenon so that a model with some explanatory power can be built of the mental model of those interviewed. Interviews should ideally be cross-cultural and large enough so that the results generalize to some extent.

Note that due to the vast differences in context between developers and users, achieving "representativeness" in the present study is extremely problematic and therefore not a goal. In this regard, while previous studies looked at either a selection of primarily students in London [1] or (our own previous) work on a small subset of "radical" developers (Briar) in contrast to high-risk users in Russia [2], we have expanded the user studies to deal with interviews with corporate developers at corporations as well as high-risk users in places such as Syria (in detail, the Kurdish area of Rojava). This leads to a much more diverse spread of what we term "risk" than previous works. Thus, we continue to distinguish users as either *high-risk* or *low-risk*, based on whether or not the failure to encrypt their messages or other metadata leakage by a secure messaging tools could realistically endanger the user or the person they were communicating with (for example, a situation where the misuse of secure messaging would likely lead to death due to conditions of war or high prison sentences due to participation in a protest). In this regard, we would take a user "at their word" in their subjective reporting if they described the repercussions of their failing to send messages securely.

3.1 Interview Selection Process

Interview subjects were either developers or users. Developers were selected due to pre-existing social relationships with the larger academic community. Although this does create bias, this can be normalized to some extent by doing a large number of interviews as well as taking into account the relatively small size of the global developer community of secure messaging applications. For developers of applications where there was no social relationship, we reached these developers via their GitHub pages. Some developers were selected due to their attendance at conferences such as Real World Crypto in Zurich in January 2018. Interviewed users were selected individuals based on either their attendance at training events in their local environments (both high-risk, in the case of Tunisia and Ukraine, and low-risk in the case of the United States and France) or at conference venues that were likely to attract high-risk users living in regions that, due to the level of repression, made it difficult if not impossible to interview them in their native environment, or would make it such that they could not speak openly in their native environment due to repression. This was the case for users from Egypt, Turkey, Kenya, Iran, where the interviews took place in March 2017 at the Internet Freedom Festival and at RightsCon 2017, as well as in person if possible. All interviews were made between Fall 2016 and Spring 2018, for a

total of 90 interviews. We interviewed (30) developers, including developers from LEAP, Pixelated, and Mailpile (PGP), ChatSecure and Ricochet (OTR), Signal, Wire and Conversations (OMEMO) as well as developers from Briar that use their own custom protocol and developers from Microsoft, Facebook, and Google working on security.

As noted earlier, we distinguish between high-risk users (30) and users (including researchers and students) from low-risk countries (30). Note that low-risk users were not randomly selected and so were far from ordinary users, and so were biased towards possessing high knowledge of secure messaging they attended training events or worked in the corporate information security sector. All low-risk users were from the USA/Western Europe. High-risk users included users from Ukraine, Russia, Syria, Lebanon, Egypt, and Iran but also users from countries in Europe (4) that are or were under considerable legal pressure and surveillance due to ongoing court cases. Some high-risk users, due to the conditions in their country, had left (4) or maintained dual residency (2) between their high-risk environment and a low-risk environment. Again, we do not claim that these high-risk users are a "representative" sample, but that the sample should be large enough to demonstrate considerable differences between at-risk users and those that do not think of themselves as particularly at risk. Some of those interviewed (18%) considered themselves "trainers," i.e. people that train others in using secure messaging tools.

3.2 Interview Procedure

Our interviews were conduct using a protocol designed approved by an external Ethics Advisor to be compliant with the European General Data Protection Framework. The forms and full set of precise questions are available online.[4] Given the risk of many users, we took whatever steps the interviewees deemed necessary to protect their privacy. High-risk users did read and consent, but copies of the form were not kept if they wished to remain anonymized without a record. Interviews were done in person or online. If the interview was done online, we allowed the users to use a tool of communication of their choice, and so some interviews were done via secure messaging tools such as PGP and Signal as well as insecure means such as Skype and e-mail. If the interview was done in person, the interview was recorded with an audio recorder with no Internet connection if the interviewee approved, otherwise written notes were taken by hand. Interviews were typically done in English, although they were done in the native language of the speaker if the interviewer could speak the language fluently. We use a dedicated encrypted hard-drive to store the interviews. Before the interview we asked our respondents to carefully read the Information Sheet and the related Informed Consent form and ask any questions regarding the privacy of their interview, their rights over their data, and our methodology. The name of the interviewee was not mentioned during the recording and withdrew any context (such as the country or the city, the precise social movement a user

[4] http://www.ibiblio.org/hhalpin/homepage/forms.zip

was involved in, and so on) if the interviewee asked for this. Interviewees did not have to answer any specific question that they felt uncomfortable answering.

We interviewed 90 people in total, composed of 30 low-risk users as in France, Belgium, the United Kingdom, and the United States; 30 high-risk activists and journalists in Syria, Tunisia, Iran, Egypt, Russia, and Ukraine; and finally 30 developers of applications such as Wire, Briar, Signal, ChatSecure, OTR, and developers at large companies including Google, Microsoft, Facebook, and Paypal. The age of those interviewed varied widely between late teens to over fifty, with the majority of users being in their mid-twenties to early thirties. Developers considered developing secure messaging applications as their main job. Low-risk users and high-risk users had a wide variety of jobs, but were all explicitly interested in secure messaging. The researcher who did the interviews attempted to summarize each of them using a manual coding scheme. The questions used allowed large amounts of free form discussion, and coding often ignores much of these nuances; thus, in the results, actual quotes are often used. When referring to high-risk users, we use the letter "H" and a number, for low-risk users we use the letter "L" and a number, and for developers we use the letter "D" and a number.

4 Interviews

The results of the interviews are presented in this section. For each category of questions, representative quotes have been chosen. The results of the interviews are summarized on a high-level in Table 1. Note that the categories 'high' and 'low' were classified with a strict equal or greater than 50% threshold in terms of importance. If a user did not mention or did not know about a certain property, it was considered to be not important (except in the case of "main threat"). Note that our statistics should not be taken without some variance, and are meant to be used as a general summary. In particular, we do not claim that interview sample size is necessarily representative of a larger homogeneous population, likely due to geographical and cultural variance (for example, there were no interviews with Chinese high-risk users). Nonetheless, the regularities that appear in our interviews should be useful to developers of privacy-enhancing technologies and security standards. The developers of secure messaging applications often put themselves "in the shoes" of high-risk users but naturally have difficulty of arranging interviews with high-risk users themselves.

4.1 Developer Motivation

Developer motivation was quite wide-ranging, but largely could be divided between those who wanted to start privacy-enhanced businesses that would serve both low and high-risk users, to those who were primarily motivated by protecting high-risk users due to human rights concerns that are more traditionally dealt with by the NGO sector. In the case of Wire, the developers felt that they were addressing issues that they had neglected in the original design of Skype,

Interview	Developers	Low-risk Users	High-risk Users
Main Threat	server (60%)	server (73%)	client (60%)
Security (Key Verification)	high (53%)	low (20%)	high (53%)
Security (Ephemeral Messaging)	high (83%)	high (83%)	high (90%)
Privacy (Metadata Collection)	high (73%)	high (60%)	high (93%)
Privacy (Pseudonymity)	high (53%)	low (43%)	high (93%)
Group Support	high (86%)	high (96%)	high (96%)
Decentralization	high (86%)	high (70%)	low (43%)
Standardization	high (70%)	high (50%)	low (40%)
Open Licensing	high (70%)	high (60%)	low (46%)

Table 1. Importance of Properties in Secure Messaging Interviews

as "after Skype was sold to Microsoft [they] had an idea of how to build a new Skype...or what Skype should look like 15 years after. One of the biggest gaps that was missing on the market was related to privacy and security." Nonetheless, they had very limited contact with high-risk activists and stated that the application was developed "mainly for private usage" by individuals, leading to some technical limitations such as group chat only supporting up to 128 users that made it unusable for large-scale social movement organizational purposes. However, the goal of Wire was ultimately not to serve high-risk activists, despite its popularity with high-risk activists due to not requiring a phone number like Signal, but to provide secure business communications (D7).

On the other hand, although Briar has very little use from high-risk activists (no users, although two developer users), the entire concept was inspired by inquiries from high-risk activists asking "if LimeWire would be suitable for communication" and that although the developer of Briar (who worked at LimeWire at the time) felt "that it may be suitable ... we can build something suitable on a more social basis," which in turn led to the development of Briar (Michael Rogers, D6). Likewise, the developers of Signal aimed first at high-risk activists, and unlike, Briar have managed to capture a large cross-section of high-risk activists (88%). In fact, the primary group that did not use Signal was developers who were building competing projects and low-risk users who disagreed with its centralization.

From our survey, it appears that more developers in projects are motivated by serving the security and privacy needs of high-risk activists than forming a successful business, although Wire, Matrix, Apple, Microsoft, and Telegram are motivated by business use-cases. Yet strangely, developers from systems aimed at high-risk users (Signal, ChatSecure, Briar) have little actual contact with high-risk users in their systems and were, at least in the case of ChatSecure, surprised by how many high-risk users actually started using their software.

4.2 High-risk vs. Low-risk Users

Developers tended to distinguish between low-risk users who are "privacy-aware" and high-risk users such as human rights activists in war-zones. High-risk users, unlike low-risk users as interviewed in earlier studies [1], have a well-defined threat model typically focused on active attacks and client device seizures (60% were more concerned by the latter), but were not certain of the extent to which they were protected by secure messaging applications. However, low-risk users had an implicit threat-model with a focus on passive traffic monitoring and server seizure (73% were more concerned with the server being attacked or monitored than client device seizure, and the rest could not decide). Almost all users had tried PGP (93%) but its average use was weekly or rarely (with a substantial group of users giving up on its usage after attempting to use it), while all users except one[5] found themselves using secure messaging daily.

As has been observed among our interviews, in more high-risk geolocations, the choice of secure messaging application can be due to the politics of its country of origin. For example, in Ukraine, high-risk activists exclude applications and online services that have servers on the territory of Russian Federation or Ukraine and prefer American-based services, with even trainers advocating usages of Gmail and Facebook. Similar dynamics were observed in Iran (with no adoption of PGP and strong preference for Gmail with two-factor authentication), and Egypt, where WhatsApp is popular as the United States is considered as not being part of the threat model. For example, "Iranians use Google and Gmail a lot, they do not care about NSA spying on them, they care about Iranian government not having access to the data. We use Google Drive to upload our personal photos for example. For us the entire motto of 'Use Signal, Use Tor' does not make sense. As soon as the servers are inaccessible for the [Iranian] government, it can be used" (H14).

Indeed, high-risk users noted that their threat model changed over time, as most Tunisian high-risk users we interviewed became less frequent users of PGP and other secure messaging tools after their 2011 revolution, even though some of them were concerned that "surveillance powers were going to be used again soon against us" (H18). As opposed to key verification where their technical grasp of the cryptographic mechanics was often shaky, interviewed high-risk users had well-conceived and often thoughtful threat models based on geopolitics, but varied in a country-specific manner in terms of application preference. Unlike low-risk users who were primarily concerned about corporate or NSA surveillance, high-risk users were more concerned by surveillance from their own nation state, and could accurately assess the geopolitical origin of messaging systems. For example, apparently, Viber had been very popular in the Middle East, until it was discovered it was created by a company based in Israel, and then its usage "declined quickly" among high-risk users (H17).

[5] A developer that used only PGP and IRC.

4.3 Security Properties

The main advantage of OTR and Signal-style protocols is that key management is no longer a barrier to entry, and this appeals even to high-risk users. Trainers often found it too difficult to teach even high-risk users the precise security properties of key management, noting that "some trainers think there should be no key discovery at all, it is better to have opportunistic or automatic key discovery as it is happening with Signal. Different encrypted messaging apps have popped up that made it a lot easier to have just an app that will pass on your communication, and the encryption part will be transparent to the user. Having encryption as a default mode is the key part in making encryption popular" (L14). The vast majority of users preferred not having to do key management. The situation got worse as the risk of the situation became worse. Although the YPG in Syria still use PGP, H1 noted that "Every single person needed walking through the steps, when you have a bunch of soldiers who don't even know how to use a computer, they have to install an email client, generate entropy. People would right the wrong email address. People would encrypt the email for their [own] PGP key. They would send me their private key. It was an absolute nightmare" (H1). Yet many high-risk users wanted some form of key verification and out-of-band authentication, even if it was hard to use in practice and they did not use it regularly. Both high-risk and low-risk users insisted on the importance that they could "see encryption happening" in the interface.

Due to their concern with active attacks, high-risk users often try to verify keys (53%) after they receive a notification that the key has been changed in Signal, WhatsApp, Wire or other applications while only (20%) of low-risk users do. High-risk users are afraid that the devices of their friends have been physically accessed, stolen or otherwise taken away by powerful adversaries willing to use physical force and subterfuge to access contacts lists. Yet high-risk users tend to confound device seizure with key material being changed, and do not realize that if a device was seized an adversary could continue communicating using the seized key material. Some do realize this possibility but then try to ascertain the identity of their contacts using out-of-band channels: "If I get a message from Signal for example, saying that my contacts device has changed or his fingerprints changed ... I normally try to get in touch with the person ... I need to hear the voice" (H11). A subset of high-risk users do verify keys on first communication, and check the authenticity of a person if the key material changes informally using context: "I verify keys in PGP, but...I verify the person by other means... we speak about same things. In Jabber also I often just do it manually, without shared secret. But I always check if I receive something warnings about the persons device" (H4). Developers do tend to verify keys, and it seems to be related to habits gained in software development. As shown by previous work, key verification is hard to both understand and use [6].

The most noticeable difference between low-risk users and high-risk users was the targeted and device-centric nature of the threat model of most high-risk users. As device seizure was a primary concern, ephemeral messages were viewed as a required feature for almost all high-risk users (93%). It was also

viewed as important by most low-risk users (83%) due to privacy concerns, and some developers were against this feature (17%) as it was felt users may be "tricked" into thinking a message has been deleted from a server when it has not. Furthermore, while developers and low-risk users preferred key material to be stored on a local client device, high-risk users almost always preferred their key material to be stored in an (extra-jurisdictional) server and for them to have the ability to delete their existing device or reinstall quickly on a new device. As noted by H16, "When I cross the airport, I delete all my messages and I export my contacts, then I put them on my laptop, and I delete my contacts from my phone. I use plausible encryption to hide them [the contacts on the laptop]. I use Veracrypt." Plausible deniability became very important to high-risk users in case of device seizure, where it was important to be able to give police authorities access to a clean device or have a second encrypted volume on their device. The fear of Google and the NSA storing large amounts of data for later decryption or data analysis in the long-term is viewed as more of a problem by low-risk users, as this problem does not have immediate consequences for many high-risk users. Instead, their "real risks are house searches, seizing of devices" (H13).

4.4 Privacy Properties

Many developers found increasing privacy through minimizing metadata collection to be the second most important feature (73%) after end-to-end encryption: "End-to-end encryption is a first step. But besides there is a whole new dynamics that needs to happen that's related to all of the metadata and what is it used for" (D14). Yet many developers confused the possibility of a third-party adversary monitoring their communication and so collecting metadata with whether they as developers personally collected data, as exemplified by one developer that stated simply "I do not have anything in my code that would let me know how many people are watching the app right now" (D4). However, many developers also believed they would have to collect some metadata in order to interoperate with features such as push notifications of arriving messages, but they try to limit the harm: "With introducing the push messaging it's the first time we're exposed to possible metadata. But we don't log anything, we don't know who is talking to who, we don't log any information" (D7). Most developers who were aware of third-party data collection and surveillance in general were supportive of using Tor and on disabling the collection of phone numbers in particular, but lacked a comprehensive plan to minimize the collection of data.

All users wanted better privacy in terms of anonymity (defined in terms of unlinkability to their identity due to lack of metadata collection) in secure messaging, and high-risk users found it exceptionally urgent (93%) compared to low-risk users (60%). The same high-risk user (H1) that wanted to get rid of the attachment of identity to phone numbers noted that removing phone number identifiers "introduces a whole new set of problems such as spam and fake addresses." High-risk and low-risk users generally supported reducing data collection and increasing privacy, although often users assumed the encryption of data to hide metadata. A user that had been active in Syria noted that "Turkey

is definitely spying on communication, it's really bad I can't use Signal over Tor ... I was always using Tor on my phone, but then I had to switch it off as it would randomly stop working ... I wish I could buy a tablet, and then my device wouldn't be tracked, and I could make Signal calls with normal internet" over Tor (H16). Ukrainian users mentioned social networks such as Vkontakte and Facebook as the main source for de-anonymizing users and insisted on a need of changing privacy settings to allow their friend lists to be more privacy-enhanced and not reveal phone numbers or other personal data in the case of device seizure.

Developers and information security trainers underlined the urgency to find a reliable solution to protecting personal data such as phone numbers in secure messaging applications, as nothing in the field of end-to-end encrypted instant messaging apps offers excellent privacy properties: "Metadata connects you weirdly with other people, and there's more sense in the metadata than in the data itself for technological reasons [...] No one from the messaging apps is trying to solve that. Instead they suggest to sync your address books so they know exactly who you're talking to even though you trust them to somehow make it into hashes or whatever. That's the issue we are not solving with the apps, we make it worse. We now have centralized servers that become honeypots, and it's not about the data, it's about the metadata" as put by Peter Sunde of Heml.is (D9). Developers and trainers associated the leaking of metadata with centralization, although it should be noted that decentralization can just as easily and perhaps more easily leak metadata [7].

4.5 Pseudonyms

The most controversial of security properties is repudiation. Some developers (OTR, Signal, Wire, Briar) believed that a protocol with repudiation (plausible deniability) should be required in secure messaging applications so that a transcript of a chat could not be provably linked cryptographically to the key material of a user in a manner the user could not deny. Post-PGP protocols such as Signal and OTR are defined not to include authentication using traditional non-repudiable cryptographic signatures, but accomplish authentication via other means. In some versions of Signal and OTR, in order to obtain both non-repudiation and 'future secrecy,' Diffie-Hellman key exchanges are combined with key ratchets, but this leads to impersonation attacks due to lack of authentication [4]. Indeed, some form of verification of the user-level authentication was in general viewed as an important feature by high-risk users, as explored earlier in Section 4.2.

The social and cryptographic uses of plausible deniability should be separated, as many high-risk activists had clear use-cases for temporary identities and concerns over transcripts being seized. Repudiation is when it can not be proven that any given message *cryptographically* came from a particular user or from an unknown third-party. OTR even went as far as to use malleable encryption to enforce this property. Yet it is unclear if such a cryptographic technical construction would ever lead to plea of plausible deniability being accepted in

a legal court (including the infamous use of OTR by Chelsea Manning to communicate with Adrian Lamo, where the logs of the chat were used to convict Manning regardless of repudiation). No high-risk users were aware of any examples where cryptographic deniability was used in actual court cases, as usually the social context and other circumstantial evidence was enough to determine if a user was involved in a conversation.

However, there are many cases where high-risk users wanted plausible deniability on the application level via pseudonyms that could not be linked easily via social context. Their use cases were all related to creating a new identifier and so chat transcripts would be attached to a new and possibly temporary identifier with its own cryptographic keys. Most (93%) high-risk users wanted some ability to set up temporary pseudonymous accounts, and were disappointed with the lack of such accounts on secure messaging applications, which in turn forced them to use shared e-mail inboxes on servers like *riseup.net*, temporary accounts on corporate email servers using GPG, or temporary XMPP+OTR accounts. In contrast, low-risk users did not in general need multiple short-term pseudonymous identities (only 43%). One Russian high-risk user used so-called "one-time secrets" to share information via accessing specific websites via Tor Hidden Services as this offered a possibility to send and receive unique self-destroying messages (H9).

4.6 Group Support

Most (96%) users (regardless of their risk level) wanted support for group messaging, although a substantial minority of developers (14%) were against it due to the problems group messaging would cause for security. Developers of secure messaging apps perceive group support as one of the main challenges, and both high-risk and low-risk users we interviewed wanted some form of group chat. Group size varied dramatically from 2 to 600, but in general the groups wanted by both high-risk and low-risk users tended to have around 25 members. Developing group chats in peer-to-peer systems "is both an important usability feature and a scientific problem" (D11). Yet who has the right to invite and ban participants of a group chat? While Signal, Wire, and Telegram offer the possibility that any member can invite new participants, while Briar puts forward a design "where only the creator of the group has a right to invite and ban participants" as it leads to better "anonymity," although "it leads to a certain centralization within a distributed system" (D6). Group management ends up being important: Russian users had a group chat compromised when members were arrested and their phones seized, but there was no ability to remove the compromised accounts and many users did not realize that the police could be reading their secure group messages until it was too late. Overall, high-risk users and low-risk users were evenly split on whether or not a system should have an administrator deciding who was in the group, while low-risk users wanted archives (65%) and high-risk tended to be against archiving old messages of a group (78%), primarily due to legal concerns.

Another open research and practical challenge is hiding the social graph of participants of the group chat. Developers we interviewed proposed different solutions to this problem. In Briar, every new member of a group chat is only connected with the creator of the group, who is by default the administrator. In contrast, Wire opted for a solution to analyze contact networks of users and to suggest new contacts to users based on this analysis. This data-mining of contacts was criticized heavily by both trainers and the security community as revealing metadata, which would be dangerous to high-risk users, and led one high-risk user to say "I'd rather not use Wire due to that feature, as I'm sure there's at least one [undercover] agent in my contact list and no way can he get the user names of my friends" (H16). Both high-risk and low-risk users also used different applications for different group purposes, with both high and low-risk users tending to use group chat on Facebook Messenger for their work or public life, while preferring WhatsApp or Signal group chat for communications that they considered private. Hiding the metadata of group connections often to strange choices of applications, such as the usage of Crypto.cat in Ukraine despite its security flaws. Many European low-risk activists are under the impression that Telegram has better security and privacy in their group chats than WhatsApp due to mistrust of Facebook, despite this not being the case [4].

4.7 Decentralization

While centralized projects such as Telegram, Signal, or Wire prevail on the market and have large user-bases, developers (86%) and low-risk users (70%) were more enthusiastic about decentralization than high-risk users (43%). Even though they agree on the fact that decentralized systems are harder to design, their motivation to work on decentralized systems was grounded in both the political and technical aspects of decentralization. Politically, decentralization offers 'empowerment' to the user, as it gives users a means to 'control' their own data and developers believe it enables better metadata protection: "You're still not owning your data, all the metadata is controlled by a centralized system, they know all your contacts, who youre messaging at what time. I want people to run their own infrastructure" (L3). Some developers believed the choice of decentralization is inherently connected to not collecting metadata, and felt that models existed which were usable and decentralized: "With Signal it's impossible to create a decentralized system because phone numbers aren't decentralized. With XMPP it's like an email address. Even users who aren't technologically savvy can understand this is my user ID, and this is my server" (D3). Developers involved into production of decentralized protocols noticed that the market reality of secure messaging makes both federated and distributed projects less privileged in terms of financial investments than centralized projects: "It is more challenging to build federated systems because you have to coordinate with other implementers, but also the real problem is the funding! People work on XMPP clients in their free time, so it is not as perfect as a centralized system with proper funding" (D13).

Unlike developers and low-risk users, many high-risk users did not bring up the need for decentralization explicitly and were more interested in getting better software immediately. However, high-risk users did bring it up implicitly in how they formed trust relationships. Decentralization is seen both as technical challenge and social experiment, as it provides infrastructure for specific communities to organize with less dependency on intermediaries. In this sense, developers, high-risk users, and trainers tend to build associations between political organization and technical infrastructure. For example, some developers and trainers justified decentralization as mirroring the organization of anti-authoritarian social movements. In terms of choice, there was a preference for systems that were considered trustworthy politically by high-risk users. These high-risk users expressed concerns about centralized systems collecting their metadata and delivering that metadata to their local adversary, although few realized this would be possible in most decentralized systems as well, albeit in a distributed form [7].

4.8 Standardization

High-risk users tended not prioritize the use of open standards (only 40% supported), although low-risk – and almost all corporate users – were concerned (50%). In stark contrast, most developers (70%) care deeply about standards, as they felt standards were "something they would eventually be working on" (D11). Yet there was widespread discontent with existing standards bodies, as the "XMPP community is very conservative" (D13) and "the IETF is not the same beast it was in the early days" (D8). Instead, most developers shared the philosophy that they would build the application first, and then focus on standardization and decentralization via the use of open standards at a later date. In the case of secure messaging, it was still felt that more development was needed on the code and standardization would only slow down existing development efforts. Developers adopted the Signal Protocol because they felt it was the best design available, even if it was not fully standardized and they had to re-code it.

Tensions between centralization and decentralization go hand-in-hand with debates over standards in online debates within developer community. A well-known argument in favor of centralization and against standards was published by Moxie Marlinspike (Signal core developer) in his blog.[6] This blog-post, called "The eco-system is moving," has attracted considerable attention and is widely quoted by developers as a reason not to use standards, as centralization offers better control while federation can be "dangerous" in terms of security (D11), as it is hard to audit all the different implementations of the protocol and ensure correct updates. Developers from PGP, XMPP, and other protocols (Briar, Ricochet, etc.) strongly oppose this critique from Signal in their own blog-posts.[7] For example, one XMPP developer working on encryption states that the "extensibility of XMPP is not a danger in itself. A good extension will spread naturally. Moreover, there's a permanent incentive to innovate in XMPP" (D13). This has

[6] https://whispersystems.org/blog/the-ecosystem-is-moving/
[7] https://gultsch.de/objection.html

led developers in certain communities to try to standardize a version of the Signal Protocol, the OMEMO standard, in the XMPP Foundation. Signal developers are concerned about the technical competence of having third-party developers standardize their protocol. Corporate developers shared this concern, and have started their own Messaging Layer Security Working Group at the IETF in order to avoid licensing issues around the Signal Protocol and create better support for secure group messaging.[8]

In terms of PGP, developers from encrypted e-mail providers such as *riseup.net* and community efforts by PGP client developers such as "Autocrypt"[9] are working on running code first so that, in the future, PGP standards can be extended to have properties such as authentication and easier key management, but they see fixing the standards as a far-off long-term goal that is "way off in the distance future" (D8). In contrast, corporate users and developers uniformly feel standards are of utmost importance for deployment. As noted by Brian LaMacchia, Microsoft "actively participates in many standardization organizations such as IETF and W3C. We also have to play in certain national standards bodies" and that standards were necessary as "early standards in crypto were set by private companies that had fundamental patents...that hindered development."

4.9 Licensing

Viewpoints on licensing varied, although most developers (70%) preferred open-source licensing due to the ability to alter and copy code. Developers found the GPL frustrating in terms of the Signal Protocol and its lack of a standard, as it prevented the integration of their application with platforms like the Apple AppStore or in their corporate eco-system, and hoped to use the same basic protocol but under a more permissive license. As stated by a Wire developer, "The Signal Protocol is open source under the GPL that means you can't integrate it into a commercial product and thats why Whisper Systems were getting large licensing agreements from Facebook and Google and WhatsApp to integrate without opening up all of their source code" (D13). Developers, even for companies that support closed-source software, tended to support open source as a necessary compliment to open standards, noting that "Open standards people get together and agree on specifications and the specification is freely available and theres no blocking intellectual properties" but "open source provide a couple of benefits: easy to test and debug interoperability, and if you make reference then anyone can download and start to use it....In crypto, its really important you make open implementations because it's the way people audit the code to make sure people dont have backdoors or privacy-violations." GPL usage was viewed as an often vital lifestyle choice by low-risk users (60%): "If I don't like mainstream in media, if I don't like mainstream in music – why would I like mainstream on my computer?" (L2). High-risk users were concerned over metadata leaking by having to pay for applications and many did not have funds

[8] https://datatracker.ietf.org/wg/mls/about/
[9] https://autocrypt.org/

to purchase proprietary applications, and a few low-risk users preferred closed-source commercial platforms with a commitment to privacy, like Threema, and were happy to have to pay for services.

Unlike low-risk users studied in previous work [1], some high-risk users understood that open-source was necessary for trust even if they mostly did not view it as important (46%) as a working program: "All security experts whom I trust all use Signal, and we must use something that is secure and easy-going and that we can use all together so we decided to use that and we just hope it is safe. I think they looked at the source code, I did not but I have to trust them" (H15). Low-risk users tended to be more strict about licensing. High-risk users who are trainers recognized that an easy-to-use interface with cutting-edge features (including new emojis) mattered: "You can say OK we verified this application, it's legit, it does what it says. But the user interface part is key in reaching the audience. Features, looking nice, easy to use. This is what you need to have success with users" (H14). Rather than look at code themselves, high-risk relied on 'word-of-mouth' from other high-risk users in terms of code quality. No low-risk or high-risk user mentioned formal verification or academic peer review as a criteria for adopting protocols.

5 Results

In order to design protocols appropriately, the intentions and needs of users need to be brought into greater co-ordination with those of developers. However, this is not as trivial as there are distinct classes of users, ranging from high-risk and low-risk users. In addition, a subset of both high-risk and low-risk users end up being trainers that train other users. Although we did not have time to thoroughly explore the entire space of possible levels of risks and levels, it does appear high-risk and low-risk users have very different threat models, with high-risk users generally being concerned over physical device compromise and targeted active attacks on their person with low-risk users being more concerned over passive attacks and server-seizures. High-risk users defined their threat model against a local active adversary, often the police or secret agencies of their government or a nearby hostile government, rather than a global passive adversary such as the NSA. In contrast, developers usually view their threat model as the NSA, a global powerful adversary, despite the lack of attention to privacy properties like metadata collection in secure messaging protocols. While developers created their applications for high-risk users, they were in general concerned mostly with the particulars of messaging protocols, and not threats such as device seizures, although the move to ephemeral messaging in Signal shows growing awareness of this threat model.

Our first initial thesis (*Developer-User Disconnect*) is in essence correct, but needs to be nuanced. Users do want confidentiality and group messaging, as do developers. Yet in other properties there is a disconnect, as many users also want privacy protection, do not need cryptographic repudiation, and do not care as much as developers about decentralization, open standards, and open source

licensing. While key management is important to keep simple, key verification ended up being important to high-risk users, even though it was widely misunderstood as a way to prevent client device seizure. Both low-risk and high-risk users are also actively interested in privacy and even anonymization to resist metadata collection. Still, the problems are subtle in points where the application does not line up with user expectations. For example, users often believe Signal protects metadata and keeps their conversations anonymous. For example, even though Signal does not log metadata (outside the last time a user installed the application and the time of last connection), a NSA-level powerful adversary could simply watch the encrypted messages going in and out of the Signal server in order to capture the social graph of users. More easily, a captured device would reveal the phone numbers of all other Signal contacts. Although some applications such as Tox and Ricochet do achieve a degree of protection against the social network mapping attacks that some high-risk users worry about, high-risk users are in general much more aware of Signal and find it easy to use, while the anonymized Tox and Ricochet applications were unknown amongst the high-risk users we interviewed. Some issues that are important for developers, such as standards or decentralization, are not in general as important to either low or high users, but viewed positively. Licensing (and having code open-source) is equally important to developers. Interestingly, high-risk users do prefer open-source code, although they usually do not inspect it themselves. More studies of how high-risk users trust code could be very enlightening.

In terms of our second initial thesis (*High-Risk User Problem*), high-risk users have vastly different behavior and therefore properties than low-risk users. High-risk users have well-defined (if implicit) threat models, prefer open-source, are more concerned over device seizure than server seizure, and are concerned over privacy, including not just metadata collection in general but having phone numbers from secure messaging applications such as Signal being leaked or captured. Therefore, we are left with the curious situation of high-risk users in Ukraine preferring Cryptocat, which suffered from serious security vulnerabilities in early versions but did not require phone numbers like Signal. High-risk users also notice privacy-invasive behavior, such as the mining of contact information by Wire. However, high-risk users are not homogeneous, as the social and geopolitical differences between high-risk users lead to vastly different eco-systems of applications. Therefore, for future work we need to explore the differences between different high-risk groups of users across a wider variety of countries to see what generalizable patterns emerge. Note that the same likely holds for low-risk users, as low-risk technological enthusiasts are likely very different than typical corporate business users, and further work needs to be done studying the variety of low-risk users as well.

A story may be illustrative of how serious the problems faced by high-risk users are, and the issues with adoption that are faced by even the most at risk users of secure messaging. When the YPG was first adopting PGP, H1 noted that "I spent several days teaching the guy who smuggles people for the YPG to use PGP and email. In the end, he got frustrated and said 'Why do I have

to learn all this ...? We're not the mafia.' He smuggles people across the border across to ISIS territory, and each foreigner worth 100,000 USD. He uses Facebook on Android." However, H1 said that recently more and more of the YPG have been moving to Signal, with the majority of his own communication being through Signal and only "occasionally" unencrypted email and PGP, although "you'd be surprised how many people still use Facebook Messenger and normal phone calls." The reason he stated that secure messaging was that the YPG media site "was bombed a few months ago, tons of people have died, buildings collapsed. Just that one thing happened, suddenly raised the awareness of electronic cyber-defense, it gave power to the voices pushing that narrative, that we need to improve our cybersecurity. Before it wasn't seen as so important. They believe that bombing was caused by electronic surveillance. Because they targeted .... due to observation of where the signals are coming from, what they are doing and what they are saying." Although it is no doubt true that people who consider themselves low-risk users may not care or understand about issues around encryption and metadata [1], a single catastrophic event can cause users to reassess their own risk levels and adopt new tools. For high-risk users, the properties of secure messaging systems may very well be a matter of life or death, and developers of security standards should actively engage with at-risk users by any means necessary.

6 Conclusion

In the case of security properties, key management was universally regarded as a problem, even by high-risk users, but high-risk users did want to have some ability to verify contacts, although they believed current interfaces were clumsy. Due to issues with key management and usability, there was a move by high-risk users away from PGP and towards applications such as Signal and Telegram. More obscure security features, such as deniability, built on top of the cryptographic primitives themselves were not viewed as high-priority, while privacy features such as ephemeral messaging, pseudonyms, metadata protection (such as hiding phone numbers), and even unobservability in using the application, were considered of utmost importance. In general, with a few exceptions such as Briar, developers were not working actively on privacy properties, although it was considered a worthy goal. All users needed group support, but groups are defined differently across different applications, and matching user expectations around the different types of secure group messaging, which will likely require different trust assumptions and protocol choices, has not been systematically done. Decentralization is taken quite seriously by developers as a design goal, although a few of the more popular applications such as Signal have given up on it, and it is less important to high-risk users except in terms of privacy. Standards are also considered important by most developers, although most developers also believe running code is more important and standards are not a concern of end-users. Standpoints on licensing varied among both low-risk and high-risk users,

although developers show a clear preference for open-source licensing due to the ability to alter and copy code.

In conclusion, protocols for secure messaging needs to be aligned with real high-risk user needs and with real-world threat models. Addressing this disconnect will require more communication between developers and users, as well as a more nuanced understanding of the contexts in which different groups of users operate and their relational graphs (which may put users that are normally, in theory, low-risk in the high-risk category). Ideally, future work will address how knowledge flows create the formation of differing priorities between developers and both low-risk and high-risk users. A more accurate mental model is needed of the differences between high-risk and low-risk users are needed: For example, it could be possible that there is a "hierarchy of threat models" so that high-risk users have more pressing local adversaries that aim to access their devices and map their social networks, while in the absence of these adversaries, concern even for high-risk users would shift to issues of pervasive surveillance and server seizures. More detailed studies are needed to fully elucidate the differences and commonalities between the threat models of users in order to build more effective secure communications tools.

References

1. Ruba Abu-Salma, M Angela Sasse, Joseph Bonneau, Anastasia Danilova, Alena Naiakshina, and Matthew Smith. Obstacles to the adoption of secure communication tools. In *Security and Privacy (SP), 2017 IEEE Symposium on (SP17). IEEE Computer Society*, 2017.
2. Ksenia Ermoshina, Harry Halpin, and Francesca Musiani. Can Johnny build a protocol? Co-ordinating developer and user intentions for privacy-enhanced secure messaging protocols. In *European Workshop on Usable Security*, 2017.
3. Matthew Green and Matthew Smith. Developers are Not the Enemy!: The Need for Usable Security APIs. *IEEE Security & Privacy*, 14(5):40–46, 2016.
4. Nadim Kobeissi, Karthikeyan Bhargavan, and Bruno Blanchet. Automated verification for secure messaging protocols and their implementations: A symbolic and computational approach. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2017.
5. Nelly Oudshoorn and Trevor Pinch. *How users matter: The co-construction of users and technology*. MIT Press, Cambridge, United States, 2005.
6. Svenja Schröder, Markus Huber, David Wind, and Christoph Rottermanner. When Signal hits the Fan: On the Usability and Security of State-of-the-Art Secure Mobile Messaging. In *European Workshop on Usable Security*. IEEE, 2016.
7. Carmela Troncoso, Marios Isaakidis, George Danezis, and Harry Halpin. Systematizing decentralization and privacy: Lessons from 15 years of research and deployments. *Proceedings on Privacy Enhancing Technologies*, 2017(4):404–426, 2017.
8. Nik Unger, Sergej Dechand, Joseph Bonneau, Sascha Fahl, Henning Perl, Ian Goldberg, and Matthew Smith. SoK: Secure Messaging. In *IEEE Symposium on Security and Privacy (SP)*, pages 232–249. IEEE, 2015.
9. Mary Ellen Zurko and Richard Simon. User-centered security. In *Proceedings of the Workshop on New Security Paradigms*, pages 27–33. ACM, 1996.

Appendix: Questionnaire

The complete set of questions is available online.[10] A subset of relevant questions analyzed in this paper are:

- **Main Threat**: Can you define "who is your enemy"? What would happen to you if your enemy got your messages? What do you worry about more, your device being seized or the server?
- **Security (Key Verification)**: How do you usually get someones public key? In person or searching on a server? How do you usually get someones public key? Does it seem to be some third party does it for you, and if so, who exactly finds the other users for you? Do you verify keys? What do you do when your software tells you something is wrong with a key? What is, according to you, the most secure and trusted way to exchange and update keys?
- **Security (Ephemeral Messaging)**: Are you more concerned over your old messages being read or new messages being read? Do you want to search through or archive your old messages? Do you need repudiation? [Explain a transcript example]
- **Privacy (Metadata Collection)**: Do you worry about any data these tools store, and what data? Do you know if these tools store your list of contacts on their servers? Do you worry these servers could be monitored, or seized?
- **Privacy (Pseudonymity)**: Do you think you have more than one online identity? How would you describe it/them? And how encryption changes it/them ? Do you think its a problem to have several online identities? Do you need to be fully anonymous? Do you need multiple identities? Do you feel that different parts of your online identity are linked to cryptographic keys? If you have to, how do you manage these keys?
- **Group Support**: Do you use group chat? How many people in average are there in your group chats? Does the group have an administrator? How are people let in the group? Do new members of groups need to read old messages, like in a mailing list?
- **Decentralization**: Do you know if [application they use] is centralized or decentralized? Does being centralized change something for you? Do you trust the centralized server? Do you trust the people behind it? What is the worse thing that could happen to them? What is decentralization? How can you explain it?
- **Standard**: Do you know what a standards is (like HTML and email)? Is the use of open standards in messaging and email important? Is the protocol you use standardized, working towards a standardization or do you prefer not to standardize the protocol?
- **Open Licensing**: Do you know what a software license is? Whats your choice of licensing? Is being able to look at source code important?

[10] http://www.ibiblio.org/hhalpin/homepage/forms.zip