Thoughts on Mitigating Abuse in an End-to-End World

Jon Callas

14 January 2020

Introduction

This paper is a discussion document for the Stanford Internet Observatory’s workshops on Mitigating Abuse in an End-to-End World. In it, I start with the assumption that end-to-end encryption is here to stay, and within that context exploring how to fight abuse. Consequently, there are few discussions about encryption and confidentiality as such. End-to-end encryption and its role in defending human rights and personal integrity is the starting point.

The form of abuse that is most concerning and immediate to our workshop is child exploitation. Our work must also include mitigating other forms of abuse including harassment, disinformation leading to violence, etc. as well as threats to other vulnerable populations including elders and ethnic minorities.

This document is a work in progress, and this version is produced for the workshop on 11 January 2020; it likely has at least one embarrassing error in it and if fortune smiles on the author, that error is this sentence.

This article reflects the opinions and experience of its author and not necessarily any group, past or current employer, or anyone else.

The right person at the right time?

This discussion needs experts in user experience design as well as community management. Despite my experience in construction and design of encryption systems and end-to-end encrypted communications, I came to encryption through community management and building systems for collaboration and communities. I am not presently an expert in these, but once was. Future discussion needs the participation of people who are presently experts in these areas. Without them, we’re like people discussing highway safety with only experts on asphalt. The problem is bigger.
Framing suggestions

At the Internet Observatory workshop last September there were some suggestions for topics and potential mitigations. For framing, that list is:

- Taxonomies and hierarchies of abuse types
- Client-side machine learning for abuse detection
- Product designs that reduce amplification of harmful content
- Limits on group size and rate-limiting on individuals
- Product designs that label content with the original sender
- Enhanced abuse reporting mechanisms that prompt users to report without exposing them to harmful content
- Mechanisms to allow for NGO/Academic study that respect individual user privacy
- Client-side interventions powered by content fingerprinting, such as warnings that forwarded content is disinformation or might be illegal CSAM under local law
- Considerations for user discovery and the privacy of initial contacts
- Increased friction of content forwarding or amplification into large groups
- Detection techniques using already existing metadata
- Differentiated treatment of conversations based upon reported/inferred user age

Some of these suggestions are vague, or difficult to implement as a feature of a technical system, and yet valuable to the task of fighting abuse as a whole. For example, having a taxonomy and hierarchy of types of abuse is important, but orthogonal to discussing how to handle different types of abuse within a technical system in that we can reason about positions in the hierarchy without having specifics. An abuse taxonomy helps us intelligently discuss and decide how a given technical measure is appropriately used. A taxonomy also informs us what new technical measures we need to create. Without the taxonomy, we can still discuss possible mitigations even without a rigorous idea of where they go.

Thus, my axioms for this discussion include that private conversation is a basic human right, and public fora benefit from community management. This discussion is about mitigations, what they are, what situations they might be appropriate for, and how we reason about matching a given mitigation to specifics, and specifics that call for a mitigation.

Principles for Evaluating Designs and Proposals

There are a number of principles that help frame proposed solutions, and guide how good an idea it is. Among them are:
• Transparency, Consent, and Control (TCC). Privacy engineering usually looks at these as a way to guide how privacy-friendly it is. The more a user knows what something does, explicitly allows it, and can change their mind at will, the more privacy-friendly something is. There are things that are bad for privacy even with full TCC. And even the control itself can have issues. For example, location privacy is pretty amenable to TCCs making it acceptable, while needing to be more complex than a simple yes or no. Apps now can be allowed to get location when on the screen, or when they are in the background.

• Whom does it serve? As described above, a tool for the user that they use in private has fewer issues than a secretive one that serves a greater societal good.

• Various slippery slopes. Many of our objections to mechanisms or tools are precisely that we don’t object to its primary use so much as we object to an apparently inevitable mission creep. In many cases, we believe that a tool looking for CSAM will inevitably end up looking for political dissent, religious dissent, radicalization, and so on.

• Public and private, the Ben Franklin Principle. Dr. Franklin notoriously quipped, “Three people can keep a secret if two of them are dead.” I believe the ideal that two people should be able to talk privately, even when separated by distance. I also believe that posts on media like Twitter are public despite the complexities of blocking in that medium. I also believe that a three-person conversation is far more like the two-person conversation in terms of technological confidentiality than the public post. The Hong Kong protestors of 2019 used the encrypted app Telegram to coordinate not so much for the confidentiality as for its ability to support a conversation among twenty-thousand participants. The Franklin observation is a social observation that can help us understand. Just as a group of trees becomes a grove, a copse, a wood, and a forest, a group of people talking together go from being private to public.

• Adversaries vs Bad Guys - the old quadrangle: attackers and defenders vs good and bad. Not all attackers are bad; not all defenders are good. Some mitigation systems can be considered to be an attack on the abusers and others as a defense against the abusers.

• Meta-abuse. Abuse-fighting systems are often effective ways to abuse and harass opponents. SLAPP suits weaponize defamation laws against critics. Swatting uses emergency response protocols to harass and threaten people. Copyright takedowns are often used to silence people and viewpoints. All systems have drawbacks and downsides. One of the main downsides for abuse-fighting tools is that they’re also tools for abusers to abuse people. We see this in social media a lot.
Client-side filtering and its discontents

The broad idea of a number of proposals is to preserve end-to-end encryption while also filtering the content of the encrypted messages within that conversation.

Social media systems filter content today; content gets moderated, pictures are scanned for CSAM, nudity, community standards, etc. End-to-end encryption forces any sort of content examination to the client itself, because end-to-end encryption precisely makes content unavailable to any systems other than the two endpoints.

Types of filtering

All forms of filtering match a target source (text, image, etc.) against some criteria. There are different types of matching which I’ll briefly describe.

A specific match tests to see if a target exactly matches a source; obviously there can also be a list of sources. Specific matches can be complicated by them being blind (e.g. matching an image against a list of CSAM fingerprints, where we don’t have the CSAM images but can know an image is on the list) or complex searching (e.g. matching text against a list of synonyms including a number of misspellings, where we definitely have a match, but there can still be false positives or negatives; for example “I want to buy some weed killer” vs. “I want to buy some killer weed” makes that point). We recently saw instances of embarrassing false positives of specific matches on Twitter, where their hunt for offensive account names flagged people who had the surname of “Dick” (among others) as inappropriate.

Fuzzy matches are typically done by machine learning systems and return a score of how well the target matches the source.

Fuzzy matching can be done on text, but typically applies to image searches because we often do not have any other way to make a good assessment of images, particularly when they’re modified, transcoded, etc. The most presently relevant fuzzy matcher is PhotoDNA. There is a detailed discussion of PhotoDNA below; fuzzy matching in general has properties that include:

- Fuzzy matching implicitly gives a closeness score. (If it didn’t, it would return a simple yes or no and thus wouldn’t be fuzzy.) Therefore, fuzzy matches ultimately require a human to interpret the results. Even a simple, specific threshold (e.g. 90% confidence or better) is a human interpretation. We don’t have any objective rationale to select a matching threshold.
- The threshold of a score determines what is a false positive or negative. For example, suppose we have a matcher that is 80% likely to match,
and we have 100 matches. We expect that 20% of those matches (twenty images) will not contain the target. If we had a million images, we’d expect to see 200,000 false positives, and merely evaluating those is a daunting task. This basic issue becomes a real problem as the number of evaluated matches gets higher, and even more so when one is looking for an unusual event\(^2\).

- Fuzzy matching algorithms are frequently proprietary, which leads to a host of secondary issues from reliability testing to intellectual property deployment issues.
- Improperly deployed fuzzy matchers can end up being “oracles\(^3\)” that aid the adversary.
- Specific matchers obviously cannot find new material. Fuzzy matchers aren’t implicitly better, especially given the problem of false positives.

These properties are potentially undesirable for our purposes, using it in automated, client-side matching. A fuzzy matcher could swamp automated reporting mechanisms and overwhelm the ability of human interpretation. It could easily obscure the real CSAM among the false positives and make finding the abusers harder through including them a crowd of false positives, particularly given a desire to try to find new material. Lastly, secret algorithms provide their own set of challenges to protect them. Thus, there’s a special set of challenges in deploying a fuzzy matcher that doesn’t exist with specific matching.

### The “Oracle” Problem

A “oracle” is a technical term of art for a thing that can be asked a question and give back an answer. While we don’t know what answer the oracle might give before we ask, the oracle will give the same answer if we ask the same question. Clever adversaries can use oracles to defeat a detection system or even cryptography. Examples:

Let us consider a fuzzy matcher that detects cats and rejects them being posted to a forum about dogs. I, as the adversary, wish to smuggle a cat picture past it. I show the matcher a picture, and it tells me that it is a cat. I make an edit, and ask again. After a few edits, I end up with a picture that the matcher tells me is not a cat, and I have thus defeated the filter because I used it as an oracle, simply asking it, “is this a cat” until I got the answer no. I used the matcher’s answers as an oracle to defeat it.

Consider that I write malware, and I wish it not to be detected by Kaspersky,

\(^2\)This is the problem that airline safety has. There are about one billion airline travelers in the United States per year. Suppose we have a terrorist detector that is right 99.9999% of the time (it makes one mistake in a million). It will match against one-thousand people. Processing one-thousand people, each of which is 99.9% likely to be a mere traveler is a hard problem in itself, even assuming that this detector works perfectly, has no bias, etc.

\(^3\)So can specific matchers, but fuzziness implicitly makes this a larger problem.
Sophos, etc. Similarly to the cat matcher, I scan my malware with Kaspersky and am told it’s malware. Again, I tweak the software, making small changes to it until it passes the anti-virus scanner. There are factories that create malware for sale, and they offer as part of their support contract that the malware they produce will pass typical antivirus software. Again, the malware producer used the antivirus software as an oracle to defeat it.

Often matching systems ignore the problem of it becoming an oracle. Antivirus software typically does. PhotoDNA, in contrast, is secret software at least in part to prevent it from being a useful oracle.

On-client matching systems almost inevitably can be used as oracles.

PhotoDNA

PhotoDNA is an image scanner used for detecting CSAM. It was developed at Microsoft in conjunction with Hany Farid, and used by nearly every online system to scan for CSAM, including Gmail, Twitter, Facebook, and Microsoft Azure. We know relatively little about its internals; Microsoft’s web site has a basic description as well as a secondary article on the video version; there is an interesting Twitter discussion started by Matt Green of Johns Hopkins University, and a paper describing mathematics that some presume is the basis for it. Cloudflare announced a CSAM scanning tool using PhotoDNA for their subscribers on December 18, 2019. Their web page describing their service has perhaps the best description of how PhotoDNA works. From that article, we know:

1. PhotoDNA’s fuzzy hashes not only describe the image, but also can encode a similarity distance between two images via their fingerprints. Cloudflare’s blog post gives some examples of distances on images.
2. There are two hash lists Cloudflare makes available, a “NCMEC NGO” list and a “NCMEC Industry” list.
3. PhotoDNA proponents have said in the past that there are no false positives, but Cloudflare’s description says that the “distance” part of the hashes allows for different types of accuracy, which means there can be false positives.

Thus, there are a number of things we know about PhotoDNA and its suitability for client-side scanning.

- There’s a legitimate desire to keep PhotoDNA from being an oracle for abusers to learn to get around it. In all the present implementations, it is being used in a network system that keeps the PhotoDNA scanner on a server.
- It’s not a simple drop-in, which its proponents have implied before. Cloudflare describes the tradeoffs between false positives and false negatives plainly and this flies in the face of statements that deploying it is trivial.
• If PhotoDNA is to be used client-side, there must be client side protections for the algorithm itself as well as the hash lists. This implies the sort of enclave that iOS has or better, encrypted storage that can’t be accessed by the owner, and more. Perhaps ironically, putting a necessarily-secret algorithm on a client device requires that it have some sort of secure computation enclave and storage for the algorithm and its hash list that the owner of the device can’t get into. We must consider the device owner to be an adversary; a device in the hands of an abuser has valuable secrets that they will want to look at – the algorithm itself.

Public posts vs private conversations

As we have seen with PhotoDNA, truly public posts are filtered one way or another. Even a system that only examines posts after a complaint has a filtering system, albeit one outsourced to the community itself. Automating that process has its own considerations, yet there’s nothing intrinsically wrong with filtering a public post while it’s being composed because it’s going to be filtered once it is made public.

In contrast, a private conversation is intended to be for the participants alone. End-to-end encryption is the technological enforcement of that intent and purpose. Erica Portnoy of the EFF has written an essay, *Why Adding Client-Side Scanning Breaks End-To-End Encryption*, covering these quandary. Her central point is that end-to-end encryption is not merely syntactic, it is semantic. A breach of the semantic expectations of end-to-end encryption breaks the technology just as much as a syntactic break of the literal encryption would. This is a subtle, but crucial point: a private conversation with an automated, robot chaperon is not a private conversation.

Public posts and private conversations are fundamentally different both in technical details and social expectations. We must design and deploy our tools so that they preserve those expectations. Private is private, and the person’s control is basic to their personal safety, security, privacy, and even autonomy. Public posts are public and in scanning and filtering public posts just about anything goes, since people will be reading them, making their own comments, and so on. In between there’s a gray area. A conversation between two people is obviously private and eroding that violates their basic human rights. Three people are almost the same, Franklin’s aphorism notwithstanding. By the time we get to hundreds or thousands of people, it’s virtually everyone. The middle is hard, and a reason why tools described below that enhance people’s autonomy become the best tools.
Mission creep

One of our largest concerns about matching within an end-to-end conversation is how one thing leads to another. This is the essence of both slippery slopes and also mission creep. I prefer to talk about mission creep rather than a slippery slope. One of the rules of slippery slopes is that they’re never as slippery nor as steep as feared, but mission creep happens with nearly every mission, the word mission itself describes situations such as this, and the creep also describes distortion as well.

There are a number types of creep to be concerned with:

1. Jurisdictional creep. If it’s available for the US, it’s going to be available for the UK, the EU, Japan, India, and so on, and then into repressive governments.

2. Scope creep. If we’re going to use it for CSAM, then why not early warning of school shootings? If school shootings, why not other forms of domestic terrorism and massacres? If massacres, why not domestic violence? If those, why not look for teen suicide, eating disorders, and other health issues? If those why not money laundering, tax evasion? Why not hate speech? Blasphemy (a form of hate speech)? So on continuing into what we know the Great Firewall looks for, including Winnie The Pooh.

3. False equivalency creep. Some of the scope creep listed above is also from a false equivalency. For example, I shifted from violence and exploitation into public health issues. We shouldn’t mix the two.

The consequences of mission creep can be a backlash against the main mission. The recent Wired article, Please Stop Sending Terrifying Alerts to My Cell Phone describes this issue. Amber alerts, those alerting about abducted children, have expanded into Silver Alerts (seniors and those with dementia who are lost or missing), Blue Alerts (threats to law enforcement), and Camo Alerts (missing current or former members of the military). The Wired article is a blunt discussion of the mission creep of the expanded alerts. Not only do they point out that the suicide rate for members of the military is lower than that for the population at large, but that the suicide rate for construction workers is twice that of the national average and wonder why not have “Hard Hat Alerts” as well? The article even calls into question the effectiveness of Amber Alerts. Public annoyance with emergency alerts is strong enough that smartphones have features to suppress them and there are “life hack” articles describing how to...

4Here’s an example, that of control of liquids in airports started from an actual terrorist threat, where the terrorist gang was planning on mixing chemicals into explosives on the airplane. While the ban was a “liquids ban” the safety threat was in having a container large enough to mix the chemicals. Today, we continue to ban liquids while encouraging people to bring water bottles to fill them in the airport. I even recently saw a Kickstarter for a pure titanium water bottle, which is precisely the sort of container one doesn’t want.

5The government of China blocks images of Pooh Bear because some people have created memes in which Pooh represents Chairman Xi, who apparently resembles the Disney take on the beloved children’s character. Xi finds this insulting.
Mission Creep is not simply a slippery-slope argument, it has consequences that can call into question the base mission itself.

What do we do when we find a match?

This is a hard problem, particularly with fuzzy matching. Is a 90% likelihood of a match good or bad? This is not a simple question, and one that has to take into consideration what happens when there are matches, misses, false negatives and false positives. A full discussion has to happen for each filter and its consequences.

Can’t do secret algorithms on the client

PhotoDNA is a secret algorithm. Other sophisticated algorithms have reasons to be secret, so that their adversaries don’t know how they work and can’t test against them. Unless there’s on-device encryption and isolation, then the algorithm will essentially be public.

When, where, and how do we use filtering?

This is a tetchy issue. In the public space, we have a lot of freedom to apply solutions. Cloudflare now offers CSAM scanning to all its users. This is a great boon to fighting CSAM, even if there are unsolved issues with false positives with potentially serious consequences and little actionable advice. We need to discuss server-side scanning of public posts, but there’s little argument about it.

Client side scanning that is voluntary and serves the user is also reasonable. It is a huge boon to a user to be able to tell them before they post that something might be hurtful, hateful, offensive, or creepy. I’d love to be able have easy access to scanning things I receive for scams, disinformation, accuracy, etc. There’s huge potential here for rethinking how we help users defend themselves, get advice and report abuse.

Secret algorithms on a client are tetchy for technical implementation reasons. Running (e.g.) PhotoDNA on a client requires the sort of system integrity that is at least as good and ideally better than what we have today. The adversaries who would abuse others are active, adaptive attackers and they will attack the local instantiation of the secret algorithm.

Lastly, it’s axiomatic that we preserve the rights of our users, including their personal autonomy and integrity. We can still help them, though. An “is this photo creepy?” button that scans things the users receive could be server-based. It can even help fight the lesser scourges like unsolicited dick pix among adults.
Consequently, we need to have deployment and implementation include user experience designers, community managers, and others who are not us, the encryption policy experts at this workshop.

**User-centric mechanisms for mitigating abuse**

There are many technical things a system can do to mitigate harm by making them less suitable for abuse than they would be. These alternatives to filtering not only mitigate risk, but in many cases can improve security and privacy.

**Improved abuse reporting**

Facebook already has a cryptographic mechanism called “message franking.”

Message franking allows a user to unseal the confidentiality of messages to report abuse. It removes the encryption from a set of messages, but keeps with it authenticity checks so that the reporter cannot alter or spoof abusive messages. Thus, moderators can both read the offensive message and know that the report isn’t modified nor created out of whole cloth. Best of all, in the normal case this does not reveal anything about the conversation.

An app can also have user experience improvements to make it easier to report abuse. These need user experience testing and careful work as they shouldn’t make it too easy to report issues without the proper back-end staff to adjudicate them. We don’t want meta-abuse, like silencing people through imagined abuse, overwhelming human moderators with complaints, and so on.

**Group Size Limitations**

Both WhatsApp and iMessage limit the size of a chat group, 33 for iMessage, 256 for WhatsApp. Each has said that it is at least in part to combat the ill effects of rumor spread, disinformation spread, etc. More research is needed as to its effectiveness, even though it intuitively seems to have some effectiveness.

There’s an important aspect to this for fighting direct abuse – it enforces in the chat system a decision about public and private. iMessage effectively makes the statement that private conversation is limited to a group of 33. WhatsApp has the larger number of 256, and arguably that should be closer to the iMessage size, especially since Facebook has a way to have large quasi-private groups within Facebook proper.

---

6 Apologies for the technical nature of these two articles. I think the abstract and introduction might be useful if you want to know more than what is above. *Fast Message Franking: From Invisible Salamanders to Encryptment* and *Message Franking via Committing Authenticated Encryption*
Content Tagging

It’s possible to add metadata that can aid in proactive or reactive mitigation of abusive content in private conversations. Mechanisms include:

- User-identified tagging such as message franking. Message franking is a good mechanism because it conserves end-to-endness, while permitting abuse reporting that is verified by strong cryptography. Strong encryption in tools like message franking aids both the reporters of abuse, the moderators who evaluate the report, and thus limits meta-abuse problems.
- Limiting the number of times a specific post can be forwarded. This is even possible while preserving end-to-endness. This has been added to Facebook systems to combat disinformation amplifying ethnic violence.
- Showing the forwarding chain. This addresses the same issue as limiting the number of forwards. Typically, email shows forwarding simply because forwarding increases indent levels, adds in annotations, etc. These are easy to remove, but typically people do not, particularly the sort of people who forward rumor and so on without thinking.
- Showing the original sender. This is a subset of showing the forwarding chain. Facebook is already doing this with photos, (Forbes article, IPTC working group article) They seem to be doing this for information gathering rather than abuse control, but it is still useful.
- Voluntary content advice. While automated scanning is inappropriate for private conversations, there’s no reason why this can’t be under user control. Let’s assume some fuzzy, machine learning scanner for disinformation, rumors, etc. If it were easy for me to check content (“Verify this against Snopes”), it would be easier for me as a participant in a conversation to call out and push back against abusive behaviors. This could be either before sending or after receipt. The key here is that it is voluntary. Of course this is easier said than done and there are other considerations, such as issues with oracles, yet it would help combat many forms of abuse.

More mechanisms for controlling abuse

Increasing friction

“Friction” is user experience slang for how hard it is to do a task. A task that takes three taps has higher friction than a task that takes two taps. It’s well known that people do higher-friction actions less often than low-friction actions, and this can be used to mitigate abuse that occurs unthinkingly. Also, tasks are often organized so that more frequent tasks have less friction than tasks someone does less frequently. Thus, it’s a one-tap operation to send a message, but changing app preferences might have so much friction that people always
do the defaults, or need “life hack” articles to tell them how to do it. Suppose we change the friction of forwarding something so that there was a message that said, “You will be forwarding this to 17 other people” and an extra tap. This increases friction. If we listed the names of the other people it increases friction even more, and forcing someone to scroll to the bottom of the list increases it even more than that.

Some of the mechanisms in sections above are also friction-increasing because they have work-arounds. For example, if we limit the number of forwards of a news article, the person can always open the link in a browser, copy the URL and send that. It’s the same result, but with higher friction. Many other options are possible, for example requiring a user to make a comment on forwarded articles.

**Rethinking user discovery and initial contact**

We’re used to a world in which someone can make an unsolicited call to any phone number, send unsolicited mail (email or postal) to any address, text any phone number, etc. Moreover, many systems have features to reduce friction in finding people and contacting them. From Facebook Messenger to Signal, these systems encourage a user to find other people on the system.

These facets have downstream consequences. Spam email, robocalls, texting scams, and so on are all consequences of a receive-from-anyone tool. These consequences are leading to a world with reduced unsolicited contact. Robocalls are driving telephones in that direction. It is a feature of both iOS 13 and recent Android to send unknown callers directly to voicemail and similar filtering of unsolicited voicemails. iOS also will filter text messages from unknown numbers into a different pane.

A system that requires, defaults to, or even allows someone to require an introduction of some sort reduces the opportunities for many types of abuse.

**Tying identifiers to identities**

Some end-to-end systems allow a user’s identifier to be selected, others tie it to a telephone number. Telephone numbers are usually strongly tied to a person’s identity. In many countries, a person must show government identification to get a telephone number or SIM. Even in countries that do not require it (including the US and UK), other considerations such as how to manage recurring billing

---

7 I had to search for instructions describing how to turn off emergency alerts, and it’s a high-friction operation that includes scrolling to the bottom of notification preferences for every app installed on one’s smartphone.

8 As is often the case with Android systems the details depend on whose phone (Google, Samsung, HTC, etc.) and what may have been changed by carriers. Some carriers provide their own phone dialers so there’s no single thing that is accurate across the Android ecosystem.
make it difficult veering towards impossible\(^9\). Consequently, end-to-end systems such as Signal and WhatsApp which require identifiers to be verified phone numbers have the implicit ability to have a strong response to abuse.

**Scanning for fake profiles etc.**

Some of this is already done, for example Facebook’s Real Names policy. Most of it is bad. There are plenty of things that can be done, such as looking for stock photos as portraits, troll clouds, etc. All of this takes a certain amount of taste and sensitivity which hasn’t happened in the past. This has a lot of potential as a mechanism for mitigating abuse, while much more work and research is needed.

**Social graph analysis**

Many systems have public or quasi-public social graphs and these can inform other decisions as well. These can inform or advise other decisions. For example, a discussion group of teenagers that is joined by an unrelated adult can be detected while preserving the confidentiality of that group’s chat.

**Context-dependent behavior**

A number of the options I describe above are changes in user experience. These changes can be context-dependent. For example, there are health-monitoring functions of an Apple Watch that are on by default for seniors (over 65) and off by default for younger people. There are many ways to use appropriate context through stated age, membership as a dependent in a family plan, parental controls, and so on to do something like utterly turning off unsolicited contact for children (and an option for others). There are also controls that might be in place for seniors by default like increased friction on forwarding and so on. There can also be things done for parents and other care-givers for information monitoring (e.g. presenting a roll-up of everyone your dependent talked to last week), supervision (new contacts require caregiver approval) that can be put in place to help people manage potential abuse themselves.

\(^9\)While I was working on Blackphone, I experimented with getting pay-as-you-go SIMs that were not tied to me. I got a US SIM without showing ID as well as a UK SIM. In each case I had to firmly deflect requests for ID, and in the case of my UK SIM getting top-ups was annoying as required imposing on local friends or the dodgy world of web sites where one pays a premium for someone to get one a top-up code from some local corner shop.
Technical solutions that also help

There are also related problems that need to be solved that can help law enforcement and others. As an example, the problem of “data inheritance” – how one’s digital assets are passed on to one’s heirs – that are among the larger problems left to be solved and these would help the “warrant-proof encryption” problem.

Completely non-technical options

Many of these mitigations are purely technical. Others require humans in conjunction with the technology to be effective. Moreover, constructing and refining these systems requires iterative development, research, and refinement over time.

An even more important consideration is that many forms of abusers are adversaries who will learn, adapt, and develop their own countermeasures against deployed mechanisms. Combatting abuse is not something where a single mitigation or even fixed set of mitigations is guaranteed work indefinitely when the abuse is intentional. We need to fund research on the behavior patterns of abusers as that will produce better technical mitigations.

Perhaps most importantly, even though it is here at the end, the governmental response systems from law enforcement to social work needs more people, those people need more training, they need better tools, resources that technical people like me don’t know enough about to describe, and other assistance including mental health support for those boots on the ground. Combatting abuse takes an emotional toll on those who do it. Many abusers are vile people doing heinous things. Not everyone can work combatting it and even those who can often cannot do it for long. All of the support mechanisms for these people cost money and that money must come from government itself.

Similarly, the content moderators and others in the private sector are doing hard work that requires a lot of support.

Lastly, we must also recognize mitigations that merely sweep the problems under the rug. Many laws and regulations enacted to combat social problems merely push the problem to unobserved margins, do not make the problem better, or create new problems often ones as bad or worse than the original problem. History is full of these failures. All of these problems are non-technical and require participation from experts teaching, medicine, and social work along side the technologists. We have to accept that this is a hard problem and if there were a silver bullet that would solve it, humanity likely would have found it by now. The Internet is new, but humans are the same.
Summary and Conclusions

In this document, I’ve sketched and gestured towards ways that we can live in a world that fights abuse while providing for end-to-end encryption. As humans, we all deserve to live in a world in which we are left alone and can talk to others openly. The Internet has given us the ability to talk to others across distances and of like interests. That ability is usually good, but can be used for ill as well. The solution to fighting the bad cannot be to prevent the good. We must fight abuse, violence and hate while encouraging the good.

These ideas for tools include things that the systems themselves can do, as well as tools that people can use to defend themselves, and tools that caregivers for children and the elderly can protect their autonomy while enabling their safety.

I hope that this will inspire people who are more skilled than me at community management and experience design to build more, to use what we’ve learned to promote not mere engagement but thoughtful and expressive community. This isn’t a matter of “balance” or zero-sum thought. We can preserve the good while fighting the bad and we also must do both.

Appendix: Different types of encryption

There are three major types of encryption that today’s devices use: storage or “data at rest” encryption, communications or “data in motion” encryption, and chat encryption which is also communications encryption but extends past a two-party point-to-point system that gives confidentiality to groups of people talking. Each has its own semantics and peculiarities.

Storage encryption

Primarily, storage encryption refers to the on-device flash memory of a mobile device. In particular, the law enforcement concern is around the iOS encryption which is not only enforced in hardware, but has hardware enforced delays after wrong guesses and hardware-enforced erasing the device if the passcode fails ten times.

In contrast, most laptop and Android “Full Disk Encryption” permits a copy of the disk or flash to be made and then the passcode to be brute-forced by other computers or clusters of computers. This has been far less of an issue for law enforcement because of this form of attack: make a copy of the disk and then guess the password quickly and at leisure.

The device encryption that appeared with iOS 4, called “data protection” was radically different from other forms of disk encryption. The device itself tied the key management to secrets that were on the device itself. This meant that an adversary could not make an image of the device and then guess all passcodes on
that copy alone; the brute-forcing had to be done on the device itself. With iOS 8 and the iPhone 5S, this was amplified so that even a brute force on a device is not feasible, absent exploitable bugs etc. Recent high-end Android phones such as the Pixel 3 and 4 have device encryption with similar goals, including per-file encryption that can be managed in different use categories.

**Communications Encryption**

By communications encryption I mean technologies such as SSL or TLS, SSH, or VPNs. These are technologies that specifically encrypt data as it leaves one computer for the network, and goes to a second computer where it leaves the network connection unencrypted. Law enforcement presently seems willing to leave these systems as-is.

**Chat Encryption**

Technically, chat encryption is a form of communications encryption, but has been treated separately by law enforcement demands. By “chat encryption” I mean tools such as WhatsApp, Signal, Wire, Telegram, and iMessage.

These systems are presently under a lot of debate. The GCHQ “Ghost User” proposal seemed directly aimed at them, even though it was phrased in such a way that it could be taken to refer to all communications encryption. Nonetheless, most of the framing and metaphor was far more applicable to chat systems than SSL.

Chat systems in general have some separate considerations because they can include many people.