Lawful Device Access without Mass Surveillance Risk: A Technical Design Discussion

Stefan Savage
Department of Computer Science and Engineering
savage@cs.ucsd.edu

ABSTRACT

This paper proposes a systems-oriented design for supporting court-ordered data access to "locked" devices with system-encrypted storage, while explicitly resisting large-scale surveillance use. We describe a design that focuses entirely on passcode self-escrow (i.e., storing a copy of the user passcode into a write-only component on the device) and thus does not require any changes to underlying cryptographic algorithms. Further, by predicating any lawful access on extended-duration physical seizure, we foreclose mass-surveillance use cases while still supporting reasonable investigatory interests. Moreover, by couching per-device authorization protocols with the device manufacturer, this design avoids creating new trusted authorities or organizations while providing particularity (i.e., no "master keys" exist). Finally, by providing a concrete description of one such approach, we hope to encourage further technical consideration of the possibilities and limitations of tradeoffs in this design space.

ACM Reference Format:
Stefan Savage. 2018. Lawful Device Access without Mass Surveillance Risk:, A Technical Design Discussion. In *2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18), October 15–19, 2018, Toronto, ON, Canada.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3243734.3243758

1 INTRODUCTION

This paper focuses on a problem of considerable controversy—balancing our collective security interests in allowing law enforcement to access the contents of electronic devices under court order (e.g., cell phones), and our individual privacy rights in securing our data against illegal and/or unreasonable access. In particular, much of this debate has centered around the special challenges posed by encrypted data on such devices since, in principle, well-encrypted data can resist all attempts to access it—whether by criminal actors or court-authorized law enforcement officers. This reality has driven a divide between those in the law enforcement community who have argued that new technical measures are necessary to ensure that a lawful access capability can be maintained and civil libertarians who argue that any potential technical measures will open the door to mass surveillance and inherently weaken security against other threats.[1]

Absent better solutions, the current state of affairs is one that should be unsatisfying to all sides—one in which lawful access is achieved via unregulated private sector device access capabilities that offer few intrinsic protections against abuse yet are inherently undependable and expensive. To be clear, law enforcement's access to encrypted devices today is via the "lawful hacking" approach, in which zero-day vulnerabilities are exploited to provide a mechanism for bypassing existing device protections[10, 15, 27, 47]. However, because vulnerabilities in well-engineered devices are challenging to find, their value is directly tied to their secrecy; device manufacturers, rightly, have an interest in improving the security of their platforms and will patch vulnerabilities they become aware of.[2] The government's obligation to share potentially exculpatory information with defense counsel thus creates a natural disincentive to use home-grown tools subject to such discovery.[3] Instead, today's status quo is one in which public dollars fund private corporations (e.g., Greyshift, Cellebrite, etc.) who develop zero-day exploits and the tooling used to bypass data protection on personal devices; these capabilities are then sold for profit in products and services used to "unlock" encrypted devices. Unlike the designs we will explore in this paper, vulnerability-based access of this kind provides no technical safeguards against surveillance use nor against misuse by criminal parties. Similarly, we are unaware of any regulatory oversight or statutory remedy that would prevent the sale of such capabilities to a broad range of parties.

To date, the research community has made little progress on improving this state of affairs. While there are many in the community who have strong feelings on the topic, much of this energy has focused on the policy aspects of the debate and there has been comparatively little constructive engagement with the underlying technical questions and options. Absent broad explorations of the technical design space or concrete engineering proposals to focus attention, the issue has become highly polarized—evidenced on both sides by loaded terminology (e.g., "going dark" vs "backdoors"), appeals to authority, and refutations of "straw man" arguments. As a counterpoint, this paper describes a particular technical design—one that makes explicit tradeoffs to try and simultaneously support lawful government investigatory interests while explicitly seeking to curtail risks of mass surveillance. It is our hope that in providing

ACM ISBN 978-1-4503-5693-0/18/10…$15.00
https://doi.org/10.1145/3243734.3243758

[1]We use the term "lawful access" in preference to other terminology because it specifies the *goal* of any such capability—that access is available via lawful process, but is not available otherwise—without the negative connotations or design presuppositions that most other terms impart.
[2]Bellovin *et al.* advocate for mandatory government vulnerability disclosures and implicitly argue that vulnerability patching is a small issue because appropriate vulnerabilities are plentiful and updated slowly [27]. However, both claims seem at odds with empirical reality for smartphone platforms with strong security engineering.
[3]This conflict is evident in several of the recent "Playpen" cases, notably *US v Michaud.*

a concrete design proposal we will stimulate a broader discussion on the technical aspects of this issue.

In particular, this paper focuses on exploring a single primary question: **Can mechanisms be designed to support court-ordered "device unlock" capability, while explicitly curtailing risks that this capability could be used for mass surveillance?** In pursuing this goal, we describe a prospective system design in which devices *self-escrow* access credentials (e.g., information that would allow the device to be "unlocked") into a co-located hardware component that can only be interrogated via *physical access* (e.g., via a hardware JTAG test access port that accesses internal chip state and *purposely* has no software-based read interface).[4] In this design, the same component also implements *time-vaulting*—meaning that an extended period of continuous physical access (e.g., 72 hours) must be demonstrated between before any release of information can take place. Moreover, even after this period, the hardware is only to provide valid data if the requestor provides evidence of authorization via a device-specific secret. This secret is to be maintained, per-device, by the manufacturer and would need to be requested by lawful parties under court order. This authorization mechanism is irrelevant to the primary goal of deterring mass surveillance, but addresses a secondary goal of resisting criminal access (e.g., stolen phones) or other uses outside an authorized legal process. Finally, we envision multiple mechanisms by which the use of this capability could be made transparent (i.e., that it would be difficult to covertly unlock a device in this manner and then return it to service).

We believe this design offers little benefit for mass surveillance use because the extended physical access requirement does not scale (i.e., it would require the regular physical seizure of millions of devices). In addition, it has the benefit of requiring no change to existing underlying cryptographic algorithms or protocols in use, does not rely on the government to faithfully escrow secrets and does not create any fundamentally new trust relationships. At the same time, this approach provides a means to unlock (and hence potentially decrypt) devices under court order.

In the remainder of this paper, we summarize some of the context, describe our design assumptions and threat model, explain our basic approach and implementation choices (particularly in the context of modern smartphone devices) and discuss some of the most relevant policy issues that have been brought up previously. Ultimately, this paper is a kind of Gedankenexperiment, one that tries to identify a reasonable set of design tradeoffs for how one might provide lawful access to locked devices without creating undue risks of extra-legal abuses. While this design is its primary contribution, we hope that it also helps encourage the security community to actively engage in exploring the technical aspects of controversial issues such as this one.

2 BACKGROUND

The tension between the state's law enforcement needs in obtaining digital evidence and individual's privacy interests in protecting their information against unauthorized access has been a source of debate for well over twenty-five years. While it is well outside the scope of this paper to provide a comprehensive review of the history, research efforts, and policy debates that lead up to this point, it is valuable to briefly touch on aspects of each to provide context for readers new to the domain and to understand the concerns that motivate this work.[5]

2.1 History

The modern conflict that motivates this paper is around court-ordered device access—an issue that was highlighted in the widely publicized San Bernardino Apple iPhone case. However, even though device access is a distinct issue, there is significant historic context—largely focused on real-time interception capabilities—that inform the debate and help explain the distrust and polarization that exists today.

CALEA & Clipper. In the early 1990s, the transition to digital communication technology created technical challenges for the government's existing ability to intercept voice calls. This transition drove two policy efforts that set the stage for much of the modern encryption debate: CALEA and Clipper.

The first challenge arose from the deregulation of the U.S. telephone system and subsequent rapid innovation in digital communications technology. The law enforcement community was concerned that its court-authorized ability to intercept phone communications might be imperiled as a result. In response, Congress introduced the Communications Assistance for Law Enforcement Act (CALEA) which mandated that telecommunications carriers and their equipment manufacturers be responsible for providing the technical capability to support voice wiretap requests. While civil liberties and technology groups opposed the measure, there was insufficient political will to stop it and CALEA passed the U.S. Congress on voice votes and was signed into law by President Clinton in 1994.

The second challenge arose, at almost the same time, over the use of cryptography over such digital links. Ostensibly concerned about criminal use of such technology, the U.S. proposed that telephony equipment suppliers incorporate the NSA-designed Clipper chip, which would implement the encryption, but would embed an encrypted version of each session key in the data stream. Because the per-device keys used to encrypt these session keys were escrowed with the government, law enforcement would be able to later decode any encrypted session if necessary.[6] Unlike CALEA, a strong political backlash emerged against government mandates on cryptographic algorithms as well as around the state's escrowing of secret keys in such systems. In the end, the Clipper approach was abandoned and export laws liberalized to allow the international distribution of strong cryptography in U.S. products and services.

The Clipper chip debacle became a touchstone narrative for describing heavy-handed government actions around encryption and CALEA would be an ongoing battleground going forward.[7]

[4]JTAG, an acronym for Joint Test Action Group, refers to a near-universally implemented industry standard serial interface for low-level debug access to device and chip hardware.

[5]Note that this account is decidedly U.S.-centric, both in its accounting of history and its discussion of law.

[6]At the time, the U.S. imposed tight export controls on cryptography, and thus the government hoped that removing export licensing requirements for such devices would make them attractive to the telecommunications sector.

[7]The original bill focused exclusively on the telephone network. However, subsequent rulemaking expanded that remit to include IP-based broadband providers, as well as

Together, these conflicts helped define the *form* of such debates and the language employed to this day.[8]

Going Dark. Over a decade later (2008), then-FBI Director Robert Mueller and his staff started briefing legislators on what he called the "Going Dark" problem [1]. At the time, he sought amendments to CALEA to mandate real-time interception capability for a range of Internet communications applications (e.g., Skype, Facetime, Google Hangouts, etc.) These efforts were unsuccessful, but both subsequent FBI directors, Comey and Wray, continued to advocate for further data access capability via the "going dark" narrative. Over time, the nature of this narrative increasingly focused on encryption (as opposed to service access) and that is the context it is most widely associated with today.

Surveillance and Snowden. In 2005 and 2006, a series of newspaper articles and a class-action lawsuit filed by the EFF made it clear that the NSA had been collecting a broad range of data from U.S. telecommunications carriers [2, 30, 45, 52]. Concern over these findings were further inflamed in 2013 when Booz Allen Hamilton contractor Edward Snowden leaked a large trove of NSA documents to several media confidants, leading to a broad range of disclosures concerning U.S. intelligence collection operations. These disclosures further undermined the relationship between technology companies and the U.S. government around questions of data access. As well, the resulting perceptions that U.S. technology companies might be subject to *de facto* surveillance created a competitive disadvantage for those companies when selling products and services into foreign markets. Finally, the extent of the government's collection capabilities and efforts sharpened concern from the research community about the risks of mass surveillance [53].[9]

While the Snowden disclosures concerned the activities of the intelligence community and not those of law enforcement (and indeed, the two communities have broad differences in interest, capabilities, procedures and focus), these distinctions were not always widely appreciated and distrust in government motivations accrued across all agencies during this time.

Conflict with Apple. While most of these prior conflicts focused around real-time data interception, the rise of smartphones created a new (and far more common) domain for digital evidence issues—data stored on personal devices.

As with most large tech companies, Apple has long operated a group to respond to law enforcement requests under legal process. Among the services they have provided (and provide to this day) is access to data stored on passcode-locked devices. However, two events led to a major conflict with the FBI over this issue.

some VoIP services, and there have been several (unsuccessful) attempts to amend the law to include access to Internet services as well.

[8]For example, in opposing one (ultimately successful) CALEA rulemaking effort in 2004, the Electronic Frontier Foundation (EFF) argued in terms that clearly echo today: "Building in 'back doors' for law enforcement is likely to increase the market demand for foreign competitor offerings to U.S. software and hardware products. If the FCC enacts the tentative rules, it will impair innovation and drive Internet development offshore. Another possible problem would be U.S.-based criminals or terrorists importing gray-market equipment that is 'CALEA-free.' CALEA-driven mandates will cause technologies to be developed overseas to circumvent U.S. surveillance capability" [34].

[9]These concerns were particularly exacerbated by evidence that the NSA may have manipulated at least one key NIST cryptographic standard to support their collection interests [40].

The first was a 2015 case in which a New York magistrate judge refused to issue a government-requested order requiring Apple to assist in unlocking a seized iPhone, arguing that existing legal process was insufficient to demand Apple's compliance. Although the finding was not precedential, Apple largely stopped complying with such requests at this point.[10]

The second issue was technical. Starting with iOS8, Apple configured its phones to encrypt most data by default and to make decrypting this data rely, in part, on a key derived from the passcode. As a consequence, Apple was no longer technically capable of directly extracting content from passcode-locked phones. Thus, in addition to the policy change driven by the New York ruling, Apple's change in technology further solidified their position.

This conflict came to a head in December of 2015, when the FBI recovered a phone used by one of the shooters in the widely-publicized San Bernardino mass shooting. The device, which ran iOS9, was encrypted, passcode-locked and configured to erase its contents after 10 invalid login attempts. The Department of Justice obtained a court order directing Apple to create modified firmware that would bypass the 10-trials restriction (as well as an inter-trial delay feature) such that the FBI could perform a brute force passcode-guessing attack to unlock the phone. More important than the details of the firmware itself, the government needed Apple to digitally sign this update so it would be accepted by the device.[11] Apple objected on a range of legal grounds, including claims that the order exceeded statutory authority and a First Amendment theory that demanding the creation of new code constituted compelled speech [4].

Before the case could be decided, the FBI obtained an independent means of opening the phone (a zero-day vulnerability in iOS which, by some estimates, may have cost as much as \$1M [57]). The Department of Justice then withdrew its request. Today, it is unclear if the vulnerability used by the FBI to unlock this phone is still present in modern iPhones or not. However, in the time since, there are at least two public vendors who sell tools or services capable of opening the latest iPhones: mobile forensic software provider Cellebrite, which offers to unlock iPhones via a service, and Greyshift, which offers a product (GreyKey) that exploits an iOS vulnerability to allow brute force password guessing [10, 51]. As mentioned earlier, today such vulnerability-based systems, funded by public dollars, represent a central element of the existing status quo for lawful access.

2.2 Research

Developing and understanding the technical tradeoffs between different approaches to the lawful access question requires significant expertise; it is precisely the complexity of the issue that makes the research community a key party to any debate.

In the mid-1990's in the heat of the conflict around the Clipper proposal, the cryptography community generated a broad range of

[10]Today, Apple will still provide access to the data stored on locked phones for which it is technically capable of doing so (iOS versions before 8.0) but only recognizes legal process for this service under California's CalECPA statute [17].

[11]In the rough design submitted to the court, the signature would have covered code that validated phone-unique identifiers (e.g., the IMEI) and hence the particular piece of signed binary code would not have been effective when used against any other phone [6].

counter-proposals offering technical tradeoffs seeking to provide, as Bellare and Goldwasser put it, “security against massive wiretapping” [24]. For example, in 1995, Shamir introduced the notion of “partial key escrow” whereby a portion of a secret key is escrowed with a trusted party and the remainder must be computed—thereby introducing an implicit time/cost element and thus potentially limiting abuses.[12] Bellare and Goldwasser extended this idea with a protocol that verifies that the necessary computation is contemporaneous with the recovery [24] (i.e., that this work cannot be front loaded). The same authors also proposed “encapsulated key escrow”, which imposes a time delay on key recovery [23] (again via a time hardness argument). Blaze further explored a design called “oblivious key escrow” where, rather than escrowing keys with a single party, keys are escrowed across thousands of parties using some variant of Shamir’s secret sharing [28]. These are simply a few examples from a broader literature, but virtually all are crypto-centric (i.e., focused on supporting lawful access via changes in the design of the underlying cryptosystem—typically via escrow or “weak” key constructions).

With the failure of the Clipper proposal, this topic became unpopular as a research focus. However, the recent revitalization of policy interest does not appear to have revived technical interest from the research community. As Boyd *et al.* write, “there has been little interest from the cryptographic community to explore compromise solutions” [29].[13]

Of the exceptions, most current research efforts represent a continuation of the mid-90’s cryptographic focus. Wright and Varia’s “Crypto Crumple Zones” work is an intellectual descendent of partial key escrow, whereby a cryptosystem is tuned such that messages can be decoded with significant cost [64] (one that is deemed to be sufficiently high that even governments would be dissuaded from using it for all but the most important cases). Boyd *et al.* have developed new approaches to partial and oblivious key escrow in a blockchain context such that key recovery not only incurs significant costs, but is also inherently public [29]. In a mix-context, Chaum’s PrivaTegrity chat system is designed to support anonymous chat applications that, as a policy option, allow collections of key mix node operators to cooperate to trace a particular user deemed to have violated norms [31, 38].[14]

Most recently, in May of 2018, a Wired magazine article described Ray Ozzie’s CLEAR proposal, wherein a device’s self-escrowed storage decryption keys are protected by the device vendor’s public key, and thus are decryptable by the vendor under court order [42, 48]. Ozzie’s scheme, regardless of any critiques of individual technical details, highlights and is responsive to where the policy fault lines exist today: it is device-centric, it moves trust from the public to the private sector, and it is relatively simple to understand.[15]

[12]This invention is cited in [24] as a private communication at the 1995 Crypto Conference and again presented at the Key Escrow Conference in September of 1995 in Washington D.C.

[13]In a similar vein, Bart Praneel argued in his invited talk at the 2016 Eurocrypt that “we don’t actually know if secure-third-party access is possible because it’s a ‘taboo’ research field”.

[14]Perhaps due to the strong negative reaction to his description of this tracing option, this section is no longer present in the current version of the paper.

[15]This work shares a number of features with Ozzie’s proposal, including self-escrow and vendor involvement in authorization. Some of this is synchronicity (self-escrow), but some is cross-pollination (vendor involvement), as we discussed the problem in detail with Ozzie on multiple occasions.

2.3 Policy

While there is a dearth of contemporary technical work focused on the lawful access question, there has been considerable policy writing—both from the technical and policy communities.

The best known of these efforts from the technical community is the 2015 “Keys Under Doormats” paper, that advocates strongly against design mandates to support law enforcement access to encrypted data. Absent any proposal to respond against, the Doormats paper considers the problem writ large, and its conclusions are built around three principal claims about lawful access: first, that it would require an abandonment of best practices in cryptosystems (in particular, forward secrecy and authenticated encryption); second, that it would substantially increase complexity and thus create new security vulnerabilities; and third, that it would create attractive targets where escrow keys were concentrated *en masse* and online [20]. While we agree with many of the concerns elucidated by these authors, the paper is frequently taken as a blanket proscription or demonstration of impossibility, which we believe is premature, if not overly dour. The reality is that there has been relatively little contemporary exploration of the technical design space or the range of tradeoffs that are possible.

In addition, a range of policy papers (and countless blog posts) have been released over the last twelve months to structure the discussion around lawful access capabilities and engage in some advocacy around particular concerns. These include the recent National Academies’ “Decrypting the Encryption Debate” study[12], Riana Pfefferkorn’s “Risks of Responsible Encryption” whitepaper[50], Access Now’s “Encryption in the US” report[13], the EastWest Institute’s “Encryption Policy in Democratic Regimes” report[14], and R-Street’s “Policy Approaches to the Encryption Debate” paper[19].

The most clear common thread across virtually every discussion on this topic is a concern that any lawful access mechanism would create security vulnerabilities that could ultimately be exploited by unauthorized actors. This sentiment is particularly driven by concerns about the weakening of core cryptographic algorithms; the argument being that it isn’t possible to weaken the algorithms for just “one side” (i.e., NOBUS; “NObody But US”). Indeed, the remarkable story of the Juniper NetScreen crypto circumvention makes concrete the abstract concern that even idealized covert manipulations of cryptographic protocols to provide arbitrary “protected access” can create real risks of third-party abuse [32].

The second major thread that appears across recent publications is a concern about surveillance risk, frequently motivated within the framework of human rights concerns. Even among those with a willingness to support the law enforcement position that encrypted data presents a critical investigatory burden, there is concern about the potential for overreach and that a capability intended to support criminal investigations not inadvertently provide a mechanism used for large-scale surveillance.

It is these two concerns that have primarily motivated the tradeoffs in this paper and it is these two issues to which we have tried to be most responsive. In particular, they have driven our focus on system-level (as opposed to cryptographic) approaches, prioritizing resistance to mass-surveillance use, avoiding the use of master (i.e., multi-device) secrets, and avoiding creating new trust relationships with government agencies.

3 ASSUMPTIONS

Any system's security rests on a series of assumptions: assumptions about the problem, assumptions about the solution and assumptions about the threat. Here we attempt to make explicit what assumptions we are operating under, including those concerning the nature of the problem, the design criteria we are seeking to meet and the threat model we assume.

3.1 Problem motivation

We start by declaring a set of three high-level assumptions upon which our work is motivated. We make these points explicit because we wish to separate debates about the technical approach embodied in our work and the claims we make about it, from related but distinct debates about political or social policy.

- *There are reasonable societal interests in allowing law enforcement to access encrypted data in particular cases.*

In general, most people understand that law enforcement provides an important societal function and that, in service to their investigations, they may be authorized to search and seize personal property (e.g., via a court-ordered warrant predicated upon a finding of probable cause). The premise behind providing these extraordinary capabilities is that reasonable societal interests in enforcing the rule of law can, at times, outweigh individual rights to liberty and privacy. This view is not universally held. For example, those with strong anarchistic or libertarian political beliefs may reject this notion entirely. Absent agreement on this question, it is definitionally impossible to compromise around the topic of lawful access.

- *There are reasonable concerns about the potential to abuse any system that would allow access to encrypted data.*

Even if one holds that there are situations wherein the state should be able to access encrypted personal data, this does not mean such a capability should be absolute or unhindered. While requirements for judicial authorization do provide one bulwark against abuse, if the underlying mechanism for lawful access is technical in nature there is no guarantee that those legal processes will be followed. Moreover, outside the domain of law enforcement, the intelligence community collects data at a different scale, using different bodies of law and legal interpretation. A reasonable person who supports lawful access for the purposes of law enforcement, might yet be concerned about any means of technical access being used at scale for such intelligence collection purposes (e.g., mass surveillance). Finally, the ability of technical mechanisms to discriminate among users (i.e., and allow government access, but not criminal access) is based on assumptions that the mechanisms are implemented correctly and critical secrets are kept secure. Thus, if the mechanisms and their constituent secrets become known outside the government, or if they have flaws that are discovered by others, then non-governmental actors might make use of the same technology for criminal purposes. Some disagree with this concern, and believe that it is possible to maintain mechanisms that would only be available to the state and/or that concerns about mass surveillance uses are unwarranted or unreasonable.

- *Lack of perfection is acceptable.*

It is the essence of compromise that both "sides" in a debate may need to sacrifice some of their goals. In this situation, it means that some interests of law enforcement will not be met (e.g., this proposal contains no way to gain expedient access to an encrypted device in an emergency) as well as some interests of civil libertarians (e.g., this proposal does not defend an *individual* against a breakdown in the rule of law or pervasive compromise of device vendors, although it does offer *collective* protection against mass surveillance). This paper operates under the assumption that such tradeoffs are inherent and acceptable so long as they do not, on balance, undermine the overall pragmatic benefits of the resulting capability. However, those with absolutist views (on either side) may reject any solution that provides less than complete privacy or complete access in all cases.

3.2 Design goals

Given these overall assumptions, we have chosen to focus exclusively on access to encrypted data on personal devices (e.g., phones, tablets, laptops) using standard features as provided by the operating system. Thus, we restrict ourselves to the problem of "device unlocking"—the process by which a device with system-encrypted storage (i.e., encrypted via a default OS service) is brought into a state whereby its contents are accessible. We do not attempt to address the thornier problems of encrypted communications, encrypted data stored by network services, nor the challenges posed by third-party applications that impose their own layers of encryption. We eschew these issues not only because they are significantly more complex but, more importantly, because we believe that they are substantially less important in practice. Indeed, while well-disciplined individuals may impose independent cryptographic controls on their data, for most users (including most criminal suspects), operating system-protected controls will subsume access to most forms of data.

Within this focus, our primary design goal is to ensure that any lawful access mechanism is not able to support a mass surveillance use case, with a secondary goal of addressing risks of criminal abuse or covert use. More concretely, in priority order we desire the following properties in a system:

- *Non-scalability.* That any lawful access capability is sufficiently constrained that there is no practical way to use it for the purposes of mass surveillance.
- *Authorization.* That the use of this capability is only made available subject to existing duly authorized legal processes (i.e., as via a warrant).
- *Particularity.* That the capability to access a given device is only useful for accessing *that* device (i.e., there is no master secret to lose).
- *Transparency.* That any lawful access capability used to access a device is transparent and cannot be easily concealed from the user.

3.3 Threat model

Our threat model encompasses a range of threats including criminal actors, rogue law enforcement officers, and well-funded organizations interested in large-scale intelligence collection. We anticipate that these actors may have access to significant resources including technical expertise, computation, and a non-trivial per-device budget.[16] However, we assume the adversary has no unique knowledge that would allow them to bypass cryptographic primitives whose computational hardness is commonly accepted as infeasibly difficult. Moreover, our baseline is the security offered by existing well-engineered devices and not the hypothetical security offered by hypothetical devices.

We separate between threats at two different orders of magnitude: threats to the privacy of a single device and threats to the privacy of large numbers of devices.

The former scenario represents the interest of an adversary in unlocking a *single* device to which they are not legally authorized to access. We assume that the adversary may have physical possession of the device, can manipulate it directly via all wireless interfaces, external wired interfaces, via direct manipulation of internal hardware interfaces (e.g., JTAG), and may monitor any passive side channels (e.g., differential power analysis, timing, etc.).

We do not include the security of the device design or code in our threat model in light of the innate fate-sharing argument (e.g., if Apple's iPhone code is compromised at the source, or can be forced to accept firmware updates, then there is little our design can do to protect the user). Similarly, we do not consider covert physical device tampering (e.g., if the adversary covertly replaces a phone touchscreen with a replacement that records the user's passcode). While we have attempted to anticipate a range of active attacks (e.g., Vcc glitching, overclocking, fault induction) we recognize that a highly sophisticated adversary may be able to decap and directly measure even tamper-resistant hardware (e.g., via scanning electron microscopy) or directly manipulate internal hardware state and circuit pathways (e.g., via focused ion beam probes) [33, 35, 60]. We are not sanguine about current devices being able to resist this level of adversary and, while we consider such actors at times in our work, we make no significant claims that our design would be able to provide such protection. Finally, our threat model explicitly does not include breakdowns in the rule of law. We fully recognize that even well-functioning democracies produce imperfect judicial decisions. However, we argue that this situation is in no way unique to technology and represents the nature of the fundamental tradeoffs arising in a civil society.

The second scenario we are concerned with is one in which any actor, at any level of sophistication, is able to bypass our design in a way that very large numbers of phones could be accessed remotely and covertly (i.e., mass surveillance). We have focused much of our design on trying to avoid enabling any such capability.

4 BASIC DESIGN

The personal devices we consider (e.g., smart phones, laptops, tablets, etc.) offer the ability to encrypt user data using a combination of device-derived secret keys and user-derived secret keys. While the implementation details can be fairly involved, inevitably access to all such keys via the operating system is anchored by a passcode or password that, when properly entered by the user, "unlocks" the device and can be used to derive or access all critical cryptographic material.

[16]Public reporting has indicated that Cellebrite charges US law enforcement customers somewhere between $1,500 and $5,000 (per phone) to unlock the latest iPhones [51, 59].

For example, in the case of modern Apple iPhones, the system combines a secret per-device UID and a user passcode to form a *passcode key*. This key, in turn, wraps other keys that then may be used to protect individual files. Note that central to this design is the fact that the UID and the passcode validation logic can only be accessed via the "Secure Enclave" component of the processor and its state is not externally visible or easily modifiable. For additional details see Apple's "iOS Security Guide" white paper [16].

In our design, we focus entirely on providing independent access to this passcode.[17] Thus we impose no changes on any underlying cryptographic algorithms, secrets or protocols used to secure device state and similarly introduce no additional security concerns to their operation. The core question of our work is this: how to safely store the passcode such that it may be reasonably accessible under court order but, in so doing, not unduly enable use that is either unauthorized or might support mass-surveillance activities?

We describe our basic design approach to this problem here, starting with two features—self-escrow and time vaulting—primarily motivated by the goal of foreclosing mass-surveillance use.

Self-escrow

Self-escrow refers to the idea that when the user sets the device passcode, this value is escrowed *on the device itself* rather than with any external third-party. Third-party secret escrow poses a range of challenges, including the need to trust a new third party, the complexity of dynamic escrow protocols themselves and the difficulty in knowing if the escrowed information has been used. By physically locating the *escrow agent* on the device itself, many of these challenges can be sidestepped (although versions of these issues will re-emerge when we discuss authorization). Moreover, to enforce this invariant, our design mandates that the escrow agent is a *write-only* hardware component (i.e., one not readable via software) and the only means of querying this component is via a direct physical connection (e.g., via dedicated pins on the hardware package). Thus, access to the escrowed passcode requires both physical possession of the device and, at least partial disassembly. By insisting on affirmative physical possession, this design forecloses the creation of any new remote access vulnerabilities. More importantly, physical scaling properties impose practical limits on how such a capability can be used; it does not lend itself to mass surveillance use since seizing all devices is likely impractical. A final benefit of this physical-centric approach is that we believe it provides a more intuitive compatibility with common understandings of the government's reasonable law enforcement powers (e.g., the ability to seize physical property under court order) than more information-centric approaches, which may appear covert by comparison.

Time vaulting

While the physical access requirement is a strong one, *time vaulting* makes it even more stringent, by requiring that physical access be maintained over an extended period of time before an escrowed passcode will be released. In particular, we require that the escrow

[17]We use the term passcode in this paper, but we intend it generically to mean all sorts of deterministic user-provided digital credentials including passwords, pass phrases, etc.

agent only respond after being continuously contacted by a requestor (i.e., via its internal physical interface) for the entirety of a *lockup period* (e.g., 72 hours). This requirement further limits the utility of our design for surveillance purposes and makes it unattractive for covert use as well (e.g., "sneak and peak" or a range of supply chain attacks). Moreover, while a modest access delay may create some inconvenience for law enforcement in time-sensitive circumstances, these appear to be the rare case in most processing of digital evidence.[18]

Authorization

While self-escrow and time-vaulting greatly frustrate large-scale use, they offer little protection for an individual stolen device nor protection against a rogue law enforcement actor (i.e., who has seized a phone without a warrant). Thus, a secondary element of the escrow agent design is an authorization protocol whereby, after the completion of the lockup period, the requestor is required to provide evidence of explicit authorization before the device will be unlocked. This evidence is provided by secret per-device state shared between the escrow agent and the device manufacturer. A lawful actor could obtain a court-order, served upon the manufacturer, to obtain this *particular* per-device authorization key. However, such a key—even if stolen—should provide no purchase on any *other* device.

Transparency

Finally, to minimize the value of this design for covert use, the escrow agent permanently modifies any device that is interrogated and whether or not it revealed the passcode. Upon startup, device firmware can detect this modification and alert the user that the device's passcode has been provided. Optionally, the manufacturer could alert the device owner via a separate protocol after they have been compelled to provide the per-device secret.

5 IMPLEMENTATION ISSUES

In this section we explore the implementation issues posed by this design in more detail, along with a discussion of potential threats when compared with the existing status quo. For the purposes of exposition, we will frequently describe this prospective implementation in the context of the Apple iPhone, but we do not believe that our discussion is unique to that class of devices.

5.1 Self-escrow

The core requirement for the self-escrow agent is that its stored value (the passcode) cannot be accessed except via an authorization protocol requiring physical access. This in turn can be decomposed into three separate design elements:

- Privileged (i.e., OS only) write-only interface for passcode update (via CPU)
- Protection of passcode storage
- Physical access requirement for authorization requests

The first of these requirements is relatively straightforward to implement. A simple microcontroller, with a serial interface and a Flash block for storage would suffice (e.g., when the OS was invoked to update the user's passcode, it could then send a copy of the new passcode to this device). However, for cost and design time, a more likely implementation context would be within existing "secure processor" environments (e.g., Apple's SEP, Qualcomm's SPU). For example, in Apple's "Secure Enclave" implementation, a separate ARMv7a core, called the Secure Enclave Processor (SEP), is dedicated to implementing security features. The SEP maintains its own hardware isolated RAM, NVRAM and memory-mapped IO, runs its own operating system (a variant of L4), and implements a secure mailbox facility for controlled communication with the Application Processor cores [16, 21, 58].[19]

This leaves the second requirement—protecting the secrecy of the passcode storage itself. The most important protection is simply that there is no "read interface" exported by the escrow agent and so there is no mode of operation whereby software can request the value of the passcode or read the memory used to store it. This design is similar in nature to how Apple's SEP stores secure keys (e.g., the per-device Unique ID AES key, or UID) that are only available to the SEP for use and cannot be read by any application processor core [16]. That said, there is a broad and creative literature documenting techniques for extracting protected state based on side-channels that leak information as the state is used (e.g., via power, timing, RF, etc.) In part to address this real threat, our design *does not access the self-escrowed passcode*, or any value derived from it, at any time before a complete and correct authorization request is validated by the escrow agent. Since our design has no side-effects that depend on the passcode value, most avenues for meaningful side-channel attack are implicitly foreclosed.

However, another potential concern is direct physical attacks. As mentioned earlier, well-resourced attackers could, after delayering via careful processing and polishing, employ Scanning Electron Microscopes (SEM) to directly image chip circuit layouts and then, employing Passive Voltage Contrast (PVC), potentially extract the charge state of individual memory cells. To date, the best published work we are aware of has demonstrated resolving Flash cell contents down to 0.21 microns in a traditional planar CMOS process [33]. We cannot rule out that such attacks may be possible against more advanced System on Chip (SoC) targets (e.g., for reference the iPhone 6S A9 processor is reportedly fabricated with 16nm features in a 3D FinFET process [39]; this is roughly an order of magnitude smaller). There are a range of protections that have been developed and proposed for addressing such attacks including obfuscated circuit layouts, tamper detectors that drain charge and internal encryption using PUFs (see Mishra *et al.* for a thorough review [46]). That said, we are aware of no foolproof solution to this threat, but we believe it is not unique to the problem described here. For example, an adversary able to read out the state of Apple's SEP (particularly the UID and "effaceable key") could also likely bypass the system's existing cryptographic protections on stored data (e.g., via offline brute-forcing the passcode key).[20] It is worth remembering that such attacks are expensive, time-consuming, and error-prone, as

[18]Scanlon describes digital forensic backlogs of 6 to 12 months as being common [55] and third-party lawful process production times are routinely weeks: e.g., Linkedin (3 weeks) [18], Facebook (2–6 weeks) [3, 62] and Ebay (20 business days) [5].

[19]Moreover, note that because the SEP handles passcode processing, any passcode entered is already available to it (i.e., there is no *new* risk created by communicating the passcode to the SEP).

[20]Indeed, we suspect that—absent other protections—the state of Apple's UID is likely easier to read out than a Flash cell because the typical implementation of eFuses are larger and create clear circuit discontinuities.

well as destructive, and thus unlikely to be a widespread mechanism used either for lawful access or by criminal actors.

Finally, we require that authorization requests are only possible with physical access to the device. This requirement eliminates the option of using wireless interfaces or existing wired ports (e.g., USB, Lightning) for authorization. Moreover, given the atypical nature of lawful access needs, we believe it is reasonable to impose a more significant physical access burden on the requestor. Consequently our idealized design is that the *only* channel that will accept authorization requests should be via dedicated chip pins that are otherwise disconnected and require disassembly of the device to access (e.g., direct access to the circuit board, if not its removal from the device casing). Again, such a requirement may impose significant reengineering costs and so a reasonable approximation for fielded devices is to leverage the JTAG pins already implemented by existing SoC implementations.[21]

5.2 Time-vaulting

Recall that the time-vaulting requirement is that any party seeking to request access to the escrowed passcode will need to demonstrate continuous physical possession over the lockup period (e.g., 72 hours).[22] There are two aspects to implementing this requirement:

- Securely measuring the passage of time on the escrow agent
- Validating evidence of physical possession

The first of these is, by far, the most important since the benefits of time-vaulting are weakened by the degree to which an adversary can artificially manipulate the escrow agent's perception of time. Moreover, while simple on its surface, secure time measurement is a tricky issue. Since the adversary may control the external environment one cannot count on wireless timing signals such as via GPS or cellular protocols to establish ground truth. Instead, one can rely on the stability of an internal hardware time source (e.g., typically based on a combination of a local oscillator driving a periodic signal and counters). Alternatively, one could implement a base timer in terms of a tuned number of iterations through a complex hardware function.[23] Assuming some such form of hardware timer is implemented directly on the escrow agent (with its memory and I/O separate from the rest of the device) this is likely to be effective against any *purely software-based attack*.[24]

However, a technically skilled attacker might choose to attack the hardware and manipulate the "count" of any timer, either via techniques such as electrical glitching or directly introducing faults to key memory cells (e.g., via Focused Ion Beam probes). Such vulnerabilities have been widely explored in the literature and can be substantially mitigated by building logic circuits that embed error correcting codes (i.e., that random glitches will be detected as faults and even specific changes will only be effective if multiple particular bits can be changed together [44, 49]).

An even more sophisticated (and heavily resourced) attacker might instead choose to attack the underlying time source itself by injecting a faster clock signal (overclock) coupled with aggressive heat transport (e.g., liquid nitrogen cooling) to convince the processor that time is running faster while still maintaining the processor's integrity. For example, most common smartphones (including those manufactured by Apple and Samsung) use a 24Mhz crystal as a base frequency source (subsequently multiplied to drive a PLL that defines the SoC's clock rate). Replacing this crystal with a faster one (or manipulating the multiplier) would in turn lead to a faster clock rate and hence an accelerated model of real time. That said, there are practical limits to how much time shifting can be expected from such an attack as processors will increasingly fail as they exceed their operating frequency range.[25] Moreover, a careful implementation might attempt to block overclocking (e.g., by imposing a bandpass filter at frequency input) or to detect it (e.g., monitoring current draw, thermal load or monitoring changes in the delay characteristics of key circuits [41, 43]).

We note that this need for a secure time measurement function is not unique to this design. Indeed, the secure timer in Apple's SEP is essential to their defense against brute force attack (and it was the request to bypass this function that was a key aspect of the Department of Justice request to Apple in the San Bernardino case [6]).

Finally, in addition to imposing a lockup period, we also wish to obtain evidence that the requesting party maintains physical possession throughout. This property is of secondary importance, but is meant to reduce risks of covert use (e.g., where the requestor acquires access to the device temporarily, starts the authorization request process, returns the device and then reacquires it after 72 hours). The simplest solution is one that requires a regular "heartbeat" message from the requestor (e.g., every 100ms) or else the escrow agent's lockup period restarts. To foreclose covert attacks in which additional componentry is surreptitiously added to the device we require that, during the lockup period, the device cannot be unlocked and displays an appropriate warning message (e.g., "Lawful Access Attempt in Progress"). We note that while more complex protocols could be employed to increase the demands on the requestor (e.g., a series of regular computational challenges from the escrow agent to the requestor) it is unclear what additional security would be provided by such designs beyond simply increasing the costs borne by law enforcement.[26]

5.3 Authorization

After the completion of the lockup period, an escrow agent then requires the requestor to present evidence that it is authorized to unlock the device. Moreover, we desire that this evidence have *particularity*—that it only be useful for one device, and should

[21]JTAG is typically not exported to external interfaces and yet is flexible enough to support the needs of our authorization protocol.

[22]Note that the 72-hour period, while arbitrary, was not selected at random. While there are a range of normal opportunities for one to become temporarily separated from a personal device, we hypothesize that most of those (e.g., a weekend trip) will be shorter than three days. We would not recommend a time shorter than 72 hours for this reason.

[23]This is how Apple implements its base 80ms passcode validation requirement using the AES engine on the SEP [16].

[24]Memory disturbance attacks, such as RowHammer [56], are also unlikely to have purchase given an isolated memory and memory interface.

[25]Aggressive PC overclockers have demonstrated speedups of up to 2x in PC settings, although it is unclear if one could expect similar effects on smart phone SoCs [11].

[26]In a similar context, some have proposed using cost itself as a limiting factor—placing an abstract computational demand on the requestor so high that only a state actor could bear it [64]. While intriguing, we do not feel confident about how to select workfactors such that they are feasible for the range of lawful criminal digital evidence needs and yet still out of the reach of organized criminal actors. This problem is exacerbated by ongoing changes in computational capability over time.

someone manage to steal such evidence, that it would provide no value for accessing any other devices (i.e., no "master keys").

5.3.1 Device protocol. We present one approach to address these requirements as follows: for each device d the manufacturer maintains a unique random device identifier d_{id} and an associated random authorization key d_{auth}. The collection of these tuples—one per device—forms the *authorization key database* and is maintained by the manufacturer.[27] Thus, at the time of device manufacture, d_{id} is burned into fuses on the device, along with a cryptographically strong one-way hash of the corresponding secret $H(d_{auth})$.[28] With physical access, a device's escrow agent can be queried and it will provide—only after the conclusion of the lockup period—its unique identifier, d_{id}. However, learning d_{id} is only the first step for lawful access and the requestor must then be able to provide the corresponding d_{auth} to the escrow agent. If the requestor is able to produce such a value, the agent can validate it by hashing the request and comparing with the locally stored value $H(d_{auth})$; if the hashes match, the escrow agent can unlock the phone using the escrowed passcode. Note that even if an adversary is able to use extraordinary methods (e.g., SEM, FIBs) to extract the value $H(d_{auth})$ from the escrow agent, this still will not let them produce the challenge value d_{auth}, due to the one-way nature of H.

Passcode encryption. Thus far, this scheme assumes that the device is able to maintain secure isolated storage (for the self-escrowed passcode) and that the escrow agent logic used to validate the authorization key is secure as well. However, this is an assumption that might be reasonably contested. Indeed, the unknown nature of the iPhone vulnerabilities exploited by Cellebrite and Greyshift have led some in the security community to argue that such a secure processor environment "does not currently exist" [37]. To address this concern, we can reduce this risk by separately encrypting the self-escrowed passcode so that compromising the secure environment *after* a passcode has been set provides no purchase on the plain text passcode (obviously, if the device can be compromised covertly before the passcode is set then there can be few meaningful defenses).

One approach to achieve this goal, similar to that in Ozzie's CLEAR proposal, is for the manufacturer to generate, for each device, a public/private key pair (d_{seal},${d_{seal}}^{-1}$), storing the per-device private key, ${d_{seal}}^{-1}$, in the authorization key database (indexed by d_{id}) and the corresponding public key, d_{seal}, burned into its device's fuses (along with d_{id} and $H(d_{auth})$ as described earlier). Upon setting a new passcode p, the device encrypts it with d_{seal} and self-escrows the resulting value ($d_{seal}(p)$). Thus, the stored passcode data is only useful to a party with both a means to read the value $d_{seal}(p)$ from protected storage *and* knowledge of the associated private key ${d_{seal}}^{-1}$ needed to decrypt it. An authorized party, will be able (after the mandated lockout period), to present the device with an authorization request message containing both the candidate authorization key (d_{auth}) and the associated private key (${d_{seal}}^{-1}$). If the authorization key is validated, the private key can then then used to decrypt the escrowed passcode and unlock the device. One drawback of this approach however is that the use of asymmetric cryptography introduces additional implementation complexities including a need for secure random number generation, a more complex hardware implementation (or a software implementation and its attendant risks) and an unclear level of protection in a future post-quantum world.

An alternative design would be to use symmetric keys, but within a forward secure framework to protect against key recovery. As a strawman design, we mandate that the manufacturer generate an additional random initial key, d_{seed}, for each device. This value is stored in both the authorization key database (again indexed by d_{id}) and used to initialize a key, d_k, on the corresponding device in non-volatile storage managed by the escrow agent (i.e., the initial value of $d_k = d_{seed}$). In addition, the escrow agent also maintains a non-volatile counter, c, to represent the number of times the passcode had been set (initialized to 0). Each time a new passcode p is configured on the device, it is encrypted with the current value of d_k before self-escrow (i.e., the escrow agent stores $d_k(p)$ in non-volatile memory), then d_k is *overwritten* with $H(d_k)$ and c is incremented. Thus, the key necessary to decrypt the current passcode is not stored on the device and the current key stored on the device offers little value due the one-way nature of H. By contrast, an authorized party can, after the lockout period and after d_{auth} is validated, present the device with the original d_{seed} value from the authorization key database. By applying H to d_{seed} $c - 1$ times, the escrow agent can reconstruct the needed key, extract p and unlock the device. This approach has the advantage that it can be implemented entirely using conventional block ciphers and hash functions that are built into existing secure processor hardware.[29]

To summarize, while adding some modest complexity, these extensions (or variants thereof) significantly minimize the useful attack surface on the device; even if a vulnerability is found allowing an outside party to probe the protected memory in the escrow agent or convince it to misbehave during its checking of the authorization key, these provide no advantage for recovering the passcode or unlocking the device absent additional secret information from the manufacturer.

Manufacturer protocol. Under this scheme, upon receiving a valid court order for a device with identifier d_{id}, the manufacturer provides the corresponding value d_{auth} (and any associated secrets protecting self-escrowed state, ${d_{seal}}^{-1}$ or d_{seed} as discussed above). We chose the manufacturer to manage the authorization key database for multiple reasons. First, as private enterprises, device manufacturers have the potential to act independently of the state and have demonstrated, at least in some situations, a willingness to contest government actions they deemed problematic to their customers [4, 36]. Second, we must already trust the manufacturer, not only for the security of their design and implementation, but also for the security of the post-sale software updates they provide. For example, should Apple choose—either for its own reasons or

[27] In our design each d_{id} is a unique random value. An alternative approach would be to encrypt each d_{id} using a block cipher keyed with a master secret. This approach simplifies key generation, but theft of the master secret would allow arbitrary generation of d_{id} values for any device whose identifier could be obtained, which we deemed an unnecessary risk.

[28] To guard against any future weaknesses discovered in H, it would be prudent to further salt the input with d_{id}, this ensuring that any attack on H must scale with the number of devices to be attacked.

[29] For those seeking a construction with crisper formal security guarantees, Bellare and Yee describe a more elaborate version using AES256 to drive a pseudo-random bit generator [25].

via government compulsion—it could create an iPhone update that would provide remote access to all stored data.

Some have argued that involving the manufacturer in the lawful access process will itself create new and undue risk for insider threat. The underlying contention is that, since lawful access requests will be more common than software updates, a proportionately greater number of the manufacturer's employees would necessarily be exposed to secret keys and that this would create undue opportunities for abuse [9, 26, 54]. We believe this argument is predicated on a highly simplistic insider threat model and what is likely a flawed comparison of the comparative risks of rogue software updates and the abuse of lawful access authorization keys.[30] Moreover, we observe that technology companies' legal process employees routinely handle far more sensitive data under warrants (e.g., the contents of our Gmail accounts, iCloud photos, Facebook posts, Twitter DMs, etc. provided under ECPA) and we are unaware of significant insider abuses in spite of this capability. However, putting aside these issues, the underlying abuse concern is thankfully one that can be substantially addressed through technical means.

The standard mechanism for storing cryptographic secrets in an enterprise environment is to use a Hardware Security Module (HSM), a strongly tamper-resistant device which effectively provides controlled, audited secret storage, frequently also combined with physical security. Indeed, Apple uses HSMs for securing iCloud escrowed keychain data [16]. HSMs are easily configured to require multiple employees to contemporaneously provide passwords and physical authentication credentials (e.g., via smartcards) before they will perform certain sensitive operations. Moreover, since the HSM is appropriately provisioned as an offline device (i.e., with no connectivity), those with physical access should be the principal vector for attack.[31] Thus, by storing the authorization key database in an HSM, any potential insider on the manufacturer's team handling legal process requests would face risks from the audit logs of such a device and might also need to recruit (or at least deceive) one or more co-conspirators as well. As another reasonable control the HSM could limit the number of *requests* that can be accepted per day). Since the identification values (d_{id}) should be selected randomly from a large domain (e.g., 2^{256}), even a quota well in excess of any reasonable law enforcement need (e.g., 1000/day) will not provide a useful mechanism for enumerating the authorization keys for unknown devices. Finally, it is possible to further encrypt the contents of the HSM in such a way that manufacturer employees never handle sensitive per-device values at all (e.g. d_{auth} and any protection keys).

Consider the following approach: a law enforcement agency (e.g., the FBI) provides a public key, LE_{FBI}, to the manufacturer who installs it in the HSM.[32] As before, a valid court order would include a statement including the d_{id} value obtained from the seized device, but would now also include a signature S covering that value, generated using the private key $LE_{FBI}{}^{-1}$. The manufacturer's employees responsible for legal process would validate the court order as they do today, then login to the HSM (using their HSM credentials together to authorize access) and provide the values for d_{id} and S. The HSM could then validate that S was a valid signature for d_{id} and, if so, lookup the corresponding tuple in the database (d_{id},d_{auth},...), encrypt them with LE_{FBI}, and return the now opaque value to the manufacturer's employees to be sent, in turn, to law enforcement.[33] In this way, legal process employees would be unable to substitute another value of d_{id}, nor be able to obtain an unencrypted version of d_{auth}. As a general matter, such an approach would be an improvement over the current state of legal process and would provide an additional tool for benign employees to distinguish valid court-orders from any potentially fraudulent requests.[34]

To summarize, we do not believe that insider threats via legal process employees are as significant as has been suggested; we are unaware of empirical evidence that current legal process requests are being subverted in this manner. Moreover, to the extent that risks of insider theft of authorization keys is a significant concern, there are meaningful technical approaches that would dramatically reduce such risk.

5.4 Transparency

The last property we turn to is transparency. Here the goal is to reveal that a device's passcode has been acquired via the lawful access procedure and minimize its value for covert use. On the device itself, the escrow agent can burn an "unlock attempted" status fuse upon initiating the authorization protocol (i.e., when the device first provides d_{id}) and another "unlocked" fuse once it completes the protocol and unlocks the device. The device firmware can report on the state of these fuses when the device starts (e.g., audible and visible notifications that the device has been unlocked by an outside party). If one needed to track *subsequent* events, one could implement a count.[35]

One shortcoming to this approach is that it assumes that the device itself will not be substituted (i.e., a variant of the "evil-maid" attack). Thus, one might unlock a phone via the authorization protocol, extract the contents of the phone and then copy the contents to a new phone (whose status fuses are still pristine). Should it be possible to complete such a transfer without detection, such a phone might be returned to the possession of its owner (a minimum of three days later) without them knowing that their device's contents had been accessed. Note that since the plaintext passcode is never revealed in our design, a potential covert actor has no way

[30]For example, any manufacturer employee authorized to update system-level code is in a position to compromise the integrity of the device's security (e.g., by inserting an obfuscated vulnerability that will appear in a future update) without any need for access to the code signing key. Moreover, the consequence of such an insider threat has far greater scope than losing an authorization key, since it potentially allows arbitrary, remote, covert access across large numbers of devices.

[31]Some have argued that "law enforcement's stated need for rapid access to data would make it impractical to store keys offline" [20], but this claim is at odds with the reality of lawful process—which is exceedingly slow in practice.

[32]Note, that the (infrequent) installation of an authorized law enforcement public key into the HSM *is* a sensitive operation and this aspect of the protocol would need to be performed with high assurance. In particular, full protection against rogue manufacturer employees requires that only well-authenticated law enforcement keys be configured in the HSM.

[33]Note that replay is not much of an issue in this context since each d_{id} is unique to a given device. However, should this be important—for example in the context of multiple warrants for the same device—a date could be added to the statement to be signed.

[34]A problem deserving considerably more attention from the technical community is the mechanisms by which court orders are authenticated in *existing* legal process.

[35]Note that some device manufacturers have indicated concern about this particular approach because it may undermine the resale value of the device for its owner.

to relock a substituted device such that it will only open with the appropriate passcode.[36]

Some manufacturers, such as Apple, already impose a per-device registration protocol (ultimately aggregated based on AppleID) that makes covert phone substitution challenging. In principal, a variant of such a protocol could also be used to inform the user that one of their devices had been opened. However, a straightforward implementation of such a protocol would require associating otherwise anonymous device identifiers (d_{id} values) with well-known identifiers (e.g., IMEI/MEID) and presupposes that all manufacturers have a direct relationship with their customers (while this is true for Apple, is decidedly not true for a range of other device manufacturers). Instead, an alternative approach would be for the manufacturer to simply publish the set of H(d_{auth}) that have been disclosed and each device (or a proxy) could periodically check to find out if their secret was among those that had been disclosed. This would require that $H(d_{auth})$ be readable on the device, or at least could be matched, but it should have little effect on the security of the authorization protocol due to the one-way nature of H. Obviously, an actor able to produce a device with modified code or able to prevent the manufacturer from communicating their disclosures is in a position to block the transparency property.[37]

5.5 Manufacturer costs

A concern commonly raised is that the costs of any such scheme may be excessive and place undue burden on manufacturers. There are undoubtedly costs to be borne in this mechanism—both up front investments and ongoing operational costs.

Perhaps the most significant upfront costs arise from the design and implementation of the lawful access technical capability in each device as well as the testing and quality assurance around the resulting hardware and software. These are real costs but seem likely modest in comparison to the extensive and high-value investments being made in improving device security (e.g., for example, most of the complex work in this proposal is already an included and necessary component of the design in Apple's Secure Enclave). There are also likely upfront costs for secure storage, authentication mechanisms and investments in operational security, but these again seem likely to be quite small relative to manufacturer's existing information technology investments (especially given that much of what is needed here already exists to support existing legal process requests).

Indeed, more emphasis has been placed on the potential ongoing operational costs (e.g., "the cost and complexity of authenticating a high volume of global law enforcement requests—not just accurately, but fast enough to meet the quick turn-around time needed in urgent investigations or possibly mandated by the applicable key-recovery law—will fall entirely on the vendor" [50]). However, the evidence is unclear if these costs are indeed large compared to the size of these enterprises or if they would dramatically increase the costs currently borne by manufacturers in support of existing criminal process requests.

To explore this question further, consider Apple's most recent transparency report: there were 4,479 device requests received from U.S. law enforcement agencies in the first half of 2017 as well as requests to access 6,407 accounts (i.e., iCloud)—or, in total, requests to access roughly 20,000 devices or accounts per year [7]. By comparison, in one recent speech the Director of the FBI claimed that his office had been unable to access the content on 7,775 devices in 2017 (across all manufacturers—Apple is roughly 30–40% of smartphones in the U.S.)[63]. Subsequent reporting has suggested that the true number of such devices is likely only 1,000-2,000 [22]. While there is little doubt that state and local law enforcement agencies would also make similar requests, it is not at all clear that their level of demand would exceed that of the FBI. We are unaware of good data on this topic (as most transparency reports do not break out requests from federal vs non-federal sources), but some insight may be gleaned from Twitter, which breaks out Federal, State and Local requests in their transparency reports. In their July 2017-December 2017 report, Twitter documents 1,761 requests for account information from U.S. sources, of which roughly two thirds were Federal [8]. Thus, while clearly creating a new criminal process demand on manufacturers would create additional operational burden, we see no evidence that this burden would be qualitatively distinct from their existing criminal process workload.

Moreover, we find the claim of high costs needed to meet the "quick turn-around time needed in urgent investigations" unconvincing. In practice, court-ordered production of digital evidence for criminal cases routinely takes several weeks if not longer (e.g., Linkedin's guidelines instructs law enforcement to "Please allow at least 3 weeks for processing" [18], Facebook's guidelines offer "2–6 weeks" for the same [3] and Ebay says requests "are proessed in approximately 20 business days" [5]). Moreover, while most heavily-served tech companies offer a mechanism for emergency processing if there are exigent circumstances (e.g., Apple provides the exigent@apple.com contact for precisely this reason), these seem to be a small minority of requests. In Apple's most recent transparency report, they received 118 such emergency requests in the first half of 2017 from U.S. law enforcement sources which is a very small fraction of total requests received.

That is not to say that such operational costs are insignificant, nor that it is a foregone conclusion that device manufacturers should bear them. Indeed, current electronic searches under the Electronic Communications Privacy Act (ECPA) already mandate cost recovery to the provider for their efforts and a similar arrangement for handling authorization key requests would seem entirely reasonable. To summarize, we do not see strong evidence that costs would be so burdensome as to make the approach we have described untenable in practice.

6 CONCLUSIONS

In this paper we have described a systems-oriented design for a lawful "device unlock" capability. We believe that this design effectively forecloses its use for mass surveillance purposes and, further, that it offers stronger safeguards and greater transparency than the "lawful hacking"-based approach that forms the core of today's

[36]However, this depends on the owner of this device noticing this discrepancy, which they may not, and a sophisticated attacker might install a false login screen on the replacement to address this discovery risk.

[37]We can envision protocols involving third-parties in separate jurisdictions that might offer stronger transparency guarantees, but these seem too complex, both operationally and politically, to be realistic.

status quo. However, we have little doubt that the technical shortcomings of our design will be swiftly identified and that there will be further refinements or yet other alternative strategies that will be superior still. Indeed, we write this paper precisely in the hope that there is sufficient additional research activity to obsolete our own work. Most of all, we hope to help dissuade others of the notion that such research "should not be done", a sentiment that we think is inherently unhealthy for the field. We contend that even the most steadfast opponents of *any* lawful access capability are well-served by a healthy set of concrete options to push against.

ACKNOWLEDGEMENTS

In doing this work, I've benefited from considerable constructive feedback from a broad community. Thanks in particular to Herb Lin for encouraging me to work on this problem, David Clark and Danny Weitzner for inviting me to participate in their events, Ernie Brickell and Ray Ozzie who, for different reasons, ended up in similar places and were generous in sharing their ideas and criticisms. In addition, I've benefited from the time and creativity of a broad range of researchers who contributed key ideas or provided constructive feedback on drafts of this paper, foremost being Mihir Bellare, Nick Weaver and Alex Gantman, as well as David Kohlbrenner, Hovav Shacham, Philip Rogaway, Aaron Schulman, Geoff Voelker, Ryan Kastner and a host of others. Finally, I'd like to thank the anonymous ACM CCS referees, as well as my shepherd from the same conference, for the candor of their feedback and their open-mindedness in appreciating the value of design-oriented papers.

REFERENCES

[1] FBI 'Going Dark' FOIA Documents – Release 1, Part 1. https://www.eff.org/document/fbi-going-dark-foia-documents-release-1-part-1.

[2] Declaration of Mark Klein in Support of Plaintiff's Motion for Preliminary Injunction. https://www.eff.org/files/filenode/att/mark_klein_unredacted_decl-including_exhibits.pdf, June 2006.

[3] Facebook Law Enforcement Guidelines. https://www.eff.org/files/filenode/social_network/facebook2010_sn_leg-doj.pdf, 2010.

[4] Apple Inc's motion to vacate order compelling Apple Inc. to assist agents in search, and opposition to government's motion to compel assistance. https://epic.org/amicus/crypto/apple/In-re-Apple-Motion-to-Vacate.pdf, Mar. 2016.

[5] Ebay Global Asset Protection: Resources for Law Enforcement Investigations and Service Levels. https://ir.ebaystatic.com/pictures/aw/pics/pdf/us/eBayLEGuideSecurityCenter2016.pdf, 2016.

[6] In the matter of the search of an Apple iPhone seized during the execution of a search warrant on a black Lexus IS300, California License Plate 35KGD203. https://www.justice.gov/usao-cdca/file/825001/download, Feb. 2016.

[7] Report on Government and Private Party Requests for Customer Information: January 1—June 30, 2017. https://images.apple.com/legal/privacy/transparency/requests-2017-H1-en.pdf, 2017.

[8] Twitter Transparency Report: United States of America: July to Deceember 2017. https://transparency.twitter.com/en/countries/us.html, 2017.

[9] Bring in the Nerds: EFF Introduces Actual Encryption Experts to the U.S. Senate Staff. https://ssd.eff.org/en/blog/bring-nerds-eff-introduces-actual-encryption-experts-us-senate-staff, May 2018.

[10] Cellebrite Advanced Unlocking Services. https://www.cellebrite.com/en/services/advanced-unlocking-services/, 2018.

[11] CPU frequency: Hall of Fame. https://hwbot.org/benchmark/cpu_frequency/halloffame, May 2018.

[12] Decrypting the Encryption Debate: A Framework for Decision Makers. National Academies Press, Feb. 2018.

[13] Encryption in the U.S.: Crypto Colloquium Outcomes Report. https://www.accessnow.org/cms/assets/uploads/2018/02/Encryption-in-the-United-States-Crypto-Colloquium-Outcomes-Report.pdf, Jan. 2018.

[14] Encryption Policy in Demogratic Regimes: Finding Convergent Paths and Balanced Solutions. https://www.eastwest.ngo/sites/default/files/ewi-encryption.pdf, Feb. 2018.

[15] Introducing GreyKey. https://graykey.grayshift.com/, 2018.

[16] iOS Security. https://www.apple.com/business/docs/iOS_Security_Guide.pdf, 2018.

[17] Legal Process Guidelines. https://www.apple.com/legal/privacy/law-enforcement-guidelines-us.pdf, 2018.

[18] Linkedin Law Enforcement Data Request Guidelines. https://help.linkedin.com/ci/fattach/get/7890851/0/filename/Law_Enforcement_Guidelines_January%202018.pdf, 2018.

[19] Policy Approaches to the Encryption Debate. https://2o9ub0417chl2lg6m43em6psi2i-wpengine.netdna-ssl.com/wp-content/uploads/2018/04/133-1-1.pdf, Mar. 2018.

[20] H. Abelson, R. Anderson, S. M. Bellovin, J. Benaloh, M. Blaze, W. Diffie, J. Gilmore, M. Green, S. Landau, P. G. Neumann, R. L. Rivest, J. I. Schiller, B. Schneier, M. A. Specter, and D. J. Weitzner. Keys under doormats: mandating insecurity by requiring government access to all data and communications. *Journal of Cybersecurity*, 1(1):69–79, 2015.

[21] Andrey Belenko. iOS Forensics with Open-Source Tools. https://www.blackhat.com/docs/sp-14/materials/arsenal/sp-14-Belenko-IOS-Forensics-Slides.pdf, 2014.

[22] D. Barrett. FBI repeated overstated encryption threat figures to Congress, public. *Washington Post*, May 2018.

[23] M. Bellare and S. Goldwasser. Encpasulated key escrow. Technical Report 688, MIT Laboratory for Computer Science, Apr. 1996.

[24] M. Bellare and S. Goldwasser. Verifiable Partial Key Escrow. In *ACM Conference on Computer and Communications Security*, 1997.

[25] M. Bellare and B. Yee. Forward-Security in Private-Key Cryptography. In *Topics in Cryptology — CT-RSA 2003*, 2003.

[26] S. M. Bellovin, M. Blaze, D. Boneh, S. Landau, and R. Rivest. Op-ed: Ray Ozzie's crypto proposal—a dose of technical reality. https://arstechnica.com/information-technology/2018/05/op-ed-ray-ozzies-crypto-proposal-a-dose-of-technical-reality/, May 2018.

[27] S. M. Bellovin, M. Blaze, S. Clark, and S. Landau. Lawful Hacking: Using Existing Vulnerabilities for Wiretapping on the Internet. *Northwestern Journal of Technology and Intellectual Propertuy*, 12(1):1–66, 2014.

[28] M. Blaze. Oblivious Key Escrow. *Lecture Notes in Computer Science*, 1174:335–343, 1996.

[29] C. A. Boyd, X. Boyen, C. Carr, and T. Haines. Key recovery: Inert and Public. *Lecture Notes in Computer Science*, 10311, 2017.

[30] L. Cauley. NSA has massive database of Americans' phone calls. *USA Today*, May 2006.

[31] D. Chaum, D. Das, F. Javani, A. Kate, A. Krasnova, J. D. Ruiter, and A. T. Sherman. cMix: Mixing with Minimal Real-Time Asymmetric Cryptographic Operations. IACR Cryptology ePrint Archive, 2016:008, 2016.

[32] S. Checkoway, J. Maskiewicz, C. Garman, J. Fried, S. Cohney, M. Green, N. Heninger, R.-P. Weinmann, E. Rescorla, and H. Shacham. A systematic analysis of the Juniper Dual EC incident. In *Proceedings of the ACM Conference on Computer and Communications Security*, 2016.

[33] F. Courbon, S. Skorobogatov, and C. Woods. Reverse Engineering Flash EEPROM Memories Using Scanning Electron Microscopy. In *Proceedings of the International Conference on Smart Card Research and Advanced Applications (CARDIS)*, pages 57–72. Springer, 2017.

[34] EFF. What's the likely global impact if the FCC approves the tentative rules set forth in the NPRM. https://www.eff.org/pages/calea-faq#19, 2004.

[35] Ernest Worthman. Why Every Chip Can Be Hacked With This Tool. *Semiconductor Engineering*, Aug. 2014.

[36] S. Ghosh. Apple, Amazon and Microsoft and helping Google fight an order to hand over foreign emails. *Business Insider*, Mar. 2017.

[37] M. Green. A few thoughts on Ray Ozzie's 'CLEAR' Proposal. https://blog.cryptographyengineering.com/2018/04/26/a-few-thoughts-on-ray-ozzies-clear-proposal/, Apr. 2018.

[38] A. Greenberg. The father of online anonymity has a plan to end the crypto war. *Wired*, Jan. 2016.

[39] D. James, D. Yang, S. Wegner, C. Davis, and A. Cowsky. Inside the iPhone 6S. http://www.techinsights.com/about-techinsights/overview/blog/inside-the-iphone-6s/, 2018.

[40] J. Kelsey. Dual EC in X9.82 and SP 800-90. https://csrc.nist.gov/csrc/media/projects/crypto-standards-development-process/documents/dualec_in_x982_and_sp800-90.pdf.

[41] J. Kong, F. Koushanfar, P. K. Pendyala, A.-R. Sadeghi, and C. Wachsmann. Pufatt: Embedded platform attestation based on novel processor-based pufs. In *Proceedings of the Design Automation Conference*, New York, NY, USA, 2014.

[42] S. Levy. Cracking the crypto war. *Wired*, Apr. 2018.

[43] Y. Li, K. Sakiyama, S. Gomisawa, T. Fukunaga, J. Takahashi, and K. Ohta. Fault sensitivity analysis. In *Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems*, Berlin, Heidelberg, 2010.

[44] V. Lomne, T. Roche, and A. Thillard. On the Need of Randomness in Fault Attack Countermeasures—Application to AES. In *Proceedings of the Workshop on Fault Diagnosis and toleraing in Cryptography*, Sept. 2012.

[45] J. Markoff and S. Shane. Documents Show Link Between AT&T and Agency in Eavesdropping Case. *New York Times*, Apr. 2006.

[46] P. Mishra, S. Bhunia, and M. Tehranipoor. *Hardware IP Security and Trust*. Springer Publishing Company, Incorporated, 1st edition, 2017.

[47] H. Nguyen. Lawful hacking: towards a middle-ground solution to the going dark problem. Master's thesis, Naval Postgraduate School, Monterey, California, Mar. 2017.

[48] R. Ozzie. CLEAR. https://github.com/rayozzie/clear/blob/master/clear-rozzie.pdf, Jan. 2017.

[49] S. Patranabis, A. Chakraborty, D. Mukhopadhyay, and P. Chakrabarti. Using State Space Encoding To Counter Biased Fault Attacks on AES Countermeasures. IACR Cryptology ePrint Archive, 2015:806, 2015.

[50] R. Pfefferkorn. The Risks of 'Responsible Encryption'. https://cyberlaw.stanford.edu/files/publication/files/2018-02-05%20Technical%20Response%20to%20Rosenstein-Wray%20FINAL.pdf, Feb. 20185.

[51] T. Reed. GreyKey iPhone unlocker poses serious security concerns. https://blog.malwarebytes.com/security-world/2018/03/graykey-iphone-unlocker-poses-serious-security-concerns/, Mar. 2018.

[52] J. Risen and E. Lichtblau. Bush Lets U.S. Spy on Callers without Courts. *New York Times*, Dec. 2005.

[53] P. Rogaway. The moral character of cryptographic work. IACR Cryptology ePrint Archive, 2015:1162, 2015.

[54] C. Savage. Justice Dept. Revives Push to Mandate a Way to Unlock Phones. *New York Times*, Mar. 2018.

[55] M. Scanlon. Battling the Digital Forensic Backlog through Data Deduplication. In *Proceedings of the IEEE International Confernece on Innovative Computing Technology*, Aug. 2016.

[56] M. Seaborn and T. Dullien. Exploiting the DRAM rowhammer bug to gain kernel privileges. https://googleprojectzero.blogspot.com/2015/03/exploiting-dram-rowhammer-bug-to-gain.html, Mar. 2015.

[57] R. Staff. FBI paid under 1 million to unlock San Bernardino iPhone: sources. *Reuters*, Apr. 2016.

[58] Tarjei Mandt and Mathew Solnik and David Wang. Demystifying the Secure Enclave Processor. https://www.blackhat.com/docs/us-16/materials/us-16-Mandt-Demystifying-The-Secure-Enclave-Processor.pdf, 2016.

[59] Thomas Fox-Brewster. It Might Cost the FBI Just $1,500 To Get Into Terrorist's iPhone. *Forbes*, Mar. 2016.

[60] R. Torrance and D. James. The State-of-the-Art in IC Reverse Engineering. In *Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems(CHES)*, pages 363–381, 2009.

[61] E. Tromer. Eran Tromer's Attack on Ray Ozzie's CLEAR Protocol. https://www.cs.columbia.edu/~smb/blog/2018-05/2018-05-02.html, May 2018.

[62] J. Williams. The Unofficial Guide to Facebook's Law Enforcement Portal Version 2. https://cdn.netzpolitik.org/wp-upload/2016/08/facebook-law-enforcement-portal-inofficial-manual.pdf, 2013.

[63] C. Wray. Raising Our Game: Cyber Security in an Age of Digital Transformation. https://www.fbi.gov/news/speeches/raising-our-game-cyber-security-in-an-age-of-digital-transformation, 2018.

[64] C. Wright and M. Varia. Crypto Crumple Zones: Enabling limited Access without Mass Surveillance. In *Proceedings of the IEEE European Symposium on Security and Privacy*, 2018.

Appendix A COMMON QUESTIONS

Because of the long history of conflict over lawful access, there are a set of standard questions that invariably arise. While these questions have been addressed implicitly in the paper, we believe there is value in calling them out explicitly here to avoid potential misunderstandings.

- *What prevents a criminal from simply using a third-party encryption product above the operating system? or purchasing a phone abroad in a jurisdiction that doesn't support lawful access?*

The answer is that nothing prevents such behavior. We do not believe there is an implementable policy option that would provide a universal decryption capability.[38] We also believe that this is a completely acceptable outcome. The reality is that most criminals will not use such third-party products (or if they do, will not use them in a secure fashion, with device-independent credentials), just as most criminals do not encrypt their telephone calls or their instant messages. It is not a new phenomenon that criminals with stronger operational security practices will require greater effort to investigate.

[38]Even the wildly overreaching Feinstein/Burr "Court Orders Act", which mandates plain-text access capability for all software, services and products sold, would not achieve this goal if implemented.

- *What about in an emergency? Surely a device could be unlocked if a victim's life depended on it?*

The nature of the system we've described is that it makes tradeoffs that are inherently compromises. The purpose of the lockout period is to undermine the use of this mechanism for covert purposes and an inherent side effect of that design choice is that there is no mechanism for "expedient access". The choice of the lockout period duration implicitly balances the likelihood and costs of one scenario against the likelihood and costs of the other.

- *What if the manufacturer gets hacked by a state actor?*

It has become fashionable to observe that, with the plethora of high-profile state-sponsored data-breaches (e.g., OPM), virtually all organizations are susceptible. In some ways this may be true, but it is worth reflecting on the consequence of this truth. Thus, if we are confident that state actors will so thoroughly penetrate a manufacturer as to compromise an offline HSM containing authorization keys and then jump the air gap again to exfiltrate those keys, how much trust can we place in the security of the significantly more valuable assets that the manufacturer must already protect? For example, capturing the signing keys for device update would allow the creation of new code updates for any of the manufacturer's devices—providing arbitrary remote access at arbitrary scale. Similarly, any developer account with commit rights to system level code is in a position to insert obfuscated vulnerabilities to be released in later software versions and, as with updates, provide large-scale remote covert capabilities.[39] Moreover, leading device manufacturers (e.g., Apple, Google) frequently have access to much of their customer's content online (e.g., via standard backup services, and via cloud services such as photos on Apple iCloud and e-mail Google's Gmail) and these are available to key employees and systems. All of these would appear to be easier and/or more attractive targets for state actors. This is not to imply that manufacturers are not indeed vulnerable to such attacks, but that we should not hold a significantly weaker lawful access capability to a higher standard of state actor-resistance than more valuable existing capabilities.

- *Don't all lawful access schemes inherently weaken the security of the device?*

Part of the answer to this question is philosophical. Security is, among other things, about limiting data access to authorized parties. If we do not believe that law enforcement agents operating under color of law are authorized, then yes, by definition, any lawful access scheme weakens security. However, this is a definition that presupposes an answer and is an inherently non-technical argument. If instead we accept the assumptions of this paper (i.e., that there is a communal interest in considering some such access authorized) then the answer to this question is more nuanced. Now the question is, what is the residual risk that the mechanism provided for lawful access will be misused in unlawful ways? We do not

[39]While the precise timeline and authorship of the several code and data implants in the Juniper NetScreen case may never be publicly known, even the most conservative account of the evidence supports a finding that malicious code commits were commercially distributed for well over a year without discovery [32].

believe the answer to this question is clear and we have yet to identify how the approach described in this paper offers significantly greater residual risks than those we already face today.

- *Isn't this scheme vulnerable to a variant of Tromer's attack on CLEAR?*

As described, a qualified yes. Eran Tromer's attack on Ozzie's CLEAR proposal is a trojan man-in-the-middle attack in which an adversary attacks a phone x, by creating an imitation phone x' that purports to be x (e.g., by announcing the d_{id} the attacker obtains from x) and then plants the imitation phone (e.g., at a crime scene) to entice law enforcement to unlock the phone using their lawful access protocol. The imitation phone then relays all authorization protocol messages it receives from law enforcement to the attacker who is then able to replay them to the targeted phone to unlock it (the attack is described in more detail here [61]). This attack exploits the implicit trust placed in the authorization protocol messages produced by a device under interrogation by law enforcement and we have made no efforts to address it (i.e., in our current design there is no attempt to authenticate that a device is who it says it is).

However, the attack places unusually demanding requirements on the adversary: that they must obtain extended physical access to the target device (i.e., to obtain its d_{id} value), then must also create a replica device with internal hardware that will spoof the authorization protocol, then must convince law enforcement to seize that device and obtain legal process for opening it. Most critically, they must devise a mechanism such that their replica device can transmit the messages it receives while in law enforcement custody (seized devices are routinely stored and accessed in RF shielded environments precisely because law enforcement is concerned about outside tampering with a device). This is not to say that such a scenario is *impossible.* However, in our estimation it is at the far end of the threat spectrum and we feel that the value in addressing this particular threat is far outweighed by the value of simplicity in the current design. That said, we are confident that should others disagree with this assessment, there are a range of technical solutions that could be employed (e.g., time-bounding to prevent real-time relaying and explicit device authentication via a shared secret before providing any authorization material) to address this concern.