The Call and the Code

Withholding one-time passcodes during unverified voice calls: a platform-level proposal and evaluation protocol

Kipkirui Tanui · Keshio Research · Working paper · November 2026 · Draft for comment

License. This paper is released under the Creative Commons Attribution 4.0 International license (CC BY 4.0). You may share and adapt it, including commercially, with attribution. The Keshio name and logo, and the partner catalogue, are not covered by this license.

Suggested citation. Tanui, K., & Keshio Research. (2026). *The Call and the Code: Withholding one-time passcodes during unverified voice calls.* Working paper, November 2026. Licensed under CC BY 4.0.

Abstract

One-time passcodes (OTPs) delivered by SMS remain the dominant second factor for mobile banking and money transfer in many markets. They are routinely defeated not by breaking the cryptography but by persuading the legitimate user to read the code aloud to a caller impersonating a bank, telecom, or government agency. Institutions respond with repeated educational messages, yet losses persist, because the warning arrives on an ordinary day and the theft happens during a live, coached call. We argue that the decisive moment is the overlap of two device-observable states: an active voice call with an unverified counterpart, and the arrival of an OTP. We propose a mechanism in which, during that overlap, the operating system withholds the digits from display, preserves same-device autofill for legitimate flows, and releases the code only after a provenance question ("Did you place this call?") with a short delay and a verified-callback path. We situate the proposal against recent platform features, identify the gap, state testable hypotheses, anticipate adversary adaptation, and describe a pre-registered evaluation. We also report an implementation constraint discovered while preparing a pilot on Android: since Android 15, third-party notification listeners cannot read unredacted OTP content, which limits what an application can do without platform support. Keshio, a small team without a research budget, contributes the hypothesis and the design. We invite platform vendors and academic groups to test it and, if it holds, adopt it. No efficacy results are claimed.

Keywords: one-time passcode, vishing, social engineering, mobile fraud, usable security, in-call intervention

1. Introduction

SMS one-time passcodes were designed to prove possession of a phone. Their failure mode in the wild is different: the user, who does possess the phone, is induced to hand the code to an attacker. In the common script, a caller claims to represent the user's bank, reports suspicious activity, and asks the user to "verify" by reading a code that has just arrived. The code is genuine, the device is trusted, and the user is authenticated. Every technical control passes. Industry commentary on telephone scams makes the same observation: the customer may complete each security step personally while believing they are speaking to a trusted party [11].

In Kenya, a CGAP working paper reports that 14 percent of surveyed digital finance users lost money to fraud [17], and the Central Bank of Kenya's March 2026 draft Financial Consumer Protection Framework includes duties to notify consumers clearly and in time about digital frauds and scams [18]. The standard countermeasure is education. Banks and telecom operators add "never share this code" to the OTP message itself and send periodic awareness campaigns. This paper starts from the premise that such messages are structurally mistimed. They are read in calm conditions, they compete with a caller who is supplying urgency and authority in real time, and the evidence that advance warnings change susceptibility is mixed. A pilot vishing-training study cites an earlier experiment in which a three-month advance warning did not change participants' susceptibility to a simulated call [6], and a separate experiment tested the decay over time of a telephone social-engineering intervention [9].

We therefore ask a narrower question than "how do we educate users?": **can the device itself interrupt the specific moment at which the code is about to be spoken?** The question is tractable because the relevant context, an active call and an arriving OTP, is observable on the device without recording or analysing the conversation.

Contributions:

1. A threat model for call-coached OTP disclosure (Section 2).
2. A review of closely related platform features and academic work, and a statement of the gap (Section 3).
3. A concrete mechanism specification suitable for platform implementation (Section 4).
4. Hypotheses, adversary adaptations, and an evaluation protocol (Sections 5, 6, 8).
5. An implementation constraint relevant to anyone building this above the operating system (Section 7).

2. Problem and threat model

Attack. The adversary places a call, or induces the victim to call, and impersonates an institution. While on the line, the adversary triggers an OTP from a genuine service (a login, a transfer, a password reset), then coaches the victim to read the code aloud. The adversary may also coach the victim to ignore warnings or to complete confirmation dialogs.

Adversary capabilities. The adversary controls the call, knows the victim's institution and often some account details, and can apply urgency and authority. The adversary cannot modify the victim's operating system.

Adversary adaptations considered. Moving the call to a voice-over-IP application; instructing the victim to hang up and "call back"; instructing the victim to confirm any dialog; requesting codes from other channels (authenticator application, email, messaging application); and spoofing a callback number.

Out of scope. SIM swap, malware that reads messages, call-forwarding interception (a distinct vector that Google has recently addressed [3]), and phishing websites that proxy a login.

Design goal. Reduce the probability that a genuine OTP is spoken to a coaching caller, at low cost to legitimate users, and without analysing call audio.

3. Related work and prior art

We searched platform documentation, technology press, and academic venues. The search was not exhaustive, and we did not review patent literature. Table 1 summarises what we found.

Table 1. Related mechanisms and their relation to this proposal

Mechanism	What it does	Relation
Android 15 OTP redaction [1]	Prevents untrusted notification-listener apps from reading unredacted content of notifications where an OTP is detected	Protects against apps, not against the user's own voice

Mechanism	What it does	Relation
Android 16 lock-screen OTP hiding [12]	Reported in developer previews to withhold OTP content on the lock screen by default, subject to security conditions	Hides content in some contexts; not reported as conditioned on call state
Android 16 in-call protections [2, 4]	During calls with non-contacts, blocks enabling sideloading and accessibility services, and warns about screen sharing; processing on-device	Same trigger as ours (call state plus non-contact), different protected action
Screen-share warning with delay [5]	Warns of a likely scam when a banking app is opened during screen sharing; the user may end the call or wait 30 seconds	Precedent for friction via a short wait
Verified financial calls (Brazil) [3]	Checks whether a bank's official app initiated a call claiming to be from the bank; can end the call if verification fails	Building block for exempting genuine institutional calls
Call-forwarding protection [3]	Blocks apps from silently enabling forwarding and warns before a typed forwarding code takes effect	Closes an OTP-interception route; complementary
iOS one-time-code AutoFill [7]	Offers SMS codes in the keyboard, extended in iOS 26 to third-party messaging, mail, and browsers	Convenience; no in-call gating found
iOS 26 Call Screening [8]	Screens calls from unknown numbers before the phone rings	Acts before the call, not at OTP arrival

Mechanism	What it does	Relation
Vishing training and detection studies [6, 10, 13]	Training users to recognise persuasion; on-device AI vishing detection; a roadmap calling for real-time interventions	Supports timing; none withholds the OTP

Two observations follow. First, platform vendors already use the combination “active call with a non-contact” as a trigger for protective friction, which suggests that a call-aware OTP treatment is a natural extension rather than a foreign concept. Second, the user-facing OTP convenience features we reviewed focus on reducing friction; early criticism of iOS Security Code AutoFill noted that it might not distinguish a login code from a transaction-authorisation code [14], which indicates that context-blind convenience has been questioned before.

Gap. To our knowledge, no deployed mechanism withholds the OTP itself conditional on call state and releases it only after a provenance check. We state this as a finding of a bounded search, not as a claim of novelty beyond that scope.

4. Proposed mechanism

4.1 Trigger

The mechanism activates when both conditions hold: (i) a voice call is active, either cellular or a VoIP session in communication audio mode, with a counterpart that is not a saved contact and not a verified institutional number; and (ii) a message or notification is classified as containing an OTP. Android already classifies OTP content in order to redact it [1, 12], so the second condition requires no new classifier. The first uses only call state, direction, and contact status. **No audio is captured or analysed.**

4.2 Hold: withhold the digits, preserve autofill

While the trigger is active, the digits are not rendered in the notification, lock screen, or message list. A notice states that a code has arrived and that the caller must not be given it. **Same-device autofill continues to function**, so a legitimate user entering the code into a field on the same phone loses nothing, whereas reading the code aloud becomes

impossible because the digits are never displayed. This asymmetry is the central design choice: it removes the channel the attack needs while keeping the channel legitimate users need.

Autofill into an attacker-controlled page remains a risk. Domain-bound codes, where a platform supports them, reduce it, and we treat the interaction as a subject for evaluation rather than assuming it is solved.

4.3 Release: provenance question, delay, and exemptions

To see the digits, the user answers one factual question: “Did you call them, or did they call you?” We prefer this to a self-assessment such as “Is someone asking for this code?”, because a coached victim can answer the latter incorrectly without noticing, whereas the former requires a visible falsehood. If the user reports that the caller called them, the digits remain hidden and a callback path is offered (Section 4.4). If the user reports having placed the call to a number not in a verified catalogue, release follows a short delay, consistent with the 30-second precedent in [5], and a final one-line warning.

Exemptions. An outgoing call to a verified institutional number is exempt. Institutions legitimately ask customers who phone them to verify with a code, and a mechanism that blocks that flow would be rejected on usability grounds [15]. Saved contacts receive a lighter, non-blocking notice.

4.4 Redirect: the verified callback

When a call ends and an OTP arrived during or shortly before it, the system presents a card keyed to the OTP’s sender identifier, offering a prefilled dialer entry to the institution’s verified line and its fraud line. Numbers come only from a signed catalogue, never from message text or from the caller. This operationalises a control recommended in industry guidance, that institutions provide customers a reliable way to terminate an unsolicited call and reconnect through an independently accessed channel [11].

4.5 Privacy properties

The mechanism is on-device. It uses three signals (call active, call direction, contact or catalogue status) and the existing OTP classification. It stores no call audio and no code. Any measurement is a local aggregate count.

4.6 Institution-side complement: the reordered SMS

Institutions can apply the same principle without any platform change. They reorder the OTP text so the warning is read first and the digits last, and, where the system knows the amount and recipient, name the transfer in the middle line. For example: “Never share this code. A caller asking for it is a scammer. You are paying KES 45,000 to JOHN KAMAU. Code: 482913.” We assume that many phones show the first line of a message on the lock screen; this assumption is untested and is part of the evaluation. The text should stay in the basic GSM-7 character set so that it fits in a single 160-character segment, and the sender identifier, gateway, and code lifetime stay unchanged. This is arm B in Section 8.1 and the subject of hypothesis H5.

5. Hypotheses

We state hypotheses so that the mechanism can be falsified.

- **H1 (disclosure).** Compared with a standard SMS warning, an in-call interruption reduces the probability that a participant reads a code aloud to a scripted caller.
- **H2 (question form).** The provenance question (“Did you call them, or did they call you?”) yields lower disclosure than a self-assessment question.
- **H3 (cost).** The mechanism increases legitimate-task completion time and abandonment by less than a pre-specified margin.
- **H4 (callback).** Presenting a verified-callback card increases the proportion of participants who end the call and verify through an independent channel.

We add a fifth hypothesis on the institution side. H5 (template): a reordered SMS, with the warning first and the digits last, reduces disclosure relative to a standard SMS (arm B against arm A in Section 8.1). We make no prediction about effect sizes. The prior literature on warning effectiveness is mixed [6, 9], and we regard a null result as informative.

6. Adversary adaptation

Adaptation	Expected effect	Mitigation to evaluate
Move to a VoIP call	Evades call-state detection if audio mode is not checked	Treat communication audio mode as a call; measure coverage

Adaptation	Expected effect	Mitigation to evaluate
“Hang up, I’ll call you right back”	Code is released at call end	Callback card persists for the code’s lifetime; re-alert on a repeat call from the same non-contact
“Tap yes on the screen”	Defeats a self-assessment prompt	Factual provenance question; short delay
Request code from another channel	Displacement to authenticator, email, or messaging	Extend classification to other channels; report SMS-only coverage
Spoofed callback number	Victim calls the attacker	Callback numbers only from a signed catalogue, verified out-of-band
Catalogue poisoning	Safety feature becomes the attack	Identity verification, independent callback, two-person approval, signed versioned catalogue, revocation

The first row is a limitation of any call-state approach. The last is the main new attack surface the proposal introduces, and we regard its governance as part of the design rather than an operational detail.

7. Implementation above the operating system, and a constraint

Keshio operates an Android application that is not the default SMS handler, and could run a small on-device pilot there. Three constraints follow, and the first is not obvious from the SMS-oriented literature.

(a) The application cannot hide or delay the message. The SMS still reaches the stock inbox and notification shade. The pilot is therefore an interruption, not a hold, and results from it should be read as a lower bound on what a platform-level hold could achieve.

(b) Android 15 limits how an application can detect the OTP. Android 15 prevents untrusted notification-listener applications from reading unredacted content of notifications in which an OTP is detected [1]. Developer reports describe affected applications receiving a placeholder in place of the content, and an application-operations setting that trusted applications can be granted [16]. An anti-fraud application built on

notification access therefore cannot rely on seeing the message. Because the content is replaced by a placeholder, an application may still observe that a sensitive notification from a messaging application arrived. **We have not verified this on current devices and treat it as a hypothesis to test before building further.**

(c) Alternatives that avoid reading content. Two options are worth evaluation: a partner-integrated signal in which a participating institution notifies the application, through a verified channel, that an OTP has been issued (carrying no digits); and a request that the platform expose an "OTP arrived" event, without content, to vetted anti-fraud applications.

These constraints are themselves a finding: the platform's protection of OTP content against malicious applications also blocks beneficial third-party interventions, which is an argument for a first-party, call-aware treatment.

8. Evaluation protocol

8.1 Randomised simulated-vishing study

Adult participants, recruited with informed consent to a general phone-use study, receive a scripted call from a trained confederate who requests a one-time code issued by a study-controlled service. No real funds or real accounts are involved. Participants are debriefed immediately and may withdraw their data, following the approach used in prior vishing-intervention pilots that withheld the full purpose until debriefing [6]. Ethics review is required before any contact.

Participants are randomised to one of four arms:

- **A (control):** standard SMS with a "do not share" line.
- **B:** reordered SMS (warning first, transfer details, digits last).
- **C:** in-call interruption with the provenance question.
- **D:** C plus the verified-callback card.

Primary outcome: the code is read aloud to the caller. **Secondary outcomes:** latency to disclosure; ending the call; initiating an independent callback; self-reported confidence. Analysis is pre-registered, intention-to-treat, with effect sizes and confidence intervals reported whether or not they favour the mechanism.

8.2 Legitimate-use cost

A scenario task set measures completion time and abandonment in benign overlaps: phoning one's own bank, helping a relative by phone while logging in, and a delivery call during a sign-in. This tests H3 and the exemption rules.

8.3 Field telemetry

On-device aggregate counts: OTP arrivals during calls; alerts shown; provenance answers; taps on callback and report actions; calls ending soon after an alert. These are behavioural proxies, not fraud outcomes. To support a fraud-loss claim, they must be paired with an institution's fraud-report data for the same sender identifier and period, under a data-sharing agreement.

8.4 Accessibility and language

The interruption must work without sound and without reading, with screen-reader support and haptic reinforcement, and be tested with low-literacy, hearing-impaired, and older participants in each deployment language. These tests are part of the study, not a separate product task.

8.5 Data protection

The study and any pilot must comply with applicable data-protection law in the deployment country, including consent for on-device processing and for aggregate reporting. We will obtain legal review before launch and before publishing aggregates.

9. Limitations

The call-state trigger will miss some VoIP calls and cannot see attacks that never involve a call. The mechanism covers SMS and does not protect codes obtained through other channels unless extended. Laboratory disclosure rates may not transfer to real victims under real emotional pressure. The provenance question may be answered falsely by a sufficiently coached participant, and its marginal value over a delay alone is an empirical question. The catalogue introduces governance and trust burdens. Finally, our literature search covered public documents and was not exhaustive; related work may exist that we did not find.

10. Requests to platform vendors and researchers

1. **Platform vendors:** expose a call-aware OTP sensitivity signal, or implement the hold-display, allow-autofill treatment as a first-party feature, building on existing in-call and OTP-classification infrastructure on Android [1, 2, 12] and on AutoFill infrastructure on iOS [7].
2. **Android:** consider a scoped, content-free "OTP arrived" event for vetted anti-fraud applications, given the limits described in Section 7.
3. **Apple:** consider extending Call Screening and one-time-code handling to the in-call case.
4. **Researchers:** collaborate on the protocol in Section 8, including replication in other markets and languages.

11. Contribution, partnership and funding

Keshio contributes the hypothesis, the mechanism design, and a path to deployment: an Android application, a catalogue of partner institutions with verified lines, and anonymised on-device counts that can feed a study. Keshio has no research team or budget to run the evaluation in Section 8, so we are seeking four things:

- **An academic principal investigator** to lead the study and obtain ethics approval.
- **A research funder.** Platform-vendor research awards for scams and financial fraud, and development-finance research programmes on consumer protection, are natural fits.
- **A platform sponsor** to test the mechanism on a device or in a first-party build.
- **An institutional partner** to supply sender-level fraud-report data for the same period.

We propose that design materials, instrumentation, and catalogue governance be shared with the research partner on agreed terms, and that results be published whether or not they favour the mechanism.

12. Conclusion

The attack that defeats OTPs is not technical, and the moment it happens is observable. Platforms have begun to use "active call with a non-contact" as a trigger for protective friction. Extending that principle to the OTP itself, by hiding the digits while preserving

autofill, offering a provenance question, and routing the user to a verified callback, is a small step with a testable payoff. We do not claim it works. We propose a design, a set of falsifiable hypotheses, and an evaluation, and we invite collaboration.

Disclosure

Use of AI assistance. Portions of this paper, including literature summaries and drafting, were prepared with assistance from Claude, an AI model made by Anthropic. The authors are responsible for the content.

Conflicts of interest and funding. Keshio operates SMS and Android software and could benefit commercially if the ideas described here are adopted. This work was not externally funded.

References

All web sources were accessed in November 2026.

[1] Android Developers. “Behavior changes: all apps” (Android 15). https://developer.android.com/about/versions/15/behavior-changes-all

[2] Android Authority. “Android 16 will fight scammers by blocking certain features during phone calls.” February 2025. https://www.androidauthority.com/android-16-phone-call-protections-3526068/

[3] Google. “New Android protections to help keep you safe from mobile scams.” October 2026. https://blog.google/security/new-android-scam-protections/

[4] Android Authority. “These new Android tools could stop scammers from robbing your loved ones blind.” https://www.androidauthority.com/new-anti-scam-features-android-show-3556831/

[5] NewsPoint. “Google enhances Android 16 with cutting-edge security features to fight phone scams.” https://www.newspointapp.com/english/tech/google-enhances-android-16-with-cutting-edge-security-features-to-fight-phone-scams/articleshow/121163857.cms

[6] “Training Users to Recognize Persuasion Techniques in Vishing Calls.” ACM CHI Extended Abstracts, 2023. https://dl.acm.org/doi/10.1145/3544549.3585823

[7] MacRumors. “iOS 26 and macOS Tahoe Expand AutoFill Feature for One-Time Codes.” June 2025. https://www.macrumors.com/2025/06/12/ios-26-macos-tahoe-expand-one-time-code-autofill/

[8] Apple. “Apple helps safeguard users from scams and fraud while protecting personal data.” https://www.apple.com/sg/newsroom/2025/10/apple-helps-safeguard-users-from-scams-and-fraud-while-protecting-personal-data/

[9] “Telephone-based social engineering attacks: An experiment testing the success and time decay of an intervention.” https://www.researchgate.net/publication/289520596_Telephone-based_social_engineering_attacks_An_experiment_testing_the_success_and_time_decay_of_an_intervention

[10] “‘I know my data doesn’t leave my phone, but still feel like being wiretapped’: Understanding (Mis)Perceptions of On-Device AI Vishing Detection Apps.” ACM, 2025. https://dl.acm.org/doi/10.1145/3706599.3719784

[11] FinCrime Intelligence. “How Phone Scams are Becoming More Sophisticated: What to Watch For.” https://fincrimeintelligence.com/how-phone-scams-are-becoming-more-sophisticated/

[12] PhoneArena. “Android 16 might finally fix a privacy problem you didn’t know you had.” https://www.phonearena.com/news/android-16-might-finally-fix-a-privacy-problem-you-didnt-know-you-had_id170453

[13] “Vishing: Detecting social engineering in spoken communication: a first survey and urgent roadmap to address an emerging societal challenge.” University of Augsburg repository. https://opus.bibliothek.uni-augsburg.de/opus4/files/121734/1-s2.0-S0885230825000270-main.pdf

[14] Tom’s Guide. “Security Researcher Has a Big Problem with Apple’s iOS 12.” https://www.tomsguide.com/us/ios-12-security-code-autofill,news-27549.html

[15] Flagstar Bank. “How scammers steal one-time codes and how to stop them.” https://www.flagstar.com/personal/learn/learning-hub/protecting-your-finances/how-scammers-steal-one-time-codes-and-how-to-stop-them.html

[16] Home Assistant Android issue #5153, “Sensitive notification content hidden in Last Notification sensor.” https://github.com/home-assistant/android/issues/5153

[17] CGAP. “Solutions to Protect Consumers from Fraud in Digital Finance.” 2026. https://www.cgap.org/research/publication/solutions-to-protect-consumers-fraud-in-digital-finance

[18] Central Bank of Kenya. Financial Consumer Protection Framework (draft). March 2026. https://www.centralbank.go.ke/wp-content/uploads/2026/04/Consumer-Protection-Framework-March-2026.pdf