Summer 8-20-2020

The Law and Policy of Client-Side Scanning (Originally published by Lawfare)

Paul Rosenzweig

The Law and Policy of Client-Side Scanning

Byline: Paul Rosenzweig

Originally published on Lawfare.

Summary: The idea—which aims to develop systems to scan photographs and messages before they are sent or received by users—is attractive, but it has far too many technical, legal and policy uncertainties to be ripe for adoption at this time.

In recent years, two unfortunate trends have converged in cybersecurity—the growth of child pornography distributed online and the proliferation of encryption systems that enable the distribution of all forms of content (both lawful and illicit) in a manner that is inaccessible to service providers and secure from observation and interception by law enforcement. The idea of client-side scanning (CSS) attempts to disrupt this confluence.

CSS is the concept that through certain forms of technological implementation, a system could be developed to scan photographs and messages before they are sent from a user (or after received by another user) in order to determine whether the images or messages in question contravene legal prohibitions. In its most obvious configuration, an encrypted file containing images of child sexual abuse would be scanned before a user could send that file using a communication application. The image would be compared to a list of known illegal images and interdicted before the image was sent.

In some contexts, it is thought that the only way to combat the growth of malicious child pornography and other illicit communication is to degrade the security of encryption systems. But such a response would come at significant cost. As a result, CSS is sometimes seen as a potential solution that comes without the potential adverse costs of encryption degradation. It would effectively “go around” encryption by allowing the interdiction of malicious materials in an unencrypted state, even in the absence of predication for law enforcement intervention. In some ways, CSS is touted as the “ideal” or “golden” solution that allows cybersecurity practitioners to have their cake (by stopping the spread of illicit content, such as child pornography) without gaining weight (by channeling law enforcement interdiction and avoiding the challenges that arise in the encryption debate). The conception is a bit idealized, since it discounts the problems associated with the technical necessity for centralized monitoring of messages by technology companies. But the framing is one that has some persuasive force.

As an increasingly attractive policy option, CSS is ripe for evaluation. Yet precious little concerted analysis has been given to the topic.

Here, I attempt to begin that analysis. What is CSS intended to achieve? Can it do so? And if so, at what cost? In the end, client-side scanning is an attractive conceptual idea but has far too many technical, legal and policy uncertainties to be ripe for adoption at this time.

Background

The concept of client-side scanning is, in its current form, principally a reaction to ongoing challenges relating to the growing prevalence of child pornography—which in current policy circles goes by the more formal name of child sexual abuse material (CSAM)—in the internet environment and the challenge that encryption technologies pose to interdicting those images.

The current focal point for the discussion is the prevalence of child sexual abuse imagery on global networks. The production and distribution of CSAM is a multibillion-dollar business and has well-known adverse effects on vulnerable children. In 2019, more than 70 million images, videos and "related content" were reported in aid of investigations of suspected CSAM. Disruption and suppression of traffic in CSAM is, without doubt, a legitimate objective of government and law enforcement.

Yet it is increasingly difficult to achieve that laudable end. For law enforcement, potential intervention points include the points of creation, aggregation, distribution, and receipt of these images. The current situation is such that CSAM is frequently transmitted in encrypted form (securing the distribution link against intervention) and is often distributed so widely that targeted surveillance of recipients is ineffectual, if not impossible. As a result, the quest is on for a system that operates outside the channels of encrypted communications with greater coverage than targeted surveillance techniques. CSS—the idea of scanning images before they are sent or as they are received—looks to intervene earlier in the distribution chain before or after encrypted transmission occurs.

Of course, none of this would matter if it were not technologically feasible to have perfect end-to-end encryption (albeit often with imperfect endpoint implementation). Debates over encryption are almost as old as the existence of encryption techniques. They go back to the Clipper Chip controversy of the mid-1990s and continue today. Most recently, the FBI has sought to compel Apple to unlock its encrypted phones and has asked developers like Facebook to create vulnerabilities in their end-to-end encryption. Attorney General William Barr has renewed his call for greater law enforcement access to encrypted messages and CSAM transmissions, and some senators have heeded his call.

Encryption, however, also has positive benefits in enhancing privacy and protecting law enforcement and national security interests by securing communications. Thus, the debate about encryption has, in many ways, evolved into a broader discussion of efficacy and competing costs and benefits.

In that context, CSS is seen as a solution that resolves those cost and benefit questions in a way that avoids some of the more significant harms from encryption degradation. But that resolution needs to take place within a broader context. The CSS technique is generalizable. Though focused on CSAM now, CSS tools can, theoretically, be used by technology providers to detect malicious behavior in other situations where law enforcement does not have predication for an investigation or access to the underlying information. Thought of this way, the use of CSS is less a response to the "going dark" problem than it is a new concept about "going around" encryption.

Defining the Technology

Client-side scanning is a system whereby information can be scanned and flagged for review prior to transmission. In other words, CSS involves the widespread distribution of software to be used as a system for scanning that is enabled for use before communications have been encrypted for transmission. In the most common form proposed today, it involves the comparison of the "hash" of a photograph intended for distribution with a file containing the hashes of known CSAM material. (A "hash" is a computer function that converts one value into another, generally smaller, value. It can, for example, take the numerical representation of a photograph [a very large value] and convert it into a smaller number. Critically, a true hash function works uniquely—the same picture converts to the same hash and only that picture converts to that hash value.)

The most well-known program in use today, PhotoDNA, automatically compares the unencrypted information accessed by a sender on his or her personal device with a data file containing the hashes of material that has been added to a database by an authoritative source. (In rarer implementations, PhotoDNA can also be applied to a recipient.) In the event of a "match" signifying the possession of CSAM, an alert of some form can be generated by the system. Notably, the "hashes" involved are not true, pure cryptographic hashes—rather, they are what is known as "fuzzy hashes" that have a small but definite theoretical error-rate associated with the matching function.

Notably, the most commonly discussed focus for implementation of a CSS system is on the sender side of the equation, because an application or messaging system can be constructed in a way that makes the hash comparison and provision of an alert a technical precondition to sending material. This does not mean that an application on the recipient side is impossible—but in current forms it seems likely to have less efficacy.

Precisely where the client-side scanning system resides may make significant difference both practically and in terms of law and policy. It is possible to create a scanning system with one of two system architectures. First, the scanning mechanism might be resident in the operating system of a device—that is, within the Apple or Google operating system for a mobile device. Alternatively, one can also envision an architecture where the scanning mechanism is made a part of the communications application—in other words, the scanning functionality is built into the application and is part of, say, WhatsApp or Signal. As a matter of policy, implementation would, obviously, vary between the two architectures. One structure would use only a small number of operating system (OS) developers as government proxies, while the other architecture would rely on the deployment of software more broadly through many thousands of application developers.

Most of the limited current public discussion of CSS contemplates the scenario in which the scanning technology resides at the app level, as opposed to the level of the operating system. There is general agreement that scanning at the operating system level poses much more significant technical and cybersecurity issues than would be the case in an application-layer scanning system. At a minimum, OS scanning systems would be more broadly invasive and would require greater permissions with deeper access to information on a device, posing significant privacy and security risks. They would also likely have a much greater computational burden on the system that would reduce performance and efficiency of devices. While a definitive resolution of the question is probably premature (and somewhat beyond my technical expertise), it seems far more likely than not that, given the technical challenges, any CSS deployment in the near future will follow the road of distribution at the app level rather than at the OS level.

Whatever the architecture chosen, the distributed nature of CSS systems has several significant technical implications. First, the fact that the database of fuzzy hashes or a representation of them will be resident on distributed devices makes it likely that users will eventually (if not quickly) be able to determine whether an image is a likely match to the database of hashes, allowing them to devise methods for avoiding or evading its limitations.

Likewise, whether resident at the OS or app level, the unencrypted nature of most applications means that at least some aspects of the CSS algorithms will be publicly available and therefore subject to public scrutiny and deconstruction or attack. This is particularly problematic because, today, CSS hash comparison systems—like PhotoDNA—are nonpublic.

The makers of PhotoDNA have not said why this is the case. It is possible that the proprietary nature of the program may be based on the need to protect the intellectual property inherent in its development. Perhaps the secrecy is just the product of natural hacker paranoia. But some security researchers also suspect that the system's nonpublic nature may reflect its technical fragility. While it is possible that one can make the algorithms more robust to some forms of transformation, it is unlikely that a complete solution is feasible.

In other words, the widely distributed structure of any effective client-side scanning system necessarily carries with it the seeds of technical vulnerability. The more widely propagated a system is, the more scrutiny it will receive—and the more probable it is that, at some point, it will be compromised. Yet given the anticipated use case for CSS systems, they can be effective only if widely distributed, and any policy discussion must take into account this technical reality.

Implementation and Policy Questions

With many novel technologies, and concepts, one often faces a choice. Should the analysis begin with fundamental policy questions, or should it begin with existing legal limitations? For my part, since new technical implementations so often open up radically different possibilities, I prefer to start with the basic policy issues. What are practitioners and policymakers trying to achieve through the new technology? In a world unconstrained by resource limitations or legal barriers, how would this technology be used to achieve it? What is the optimal implementation mechanism? In the sections that follow, I first identify some of the implementation choices that will affect policy possibilities and then identify some of the recurring salient policy questions that will need to be resolved.

Implementation Issues

Several policy choices will be inherent in the architecture of any solution. How client-side scanning is implemented will directly impact both the feasibility and the impact of any program. Several implementation questions are readily apparent even before initial technical development is undertaken.

First, will client-side scanning be mandatory or voluntary? In other words, will the government choose to make inclusion of CSS in communications applications (or operating systems) mandatory in a manner akin to the way that access to telecommunications systems is mandated under the Communications Assistance for Law Enforcement Act (CALEA)? If a CALEA-like mandate is adopted, that will naturally make the governmental nature of the CSS system more evident, with likely legal consequences. It will also make it more probable that the government will be required (as it was with CALEA) to fund some of the implementation costs that it demands of service providers.

Alternatively, the government might choose to incentivize the implementation of CSS systems through any number of methods. Traditional possibilities, like tax credits, are plausible. But so too are legal incentives that verge on mandates. For example, some commentators have suggested that the inclusion of CSS systems in communications portions of a product be a condition of retaining the Section 230 content safe harbor—in other words, Facebook might be required to deploy CSS on its Messenger platform as a condition of retaining protection against liability for the postings of its users. While not a mandate, strictly speaking, this would naturally have significant impact.

There is also the related question of whether or not CSS deployments will have the capability of end-user control. In other words, one can imagine certain technical implementations that would allow a user to toggle "off" the CSS screening function in, say, a communications application. Presumably, if the function is voluntary for users, most malignant actors will refrain from installing the application in question or toggle the functionality off when using the application.

But if CSS is a mandatory component of any installed communications application (or operating system), that will implicate Fourth Amendment concerns (discussed more fully below), fundamental privacy issues, and even questions of statutory authorization. At a minimum, it would call into question the validity of any reliance on the consent of the user as a justification for the installation.

Second, the use of client-side scanning to identify prohibited content will require an architecture that relies on an authoritative data source—or perhaps more than one such source—that is continuously or routinely updated to include newly identified content. As I have noted, the principal proposed use for CSS is the interdiction of the transmission of illegal material (such as CSAM). But what entity will be the source of the list of prohibited content? And who gets to define the terms under which content is added to (or not added to) the prohibition list?

This raises preliminarily a question about transparency. Who knows what is in a CSAM database and who can audit it? While transparency is, overall, a positive value, the challenge here may be that enhanced transparency creates a greater availability of CSAM—a truly perverse result that reflects a risk requiring mitigation.

More importantly, the identity of the source of the list of prohibited material is of critical significance. Using a government-controlled list would necessarily implicate free-speech considerations—it would put the government in the difficult and controversial position of defining permissible content. It would also more tightly tie the operation of a CSS system to government action in ways that might implicate constitutional concerns. Likewise, private action (if mandated by the government) may be legally no different, though implementation issues in the private sector would, presumably, be somewhat less cumbersome.

At the same time, voluntary private action avoids many of the legal issues but implicates economic questions. Private actors may engage in rent-seeking behavior, preferentially favoring or disfavoring certain competitors. If the private actors are not-for-profit, they may face funding questions while for-profit entities would be motivated by economic incentives. The current sources for authoritative data are a mix: The National Center for Missing & Exploited Children (NCMEC) is a private, nonprofit 501(c)(3) corporation, whose listings are often supplemented by the private holdings of major for-profit tech providers. Notably, since some aspects of reporting to NCMEC are mandatory (as discussed below), this also raises the question of whether NCMEC is truly private or might, in the legal context, be viewed as a "state actor."

Third, how is notice of offending content provided, and to whom? In other words, what happens with the information when a positive hash match to offending material occurs?

A number of architectures of a client-side scanning system are possible, each with different implementation requirements and consequential legal and policy effects. In the simplest form, CSS might merely prevent malicious content from being transmitted—a matching hash would, in effect, toggle off the communications system.

Or, somewhat more ambitiously, in addition to prohibiting transmission, a CSS application might provide notice of that to the service provider who installed and managed it. Under current law, the provider is required to report facts or circumstances related to a CSAM violation, but the law is permissive as to what information can be included in the report. While providers today often endeavor to identify the purveyors of CSAM, they are not mandated to do so. Continuing this structure would provide greater privacy protection for users generally and still allow the service provider to act administratively to delete malignant accounts. From the law enforcement perspective, however, it would simply replicate the existing problem—lack of complete visibility into the widespread distribution of CSAM.

Hence, the law enforcement preference would be for a positive hash match to result in notification directly to law enforcement. This would make the communications providers less liable to public pressure. Unless such a system were mandatory, however, it is unlikely that users will voluntarily choose a system with direct law enforcement access. And since the notification would be a predicate for further investigation, it would have to be cabined by procedural rules describing when such matches would be transmitted and precisely what further investigative steps would be permitted.

Fourth, any client-side scanning system is dependent on the accuracy of the matching algorithm for its efficacy. One aspect of that question is the problem of false negatives—how many instances of CSAM will the matching algorithm miss, for example, because of minor changes in the content. But the efficacy question also implicates the challenge of false positives. Is there a risk of nonmalicious content being mistakenly identified as malicious?

Current matching algorithms like PhotoDNA have yet to give public evidence of true false positives, that is, a positive collision of a hash and an innocent image. However, given the obscurity of PhotoDNA code and the use of "fuzzy hash" matching, this might be a fragile conclusion. (I would welcome correction through links to publicly available evidence of a true false positive.)

A more significant false positive issue may lie in the challenge of inadequate database control. This can be addressed in part by punishing the bad-faith addition of hashes to the database, such as through imposing fines or criminal penalties. But that may not capture the cases in which the bad-faith addition is made by a government actor for an illegitimate purpose (such as censorship). Alternatively, there could be an expansion of existing systems that allow users to challenge existing hashes in the database if they think certain images are not CSAM and shouldn't be flagged as such.

Conversely, the more insidious problem may lie in mitigating the consequences when an image is incorrectly added to the database, either by machine-learning error or by purposeful manipulation. There is a history of such erroneous additions (for example, family pictures of a young bathing child), and it is not clear how quickly database holders remove incorrectly added material. For private-sector alerts, this means that content moderators will have to review the same content over and over again. For public law enforcement alerts, it will mean the diversion of scarce resources and the needless criminal investigation of the innocent.

Fifth, and relatedly, there is a fundamental question of efficacy. To be sure, PhotoDNA is a proven technology that has been in use for more than a decade. But even so, as a recent study by the New York Times demonstrated, the systematic linkage between the NCMEC database and screening systems (in this case, search engines) is incomplete and continues to yield false negatives.

There has also been some suggestion that resolving the hash comparison may sometimes take so long that the file transmission occurs before the alert is processed. Certainly, if this is true, the time lag involved in executing the scan is also worth resolving as an implementation and technical feasibility issue before deployment. Related to this is the reality that the majority of phone users around the world still use lower-end devices with limited processing power—a circumstance that might exacerbate the comparison-delay problem. High-end smartphones more amenable to running client-side scanning systems are far less prevalent outside wealthy Western countries.

And so, while no system should be expected to operate perfectly, policymakers might reasonably ask for assurances that the game is worth the candle—that is, that the system will effectively operate and integrate with communications or search functionalities—before implementing a new, pervasive, mandatory technology.

Policy Implications

Irrespective of the specific architecture that is chosen, the implementation selection will also impact more fundamental values and raise questions independent of the precise methodology adopted.

First, there is the question of hash control—that is, the practicality of controlling the critical gating function involving the creation, monitoring and management of the hash lists of prohibited products. Even leaving aside critical policy questions of who creates that list (as discussed earlier), any client-side scanning system will need to provide security assurances against degradation, disruption, denial or destruction of the hash data. It is a truism that all databases are subject to potential intrusion—and this database will be no different. Inasmuch as no perfectly secure data storage and distribution system is known to exist presently, some pretty heroic assumptions would be required to be completely confident of hash control.

This is especially true here, given the distributed nature of the CSS system. The hash database will necessarily be widely distributed as well, and available to the public. This suggests that the database will, therefore, be easily corruptible. In other words, users may be able to change the database in order to allow any content they wish to be distributed by deleting the related hash. Likewise, since hashes cannot themselves be tested to ensure their conformance to submitted material (at least not using current technologies), one cannot determine whether the hash deployer has included only hashes that are legitimately prohibited and not, say, added material to the prohibition list for idiosyncratic or political reasons that are beyond the intended scope of the hash-based CSS system.

Second, there is the issue of basic cybersecurity. Even if it were possible to ensure the security of the delivery of the hash list, it is important to further consider the consequences that will result from the inevitable hacking of device-side applications or the operating system where a client-side scanning system is resident. One can readily imagine malicious hackers taking advantage of the deployment of CSS applications by engaging in false-flag operations; creating willful blind spots; and using the CSS application as a gateway to device control as a way of enabling more significant intrusions. Any CSS system will, of necessity, have significant administrative privileges in the environment in which it operates, thus magnifying security concerns. The problems are likely of even greater magnitude if the CSS system is deployed at the operating system level. Again, no perfectly secure system is capable of development—so these risks, while theoretically mitigable, are also significant and inevitable.

Third, there is the issue of scalability and mutability. The distributed nature of the client-side scanning system scales well, in the sense that—to amend the common saying—"many phones make light work." Each phone will, of necessity, do its own processing, localizing the comparison and—hopefully—bringing to bear the processing power of many millions of devices. But there is a larger technical problem of scalability in the sheer size of the hash database.

Today, the NCMEC database contains more than 4 million distinct hashes of malicious content. It is said (off the record) that Facebook's private holdings are two to four times as large (that is, between 8 and 16 million hashes). Google's system uses artificial intelligence (also known as machine learning) to identify prohibited content and, as a result, grows daily.

Limited experience with publicly defined hash databases suggests that, at some point, the entire database will become too large to be "pushed" routinely to applications. (This is reminiscent of the phenomenon of blockchain bloat, in which blockchains increase in size as transactions grow, making their transmission and use more computationally inefficient.) Instead, providers will need to push a smaller "curated" list to end-user devices. This limited database push may, nonetheless, be of some utility, because most distributors of CSAM publish large quantities of material and only one alert is needed to develop predication for further investigation. Still, the constantly growing and constantly changing nature of the hash database will pose some interesting technological challenges that will only grow over time.

Fourth, there is the question of the form of the content. In current deployments, client-side scanning systems are envisioned as being solely photo-based (as a result of their focus on CSAM). Current architectures revolve around hashing photos or video. But at least in theory, there is no technical reason why one could not also put text on a prohibited list. Indeed, the databases would be smaller and computationally easier to manipulate. This prospect, however, moves the discussion from the narrow area of prohibited graphics to one of prohibited words—a significant policy change.

Fifth, there is the question of the commercial impact of client-side scanning on consumer devices. Obviously, this would cause increased processor usage and, thus, decreased battery life. There may also possibly be network usage effects, which may be costly for some people, and, as noted earlier, degradation in the operating speed of a device as computing power is used by the CSS application. Even more importantly, client-side scanning changes access to the device. Presumably, in order to avoid malicious reconfiguration, CSS will require locking down devices even further to prevent tampering with the scanning features. This will only exacerbate the trend of taking user control away from computer device owners—a problem that arises across a variety of digital media.

Sixth, it is essential to note the significant elephant in the room—the prospect of losses of privacy and control, and the resulting possibility of a slippery slope to greater authoritarianism.

It is not hard to imagine that authoritarian regimes would repurpose client-side scanning technology to maintain their political power. Recall that Winnie the Pooh images were banned in China because of Pooh's resemblance to President Xi Jinping. It is no great stretch to imagine a CSS hash check being deployed in service of the suppression of dissent in China. There is at least some reason to think that creating a technology that offers that possibility might be a strategic geopolitical error.

Even when deployed in a Western democracy like America, CSS will be problematic. Inasmuch as a CSS system is likely to be ineffective if voluntary, it is important to confront the reality that adopting this solution will involve a mandatory derogation of individual control and resulting loss of privacy. Under any sensible implementation scheme, users are likely to be obliged to give someone (the provider or the government) mandatory access to their devices if they wish to use certain types of applications (such as communications messaging systems). This would be a deeply disruptive fundamental reworking of American society's notions of privacy and control. Even if confined to CSAM, this development will have an inevitable chilling effect. It will create the impression of surveillance that is broader than any actual implementation—and that will come with a concomitant self-editing of discourse in society that seems undesirable.

Finally, there is the inevitable political question of subject matter and scope. If client-side scanning is deployed in the service of a fight against CSAM material, there is a nonzero possibility that the technology will eventually be repurposed to other laudable goals—first, perhaps, to counterterrorism, and then to some other task. While it seems highly unlikely that CSS will be used to search for political content—perhaps anti-Trump memes, for example—the technology is inherently neutral and could be used in any number of situations for which Americans might have greater skepticism. It is easy to imagine copyright holders calling for CSS to be deployed in defense of intellectual property rights—and it is also easy to see as well why some people might view that as problematic.

Consumption of CSAM is clearly an awful thing. But to some degree, one is forced to wonder whether focusing on distribution channels rather than the drivers of consumer behavior is the wisest policy course, as a general matter. Given the policy questions outlined above, the focus on CSS as a magic bullet has a bit of the feel of looking under the light for your lost keys.

Legal Issues

With a conception of policy challenges in hand, it is worth looking at what, if any boundaries are placed on our choices by existing American legal limitations. (Quite obviously, these would not apply in the case of a foreign-based deployment.) Here, too, the issues are many and their resolution is indeterminate.

First, there is currently no obvious statutory authorization for a client-side scanning program. If the government were to engage in CSS directly or to seek to regulatorily mandate it for development by application creators, it would first require statutory permission since no existing authority supports such a program. Indeed, aspects of the Computer Fraud and Abuse Act and the Federal Wiretap Act might be read to prohibit it—especially if the CSS system were deployed by a private actor.

Second, though a mandatory client-side scanning system imposed by law or regulation would almost certainly be more effective than a voluntary program, it would stand on shakier legal ground than any voluntary program. There is, of course, a legal history in the United States of government authority to compel access to information in the telecommunications arena. The CALEA statute from the early 1990s required service providers to construct their telecommunications systems in a manner that enables government access to those communications. To be sure, that stricture was placed on service providers at a higher level in the architecture of the telecommunications system—not, as a CSS system would be, on application developers and end users. And the CALEA precedent is a bit rickety, as it predates the technological explosion of the 21st century. Still, there is some precedent for the idea that the government can mandate certain technical configurations.

Third, as a mandatory system, client-side scanning programs would face a number of problematic constitutional challenges precisely because it directly impacts end users rather than corporate service providers. I outline some of these challenges here, though each could benefit from an extended discussion:

> - The Fourth Amendment places a limitation on unreasonable searches and seizures. Absent an exception (and none appears applicable here), the definition of reasonableness generally turns on whether or not the search in question is based on probable cause or some other quantum of reasonable suspicion. Almost by definition, the client-side scanning hash-matching function will be problematic—it will almost certainly be characterized as a search of the contents of a user's device and its very purpose is to operate in contexts where there is no suspicion of any sort. Mandatory inclusion of CSS would also negate the idea of installer consent. This is exactly the problem that the Supreme Court identified in *Riley v. California*, when it characterized mobile phones as an essential technology from which one cannot opt out—and why the court said that law enforcement access to the contents of a mobile phone required probable cause and a warrant.
>
> Nor would it appear to matter that the applications are developed and deployed by private-sector service providers. To the extent these providers apply CSS protocols as part of a government mandate, they are likely to be seen as acting as the government's agent. Only if the systems were adopted by technology providers in a truly voluntary manner would Fourth Amendment considerations seem to disappear.
>
> At the same time, it is possible that CSAM hash matching might be construed as searches that do not, as the Supreme Court wrote in *Illinois v. Caballes*, "expose noncontraband items that otherwise would remain hidden from public view." As some observers have argued, if possession of a particular item is illegal and if the method of inspection (here hash comparison, but in *Caballes* dog-sniffing for narcotics) reveals only the illegal material, no legitimate expectation of privacy is compromised. Suffice it to say, given this complexity, the legal issue is not in any way free from doubt and will be subject to litigation.
>
> - The same is true of Fifth Amendment concerns. To the extent that the client-side scanning systems are mandatory, some observers will argue that it violates the constitutional prohibition on compulsory self-incrimination by requiring senders to provide evidence against themselves. The argument seems shaky and a bit implausible. No individual is being asked to testify against himself or herself—at most the information secured is evidentiary in nature, and for more than 50 years compelled evidentiary materials have been deemed nontestimonial and outside the scope of the Fifth Amendment (otherwise breathalyzers could not be required from drunk drivers). Nor is any individual compelled to send CSAM on his or her device—that act is wholly voluntary. Thus, I suspect that the disclosure of images transmitted will not be seen as self-incrimination any more than it is when a prisoner voluntarily speaks on a monitored prison telephone network. While this challenge will likely be less successful than that

arising under the Fourth Amendment, it is a nonfrivolous concern that will, at a minimum, lead to litigation risk during its resolution.

- Third, there may be First Amendment concerns. CSAM is not, of course, protected by the First Amendment. But other speech is and, to the extent that use of client-side scanning may chill such speech or inadvertently impinge on protected speech activity, a First Amendment challenge will lie. In addition, the providers of communications services may have First Amendment claims of their own. During recent litigation over encryption between Apple and the FBI, an argument was advanced that code is speech and forcing providers to write code was forced speech. Though the issue has yet to be resolved definitively, a similar argument would apply here.

Finally, it is worth noting that overarching all of these considerations is the issue of consent—that is the argument that none of the constitutional issues identified is of consequence, because by installing the CSS-enabled communications application (or OS) the user has consented to whatever intrusions it may engage in. One should be skeptical of this argument: In the modern era (where a mobile communications device is considered an essential tool), it is likely asking too much to construe installation of a CSS-enabled system on a device as plenary consent to the intrusion. Certainly, that construction would be unpersuasive in the European Union, where the General Data Protection Regulation system has a strong form of consent—and it is doubtful, at best, in the United States. While the consent argument may, in the end, be legally sufficient to carry the day, it is fair to say that at present the question is indeterminate and contested.

What Then? Positive Answers

Taking all of these factors into consideration, it seems reasonable to reach an interim, tentative conclusion: Client-side scanning is an attractive conceptual idea but has far too many uncertainties to be ripe for adoption at this time. Serious policy analysis and legal assessments remain to be done before CSS is ready for legislative consideration.

So can other solutions be offered? There can be little doubt that CSAM is a problem of real significance, and it would be unwise to simply throw up one's hands and deem the problem insoluble. A few alternative courses of action are possible, which involve, for the most part, assessment of nonencrypted activities.

In reviewing these alternatives, it is worth sounding a brief cautionary note: The distribution of CSAM is an adaptive adversarial process. Those who wish to act maliciously are aware of and respond to the counteractions of those who would prevent their activity. As in any adversarial environment, it is possible to outline countermeasures only at a high enough level of generality to describe them without disabling their utility. While the techniques described below are, from my perspective, well validated, a full exploration of them would be inappropriate in this public discussion.

Within the life cycle of malicious conduct, the first layer of defense is simply to prevent bad connections from happening. In other words, the goal is to disrupt the discovery of malicious material, such as CSAM. After all, a critical factor enabling the circulation of malicious material is the ability to find that material somewhere on the dark web. One possible response is to find ways to disrupt discovery on the dark web so that repositories of CSAM and other malicious material are harder to find.

A second useful method of disrupting distribution involves examining nonencrypted public conduct. Even if messages remain encrypted and private, what one says in the public square (for example, on the public portions of a messaging platform) will often be indicative of malicious behavior. Service providers can also look at the unencrypted but nonpublic parts of a platform. They can, for example, examine group names, profile pictures and thread names that reveal something about content. With better intelligence about the CSAM production cycle, these signals are capable of identification.

Similarly, there might be behavioral signals within an application that would allow one to derive an inference of illegal activity. Some of these, such as a nonhuman rate of typing, might be indicative of automated activity. Others, such as membership in many cloaked groups, would be suggestive of an effort to conceal activity.

Yet another valuable method for surfacing malicious conduct is through encouraging user reporting outside of encrypted channels. In the case of CSAM, actors will need to conduct their activity, for example, in chat rooms and may often have to act publicly to “groom” victims. Making reporting easier for the victims is a vital technique. So, too, is incorporating better intelligence from nonapplication sources, such as law enforcement or the NCMEC.

One final technical way to disrupt CSAM distribution is to limit an application’s dissemination capabilities. For certain types of files, one could imagine putting restrictions on how those files are shared within a communications system—for example, by reducing the number of permissible recipients or restricting the size of files. Since much of CSAM is distributed in bulk, this would limit (though not eliminate) a portion of the problem. Of course, this solution is both over- and underinclusive in its scope, but as a theoretical matter it seems a reasonably valid approach.

To be sure, even these tools might suffer from the same slippery slope problems that CSS does. But on reflection, there is at least some reason to believe that the slope here is less slippery. Unlike CSS—where the most common hash-matching algorithm has not been made public—these tools, and the fact of their deployment, are relatively well understood. Slippery slope issues can never be eliminated in any content moderation environment, but hopefully greater transparency will mitigate that failure mode. And it bears noting that these tools do have a track record of utility—at least one messaging system reports banning more than 250,000 accounts each month because of suspected sharing of CSAM, even in an encrypted environment.

Finally, there are also at least a few nontechnical answers to the problem that merit brief mention. Most involve enhanced legislative or executive action. Without delving too deeply into the topic, it is clear that law enforcement efforts against CSAM are chronically underfunded. The NCMEC and the Internet Crime Against Children task force programs could use greater resources. In addition, certain regulatory restrictions stop tech providers from assisting law enforcement to the fullest extent practicable. It is worth considering, for example, loosening data deletion requirements for service providers and creating a safe harbor for artificial intelligence training on CSAM.

Conclusion

There is a certain strain of technological optimism that pervades a great deal of thinking about intractable questions like the challenge of interdicting CSAM. Policymakers have a hope—almost an expectation—that if technologists put their minds to it, they can find a simple solution to the problem.

Client-side scanning is another instantiation of that optimism. In the face of the pervasiveness and horror of child exploitation and with law enforcement's legitimate desire to intercept malicious material at the source, CSS is a system that affords what seems to be an elegant answer.

But the perceived elegance is, in the end, deceiving. At least in its current versions, CSS hash-matching systems are not technically robust and lack the transparency that fosters accountability. More notably, any of the possible architecture configurations would raise a host of intricate legal and policy questions. And all have yet to be answered in any definitive way, either by the technologists who might deploy the system or the proponents who advocate for its consideration.

Thus, client-side scanning has the virtue of an attractive concept—it almost but doesn't quite meet the demand that technologists just "magic harder." On closer inspection, however, it is a concept that is not yet ready for prime time. Perhaps, with more attention, the serious policy and legal issues inherent in client-side scanning will be resolved. But for now, that hard work still lies ahead.