Considus

TECHNICAL WHITE PAPER

Catchlight for iPhone

Who might read your notes

An adversary-by-adversary threat model: what a thief, a household member, a cloud provider, Apple, a warrant and the maker can each actually do.

Applies to Catchlight 1.0

Introduction

There's a line in Catchlight that takes up to 100 characters of your note and puts them on your lock screen.

It's not a bug and I didn't find it by accident, it's how reminders work. When one fires, the notification has to say what it's reminding you about, and the thing it's reminding you about is your writing. The character limit is there deliberately to cap how much gets out. But it does mean that an app I've spent a year making unreadable to me, to Apple, to Dropbox and to anyone with a warrant will happily display a line of your writing to whoever happens to be looking at your phone on the kitchen table.

Whether that matters depends entirely on who you're worried about. And that's the question almost nobody asks, including most of the people selling you the answer.

People fit good alarms and leave a key under the flowerpot. Not because they're stupid, but because the alarm is the bit you can buy and the flowerpot is the bit you have to think about. Privacy software is full of very good alarms.

Here's the list of everyone who could conceivably read your notes, and what each of them can really do. Some of them can do more than this industry usually likes to admit. One of them is me.

Three things in here aren't in the architecture paper, and each one is a real exposure rather than a theoretical one. The lock screen, above. The fact that your notes are deliberately kept out of your iPhone backup, which protects you and also takes away a safety net. And the single network request the app makes, which happens under exactly one condition.

Ask who, not whether

Nearly all privacy advice starts with a product. It ought to start with a person.

Write down who you don't want reading your notes. Actually write it, it takes a minute. The answer is completely different for someone worried about losing their phone in a taxi, someone in a relationship that's gone wrong, a journalist protecting a source, and someone who just doesn't fancy an advertising company reading their shopping list. Those are four different problems and three of them need more than any notes app can give them.

That last point is one I'd rather make now than bury. If your physical safety depends on this, an app store is the wrong place to be shopping and I'm the wrong person to be advising you. Everything below assumes an ordinary person with ordinary opponents.

Why I've built the model this way round

There are two established ways to do this and I've used neither.

The STRIDE family works outward from the system. You draw the components, then ask what could go wrong at each one, spoofing, tampering, information disclosure and so on. LINDDUN does the privacy version of the same move, seven named categories worked over a data-flow diagram. Both are good, both are what I'd reach for if I were auditing somebody's code, and both produce a document written for engineers.

The trouble is that you're not auditing my code. You're deciding whether to trust an app with your writing, and that decision has a different shape. It isn't "which components are weak", it's "who can get at this, and what happens if they do". So the model is organised by adversary, because that's the question actually being asked.

Somebody holding your phone

This is the one nearly everybody actually meets.

If it's unlocked, they see what you see. All of it. Encryption protects data sitting still and data in transit, it does nothing for a screen that's already lit up.

Catchlight adds one more door. The app locks on its own, so somebody holding your unlocked phone still hits a PIN or a Face ID check. One thing sits outside that door on purpose. The capture screen opens without the check, because getting a thought down in two taps is most of the point, so somebody already past your device passcode can write a new Take. It opens blank and saving takes a Face ID check, so they can add to your Takes and not read them. That is the entire defence at this level.

If it's locked and they don't know the passcode, they get nothing. The database is protected until first unlock, and the master key sits in the Keychain wrapped by the Secure Enclave, marked to this device only. A locked iPhone is a hard target, and most of that work is Apple's.

One thing still gets out, and this is the flowerpot. That notification preview. If your iOS settings show previews, a reminder will put up to 100 characters of your words on the lock screen. Show Previews, set to When Unlocked or Never, fixes it, and it covers every app you own, not just mine. The app does not suppress it and does not try to, so that setting is yours rather than mine.

And then there's the person who lives with you and knows your passcode.

The adversary the privacy industry names least, and an awful lot of people actually have. Your passcode opens the device, the device opens the app, and zero-knowledge encryption protects you from a company rather than from someone on your sofa. They're not picking the lock. They live here.

The per-note lock in v1.3 changes that, and it isn't built, so Catchlight v1.0 does not protect one person in a household from another. Any app telling you otherwise is selling you a flowerpot.

The companies in the chain

Three organisations touch this in some form, and they can do very different amounts. Figure 1 draws the boundary, including the one place a Take deliberately crosses it.

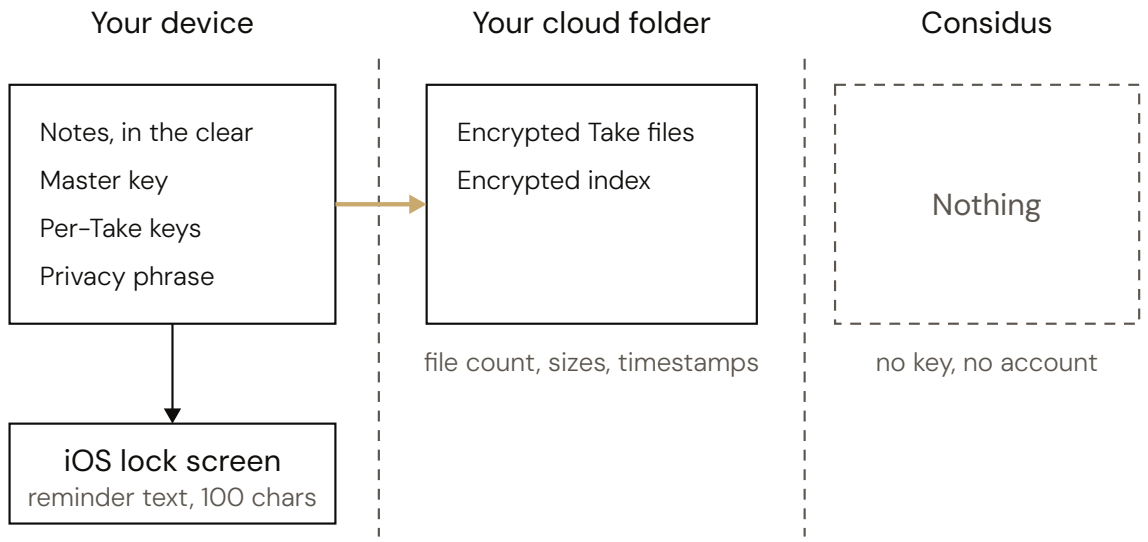

Figure 1. What crosses the boundary. Encryption happens before anything is written, so the provider receives ciphertext and an encrypted index. What it can still see is the shape of the files, never their contents. The one deliberate exception: a reminder puts up to 100 characters of a Take on the lock screen. iOS notification previews can be set to appear only when unlocked.

Your cloud provider holds your encrypted files, because you picked the folder. They can see how many files there are, how big each one is, and their own timestamps on them, which is enough to know you write most nights or that you wrote a great deal on one particular Sunday. They cannot see a single word. The index is encrypted as well, so the folder won't even tell them how many notes you keep or what you've deleted.

Apple is in a stronger position than anybody else here, because Apple runs the operating system. What Apple gets is a record that you bought the app, the same as for every paid iOS app ever sold. If you chose iCloud Drive as your folder, Apple is also your cloud provider and the paragraph above applies to them too.

What Apple doesn't get is your notes in a device backup. The database and its sidecar files are explicitly marked as excluded, and the master key is marked device-only, so neither of them travels in an iPhone backup. Your note text is never indexed by iOS search or Siri either, because that setting is off by default and the levels that would expose the text are locked out of the interface entirely.

That backup exclusion is a trade and I'd rather you heard it from me. Your iPhone backup does not contain your notes. Restore a backup and your notes don't come back. The encrypted folder plus your Privacy phrase is the recovery path and it's the only one there is.

Then me. I hold no notes, no key, no phrase, no email address and no idea where your folder is. An order served on Considus produces nothing, because there is nothing sitting anywhere to produce.

Which is the claim you should be testing hardest, so here's the important part of it. That isn't a promise about how I'll behave, because a promise is worth very little and can be revised by whoever owns the company next. It's a description of what the architecture leaves possible. If I sell up, or give up, or simply change my mind one Tuesday, your notes are still in your folder, encrypted under a key only you hold. The format is set out in full in the architecture paper, so somebody could write a reader for it, and the app's own Markdown export works whether or not you are still paying. A new owner inherits no ability to read them, because there's nothing to inherit.

When somebody can compel a person

Encryption fails differently when the attack is on you rather than on the maths.

A warrant gets four different answers depending on who it's served on, and Figure 2 has them side by side.

SERVED ON	WHAT IT YIELDS
Considus	Nothing. No account, no key, no data
Your cloud provider	Ciphertext, and the shape of the files file count, sizes, provider timestamps
Apple	The App Store purchase record plus iCloud Drive contents, which are ciphertext
You	Everything, the moment you unlock

Figure 2. Four answers to one warrant. The architecture handles the first three rows and cannot help with the fourth. Somebody with a warrant does not pick the lock, they knock on the front door and ask you to open it.

That last row is the one that matters and it's the one no architecture anywhere can do anything about. Somebody with a warrant doesn't pick the lock. They knock on the front door and ask you to open it.

Border officers in a lot of countries can require a device to be unlocked, and an employer who manages your phone has powers over it that no app overrides. Device management can install profiles, read data and take screenshots, and there is nothing I can build that stops any of it.

One practical thing rather than legal advice, because the law here differs by country and keeps moving. A face or a finger can be used on you while you sit still. A passcode has to be remembered and typed. If that distinction matters to you, iOS will drop back to passcode-only in a couple of seconds, hold the side button and a volume button until the power-off screen appears.

The network

Catchlight makes one network request. Once, ever, and only when you share a web link into it, at which point it fetches that link's title and preview image so the thing you saved looks like the page rather than a naked URL. It's in the privacy policy.

That's the lot. No analytics, no third-party crash reporting, no advertising identifier, and no server of mine for the app to talk to even if it wanted to. Sync is a file operation inside your own folder, not a call home.

What I actually think is going to happen to you

Ranked by how likely it is, which is not the order this industry usually puts them in.

You'll lose your Privacy phrase and your device. By a distance the most likely way to lose your notes, and it isn't an attack at all, it's a Tuesday.

Somebody will pick up your unlocked phone. Ordinary, common, and the thing all this encryption helps with least.

Somebody will read a notification off your lock screen. Small, real, and fixed by an iOS setting in less time than it took you to read this sentence.

Somebody who knows your passcode will go looking. Rare across everybody, absolutely severe for the people it applies to, and not solved until v1.3.

Somebody will serve a legal order. Handled well by the architecture, and far less likely than any of the four above it.

Somebody will break the cryptography. Least likely thing on this page, most discussed thing in the category.

That list is upside down compared with how privacy software is sold, and writing it down that way round is most of why this paper exists.

So what should you actually do

If losing everything is the worry, write your Privacy phrase on paper and keep it away from the phone. That single act is worth more than everything else on this list combined.

If it's a lost or stolen phone, use six digits or more for your passcode, and turn Catchlight's app lock on.

If it's the lock screen, go and change the notification preview setting now, before you forget.

If it's somebody in your house, understand that v1.0 doesn't solve it. Wait for the per-note lock, or keep the things that would really hurt out of any notes app, mine included.

If it's a work phone, don't. If it's a border crossing, turn the thing fully off before you get there, so it wants your passcode rather than your face.

The decisions behind those answers

Four of the things above were choices rather than accidents, and a threat model that lists exposures without saying why they exist is only doing half the job.

Why the notification carries your words at all. A reminder that says "Catchlight reminder" is useless, you'd have to open the app to find out what it wanted, which rather defeats the point of a reminder. It carries the outstanding text instead. The 100-character cap is there to bound how much crosses out of the app rather than to make the banner tidy, and the reminder deliberately reads out only the tasks still outstanding rather than the whole note. It's a genuine trade between a useful reminder and a quiet one, and I came down on useful, with the iOS preview setting as the escape hatch for anyone who'd rather have quiet.

Why your notes are kept out of your iPhone backup. A device backup is a copy of your data in a place governed by different rules from the ones this app lives under, and for a great many people that place is iCloud. Putting encrypted notes and their key material into it would have quietly undone the thing the architecture exists to do. Which is why the database, its three sidecar files and the master key are all excluded. The cost is real and I've said it twice already, restoring a backup restores no notes, and your folder plus your phrase is the only way home.

Why Lock and Hide are two features and not one. Both are coming in v1.3 and they look like one switch, so the split is worth explaining. Lock adds a re-authentication step and **keeps syncing**, because every note already leaves your device encrypted under a key I have never held, so withholding it from your folder would buy you no privacy at all and would cost you your only backup. Hide is the one that excludes a note from sync, and it carries a caveat that must never be dropped, which is that a hidden note has no backup anywhere and dies with the phone. Building them as one control would have buried that difference under a single toggle.

Why Siri and Spotlight see nothing by default. Letting the OS index your note text makes search better and hands your words to a system I don't control. The setting exists, it defaults to off, and the levels that would expose the body of a note are locked out of the interface entirely rather than merely defaulted off. That's a deliberate choice to make the dangerous option unreachable instead of available and discouraged.

Where this sits against the formal frameworks

I'm not claiming a certification, an audit, or a completed formal analysis. What follows is an honest map, so somebody who works in this field can see what I've covered and what I haven't.

LINDDUN is the privacy threat taxonomy from KU Leuven, and its seven categories are the closest formal thing to what this document does by hand.

LINDDUN CATEGORY	WHERE CATCHLIGHT STANDS
Linking	No account, no email, no identifier of any kind held by me, so there is nothing on my side to link records against
Identifying	The same answer. I hold no identity to attach data to
Non-repudiation	Largely not applicable. Single-user local app, with no claims being made to a third party
Detecting	This is the residual risk, and the framework names it better than I did. Your cloud provider can detect that you wrote, roughly how much, and when, from file counts, sizes and timestamps, without reading a word
Data disclosure	Content is sealed per note. The known disclosure is the notification preview, above
Unawareness	The category this paper and its companions exist to address. The backup-exclusion consequence was, until very recently, a real failure here
Non-compliance	No personal data is collected, so most obligations have nothing to attach to. What remains is the complaint and reporting route, published separately

Detectability is the weak spot. It's the one category where the architecture cannot give you a clean answer, and it isn't fixable by better encryption. Closing it would need traffic padding, decoy files and fixed-size writes, which is a real technique with a real cost, and it is not something Catchlight does today.

OWASP MASVS is at version 2.1.0 with eight categories, and four of them bear on this document. MASVS-STORAGE covers on-device storage and specifically whether data leaks into logs, caches or backups. MASVS-PLATFORM covers

interaction with the operating system, which is where notification previews and system indexing live. MASVS-PRIVACY was added to the standard in January 2024 and covers much the same ground as LINDDUN. MASVS-CRYPTO is handled in the architecture paper rather than here.

To be exact about the claim, the design was built and reviewed against this guidance by me. No independent MASVS verification has been carried out at any level, and nothing here asserts one.

On the risk ranking. The order in the section above is mine, arrived at by weighing how likely each thing is against how bad it would be, which is the shape NIST SP 800-30 sets out for risk assessment. I have not run a formal scored assessment, so the ranking is a judgement rather than a calculation, and I would rather show you the judgement than dress it up as arithmetic.

What isn't in here

The cryptographic design is in paper 1 and I haven't repeated it. A compromised operating system beats everything described above and I make no claim against it. Targeted attacks by a serious well-funded adversary are out of scope, as I said at the top. And your cloud provider's threat model is theirs to publish, not mine to summarise on their behalf.

I've spent a year on the alarm. This is the paper about the flowerpot, and I'd rather write it myself than have someone else find it.

References

Frameworks and guidance

LINDDUN privacy threat modelling, KU Leuven

OWASP Mobile Application Security Verification Standard v2.1.0

OWASP Mobile Application Security Testing Guide

NIST SP 800-30, Guide for Conducting Risk Assessments

Apple Platform Security

EFF Surveillance Self-Defense

Companion documents

The Catchlight encryption architecture, for the cryptographic design this paper assumes

How to tell whether a notes app is actually private, for the seven tests that turn this model into something you can check yourself

What AI features cost you in a notes app, for the system and keyboard paths that reach a Take without the app opening them

Version history

VERSION	DATE	CHANGE
1.3	5 September 2026	Founder-voice pass. Two prose semicolons replaced, a line telling the reader to read another line twice now just states the point, and a roadmap tail was cut. No technical change
1.2	1 September 2026	The Companion documents block was a single unlinked line naming one paper. It now lists three and each is a link. In-prose references to the architecture paper are linked on first mention and in "What isn't in here", the section whose whole job is to send you elsewhere
1.1	1 September 2026	Two corrections. The capture screen was listed as an exposure on a passcode-locked phone, opening any app from a locked phone is gated by iOS, so it belongs with Catchlight's own lock and has moved there. And "a format other software reads" claimed present readability of sealed blobs, where what exists is a published format anyone could write a reader against, plus a Markdown export that outlives a subscription
1.0	31 August 2026	First release