

Refusals That Reveal: Server-Enforced Teen Contact Policies and Their Age-Inference Side Channels in a Production Social Platform

Tafadzwa Innocent Tauro, M.S. Cybersecurity, Quinnipiac University

Abstract—Platforms that admit teenagers increasingly restrict who may contact them: Instagram limits teen accounts to messages from people they follow, and TikTok disables direct messages for users under 16. We show that such protections, if implemented naively, can themselves disclose which accounts belong to minors. We present the child-safety architecture of Kibhi (kibhi.com), a production short-video and messaging platform for users aged 13 and over whose web and Android clients share one API, in which every contact rule is enforced at the server. We define refusal indistinguishability, the requirement that a refusal caused by a protected user’s age be indistinguishable in content, status, and latency from refusals any user can trigger, and give a taxonomy of seven age-inference side channels in contact APIs. An attack suite that drives the real HTTP API as a malicious client found two oracles in the deployed system. First, refusals to teens were uniquely worded. Second, after the wording was fixed, a teen’s call refusal still took 96 ms longer than a block, letting one probe identify a teen with 96% accuracy. Correcting the call path alone moved the leak to direct messages, where teens’ refusals became 41 ms faster. After all paths were made constant-work, refusal latencies were statistically indistinguishable (direct messages $p = 0.99$, calls $p = 0.45$) and the suite passed 13 of 13 checks. We also enumerate every server route that creates contact, quantify contact-discovery enumeration (an unkeyed hash of every North American number inverts in about five hours on one laptop core; a new daily cap cuts per-account API probing by 98.75%), and map the design to COPPA and the U.K. Age Appropriate Design Code. The attack and conformance suites are released for reuse.

Index Terms—Child online safety, side channels, timing attacks, access control, privacy, age assurance, contact discovery, safety by design, social networks.

I. Introduction

ONLINE enticement, in which an adult communicates with a child with intent to commit a sexual offense or abduction, is growing quickly: reports of it to the CyberTipline of the U.S. National Center for Missing & Exploited Children rose from 44,155 in 2021 to 186,819 in 2023 [6]. Platforms have responded by restricting who can reach minors. TikTok disabled direct messaging for users under 16 in 2020 [7]; in 2024 Instagram moved users under 18 into “Teen Accounts” that are private by default and can be messaged only by people they follow or are connected to [8]. Regulators frame the same expectations

as design duties: COPPA restricts collecting personal information from children under 13 [9], the U.K. Age Appropriate Design Code expects high-privacy defaults for under-18 users [10], the U.K. Online Safety Act 2023 requires services likely to be accessed by children to assess and mitigate risks to them [11], and Article 28 of the EU Digital Services Act requires online platforms accessible to minors to ensure a high level of privacy, safety, and security for them [12].

This article makes two points about such protections. The first is familiar but often violated in practice: a contact rule protects nobody unless the server enforces it, because a modified client or a direct API call bypasses any rule that lives in an app. The second is less obvious. A rule that treats minors differently is itself a source of information. If a stranger’s message to a 15-year-old is refused with different words, a different error code, or a different response time than a message to an adult who has simply turned messages off, then the refusal tells the stranger that the account belongs to a minor. Identifying minors is exactly what a predator wants to do first. The protection becomes an age oracle.

We study both points in Kibhi (<https://kibhi.com>) [1], a short-video and messaging platform that the author built and operates in production, with a web client and a native Android client that share one HTTP API. The contributions are:

- 1) A server-enforced teen contact policy, stated formally, with its age gate, defaults, and message-request mechanism (Section IV), and an enumeration of every route that creates contact, showing where each enforces it (Section V).
- 2) A definition of refusal indistinguishability and a taxonomy of seven age-inference side channels in contact APIs (Section VI).
- 3) An open attack suite that drives the real API as a malicious client, and its results on the deployed system: a content oracle and a timing oracle, the observation that fixing one timing path moved the leak to another, and verified elimination of both (Section VII).
- 4) A cost analysis of contact-discovery enumeration against stored hashes and against the matching API, with a rate-limit change that reduces per-account probing by 98.75% (Section VIII).
- 5) A mapping of the design to COPPA and the Age

T. I. Tauro is an independent researcher in Connecticut, USA, and received the M.S. degree in cybersecurity from Quinnipiac University, Hamden, CT, USA, in 2026 (e-mail: tafadzwatauro09@gmail.com).

The author designed and operates the Kibhi platform evaluated in this article.

Appropriate Design Code that reports where it falls short (Section IX).

A testing checklist distilled from this work is given in Section X. All measurements were made on a development database with throwaway accounts; every defect found was fixed in the production code base.

II. Background and Related Work

A. Teen protections on large platforms

TikTok’s 2020 change was described at the time as the first restriction of private messaging by age on a major platform [7]. Instagram’s 2024 Teen Accounts apply private-by-default profiles and message restrictions to users under 18, with parental permission required for under-16s to relax them [8]. In April 2025 Meta extended Teen Accounts to Facebook and Messenger, limiting teens to messages from people they are connected with, and reported that 54 million teens had been moved into Teen Accounts on Instagram and that 97% of those aged 13 to 15 had kept the protections on [2]. These announcements describe policies; they do not describe how enforcement is implemented or whether refusals are distinguishable, which is the gap this article addresses.

B. Recent regulatory developments

Since 2025, age-differentiated treatment has moved from voluntary practice to legal obligation in several jurisdictions. On 14 July 2025 the European Commission published guidelines under Article 28(1) of the Digital Services Act listing measures that platforms accessible to minors should take against risks such as grooming [3]. From 25 July 2025, Ofcom has enforced its Protection of Children Codes under the U.K. Online Safety Act, which require higher-risk services to use highly effective age assurance [4]. On 10 December 2025, Australia began requiring social media platforms to prevent people under 16 from holding accounts, with fines for platforms of up to about 50 million Australian dollars [5]. Each of these regimes leads platforms to treat minors differently from adults. The finding of this article is that such differences must be implemented so that they do not themselves reveal who the minors are.

C. Age assurance

Methods for establishing age range from self-declaration to document checks and facial age estimation [10]. Guidance on COPPA describes a neutral age screen, which asks for a date of birth without indicating which answer grants access and prevents an immediate retry after an under-age answer [9]. Self-declaration is cheap and privacy-preserving but easy to falsify; Section XI discusses what this leaves open.

D. Side channels in web applications

Timing differences in web responses can reveal private state. Bortz and Boneh showed that response times expose, for example, whether a username is valid at a site, and that such differences are measurable remotely [13]; Crosby et al. quantified how small a difference a remote attacker can resolve with repeated measurements [14]. Account-enumeration testing, which checks whether login, registration, or password-reset responses differ for existing and non-existing accounts, is a standard part of web security testing [15]. We apply the same reasoning to a different secret: not whether an account exists, but whether it belongs to a minor.

E. Contact discovery

Hagen et al. showed that the contact discovery of popular messengers can be abused at scale, crawling 10% of U.S. mobile numbers on WhatsApp and all of them on Signal, and that hashing numbers adds little protection because the space is small [16]. Kales et al. proposed private set intersection for mobile contact discovery [17]. In 2022, data on 5.4 million Twitter accounts was compiled through an interface that revealed which account a phone number or email belonged to [18].

III. Threat Model

We consider six adversaries.

- A1, contacting stranger: an adult who wants private contact with a person aged 13 to 17 they do not know.
- A2, under-age registrant: a child under 13 trying to create an account, possibly retrying.
- A3, malicious client: any adversary using a modified app or calling the API directly.
- A4, enumerator: an account holder submitting phone numbers to learn which belong to users.
- A5, database thief: an attacker with a copy of the database but not the server’s secrets.
- A6, age profiler: an account holder who probes accounts to learn which belong to minors, as a precursor to targeting them through any channel.

The server and its environment are trusted and traffic uses TLS. Accounts must verify an email address before they can message, call, or post. The declared date of birth is not assumed to be true.

IV. System Design

Fig. 1 shows the architecture. Both clients, and any third-party caller, reach the same API; every policy decision is made in server code before a row that would create contact is written.

A. Neutral age gate

A date of birth is requested at sign-up, or once on the next visit for older accounts. An under-13 answer creates no account, the response does not reveal which date would succeed, and an HTTP-only cookie blocks further sign-up

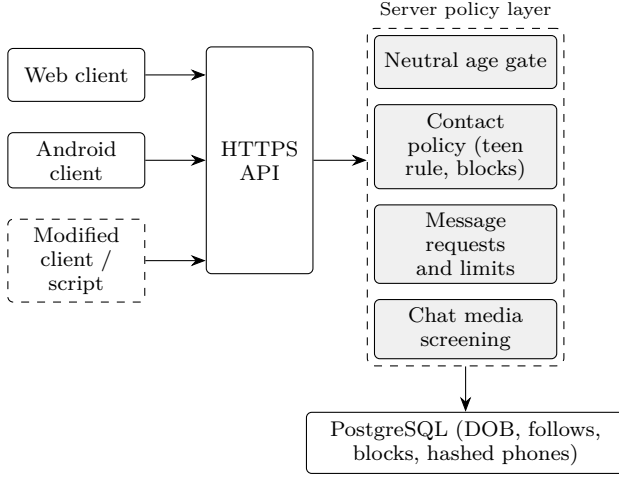


Fig. 1. Every client, including a modified one, reaches the same API; contact is created only after the server policy layer allows it.

attempts from that browser for 30 days. The date of birth is never sent to any client and cannot be edited by the user. An existing account whose holder reports being under 13 is deleted with its content.

B. Teen defaults

A user whose date of birth places them between 13 and 17 is made private, is excluded from “people you may know” suggestions shown to people who do not already know them, and cannot enable location-based discovery.

C. Contact policy

Let $\text{teen}(t)$ hold when t is aged 13 to 17, and

$$\text{known}(t, s) \equiv \text{follows}(t, s) \vee \text{wrote}(t, s) \vee \text{accepted}(t, s), \quad (1)$$

where $\text{wrote}(t, s)$ means t has messaged s and $\text{accepted}(t, s)$ means t accepted a request from s . A direct message, a call, or an addition to a group or community from s to t is permitted only if

$$\neg \text{blocked}(s, t) \wedge (\neg \text{teen}(t) \vee \text{known}(t, s)). \quad (2)$$

For adult recipients, a message from someone not known to them becomes a message request: it waits in a separate folder, read receipts are withheld until the recipient replies or accepts, media from people the recipient does not follow are covered until tapped, and a sender may place at most three messages in a pending request and open at most 20 new requests per day. For teen recipients the message is refused.

D. Chat media screening

Photos and videos in direct and group chats are screened for nudity, sexual content, graphic violence, and weapons before the recipient can retrieve them, using a commercial classifier with an on-server fallback model. Media that fails is refused. For messages that arrive as requests, screening fails closed: media that could not be checked is refused rather than delivered.

E. Contact discovery

The Android client computes SHA-256 of each contact’s normalized email address and E.164 phone number on the device and sends only the hashes. For users who verified a phone number by SMS, the server stores $\text{HMAC}_K(\text{SHA256}(n))$ under a 256-bit server secret K [19] and the last four digits for display. Submitted hashes are keyed the same way for comparison and are not stored.

V. Enforcement Coverage

Table I lists every route that writes a row creating contact between two users, found by locating each database function that inserts into the messages, calls, group- and community-membership, follow, and request-acceptance tables, and then each route that calls one of them.

Two routes create membership or acceptance without the teen rule, and both are initiated by the person being protected. The residual exposure is in communities: a teen can be in a community only if someone they follow added them, but other members of that community, and of its groups the teen joins, may be adults they do not know. Messages there are visible to all members and can be reported, but are not restricted by the contact policy.

VI. Refusal Indistinguishability

A protection that treats minors differently must not let an observer distinguish minors by the way it refuses.

Definition 1 (Refusal indistinguishability): Let $R(s, t)$ be the distribution of responses (status, headers, body, and latency) that sender s receives when an action toward t is refused. A contact API is refusal-indistinguishable with respect to age if for every sender s , every teen t refused by the teen rule, and some adult a refused by a setting any user can choose (messages turned off, or a block), $R(s, t)$ and $R(s, a)$ are computationally indistinguishable to s .

The definition requires an adult cover: a refusal reason open to everyone, so that a refusal reveals only that one of several reasons applies. Without such a cover no wording can hide the reason.

Table II lists the age-inference channels we identified in a contact API, with their status in Kibhi after this work. Channels C1 to C3 concern the refusal itself and are testable; C4 to C7 are structural signals that arise from the protections’ defaults and can be reduced but not eliminated.

VII. Evaluation

A. Policy conformance

A harness (`child-safety-eval.cjs`) imports the production policy modules unchanged and runs them against a development PostgreSQL database with six throwaway accounts: a teen T (born 2011) and five adults, where T follows adult B and has messaged adult C . All 11 scenarios in Table III produced the expected decision. The teen account was private by default and did not appear among a stranger’s 50 suggested accounts.

TABLE I
Server routes that create contact with a user, and where each enforces the policy.

Contact created	Route	Enforcement
Direct message	POST/api/messages/[username]	Block check; directMessagePolicy: request rules, teen refusal, request limits
Forwarded message	POST/api/messages/forward	Same directMessagePolicy for each recipient
Voice or video call	POST/api/calls	Block and teen checks together; recipient must have a verified email
Add members to group	POST/api/groups, /api/groups/[id]/members	allowedMemberIds: block, recipient’s message setting, teenGroupFilter
Add members to community	POST/api/communities, /api/communities/[id]/members	allowedMemberIds as above
Group message	POST/api/groups/[id]/messages	Sender must be a member (membership gated above)
Join a community’s group	POST/api/groups/[id]/join	Self-initiated; caller must already be a community member
Accept a message request	POST/api/messages/[username]/request	Self-initiated by the recipient

TABLE II
Age-inference side channels in a contact API.

Channel	Status in Kibhi
C1 Refusal wording	Found; fixed (identical to “messages off” and block)
C2 Status and error code	Found (distinct code on calls); fixed
C3 Refusal latency	Found (calls, then messages); fixed, verified
C4 Private profile by default	Residual; adults may also be private
C5 Absent from suggestions and location features	Residual; also true of adults who opt out
C6 Silent omission from a group	Same answer as any refused member
C7 Contact-discovery match	Matches only users who verified a phone; no age signal

TABLE III
Policy conformance on a live database (11/11 as expected).

Scenario	Expected	Actual
Stranger adult messages teen	deny	deny
Stranger adult calls teen	deny	deny
Stranger adult adds teen to group	deny	deny
Adult the teen follows messages teen	allow	allow
Adult the teen follows calls teen	allow	allow
Adult the teen follows adds teen to group	allow	allow
Adult the teen wrote to replies	allow	allow
Adult the teen wrote to calls	allow	allow
Stranger adult messages adult	request	request
Stranger adult calls adult	allow	allow
Stranger adult adds adult to group	allow	allow

B. Malicious-client attack suite

The attack suite (`http-attack-eval.cjs`) runs the application server locally against the development database, creates and signs in throwaway accounts through the real login endpoint, and issues requests exactly as a modified client would. A denial counts as a pass only if it is the policy’s answer (HTTP 403, or 400 when the only requested member was refused), not an authentication

TABLE IV
Malicious-client attack suite after all fixes (13/13 pass).

Attempt	Expected	Result
Stranger: direct message to teen	deny	403
Stranger: voice call to teen	deny	403
Stranger: create group with teen	deny	400
Stranger: community with teen and an adult	teen omitted	omitted
Stranger: message unknown adult	request	200
Stranger: forward that message to teen	deny	403
Anonymous: message teen	deny	401
Teen vs. messages-off adult, message refusal	identical	identical
Teen vs. blocking adult, call refusal	identical	identical
Teen vs. messages-off adult, forward refusal	identical	identical
Followed adult: message teen	allow	200
Followed adult: group with teen	allow	200
Followed adult: call teen	allow	200

failure. This rule mattered: the first run of the suite reported every attack “denied” because a cookie-handling bug left every request unauthenticated, and only the control requests, which also failed, revealed it.

On the system as deployed, every attack was refused, but the refusals were not indistinguishable. A stranger’s message to a teen returned “@username only gets messages from people they follow,” and a call returned a distinct error code; no adult setting produces either. Any account could therefore classify an arbitrary account as a teen with one request (channels C1 and C2). The fix returns, for a teen, exactly the response of an adult with messages turned off (messages and forwards) or of a block (calls). After the fix the suite passed 13 of 13 checks (Table IV), including byte-for-byte equality of status and body for the three refusal pairs.

C. Timing side channel

Identical content does not imply identical timing. The timing harness (`timing-oracle-eval.cjs`) sends refusals from a stranger to a teen and to an adult cover in randomized interleaved order, 100 pairs for direct messages (adult with

TABLE V
Refusal latency, teen minus adult cover (median, ms).

Round	Path	Δ median	z	p	δ
1 (deployed)	Message	+7.2	1.07	0.29	0.09
	Call	+96.2	8.83	$< 10^{-15}$	0.94
2 (partial fix)	Message	-40.9	-8.92	$< 10^{-15}$	-0.73
	Call	-3.2	-0.34	0.74	-0.04
3 (fixed)	Message	+0.2	0.01	0.99	0.00
	Call	-3.1	0.76	0.45	0.08

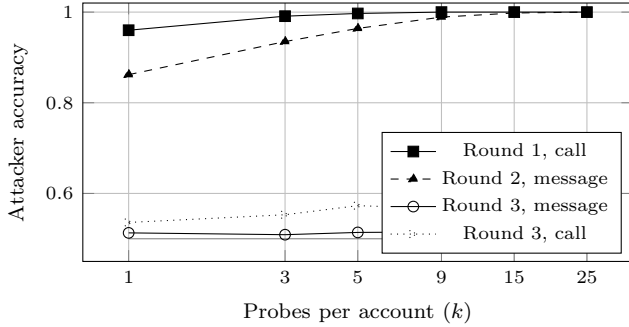


Fig. 2. Accuracy of an attacker classifying an account as teen or adult from k refusal timings. The resampling estimate is optimistic when the medians differ by chance; in round 3 neither difference is statistically significant.

messages turned off) and 60 pairs for calls (adult who blocked the caller), paced below the platform’s rate limits, from a client on the same machine as the application server, which reached a cloud database over the internet. We compare latencies with a two-sided Mann-Whitney U test and Cliff’s δ , and estimate an attacker who takes k probes of an unknown account, compares their median with the midpoint of the two medians, and guesses (accuracy by resampling).

Table V and Fig. 2 show three rounds. As deployed (round 1), a teen’s call refusal took 96.2 ms longer than a block: the server returned immediately after the block check but ran further queries for the teen rule, so a single probe identified a teen 96% of the time. In round 2, the block and teen checks were made to run together and the teen check to run its two lookups in parallel. This closed the call channel but opened one in direct messages: an adult’s refusal still performed the request-limit lookup that a teen’s skipped, so teens’ refusals became 40.9 ms faster, again identifiable from one probe 86% of the time. In round 3 the request-limit lookup runs for every recipient, and neither path shows a significant difference.

The resampling estimator reuses the measured samples both to set the threshold and to test it, so it is biased upward when the true difference is zero; the small rise for round-3 calls at large k should be read together with $p = 0.45$. A remote attacker sees additional network jitter, which raises the number of probes needed but, as prior work shows, does not remove a difference of tens of milliseconds [14].

TABLE VI
Policy decision latency after the fixes (ms, $n = 200$).

Check	p50	p95	mean
SELECT 1 round trip	37.4	44.0	38.1
Message policy, stranger $\rightarrow$ teen	86.2	124.2	90.8
Message policy, adult $\rightarrow$ adult	86.9	121.4	94.2
Call policy, stranger $\rightarrow$ teen	43.8	49.5	44.2
Group filter (two candidates)	77.7	89.1	79.4

TABLE VII
Per-account capacity to probe phone numbers through the matching API.

	Before	After
Numbers per day	2,400,000	30,000
One area code (10^7)	4.2 days	333 days
All 10-digit numbers (10^{10})	11.4 years	913 years

D. Overhead

After the changes, each decision was timed 200 times after 20 warm-up calls, from the same machine, against the cloud database (Table VI). The direct-message policy costs about the same for a teen and an adult (86 ms, roughly two round trips) because both paths now perform the same lookups; the call check needs one round of parallel lookups.

VIII. Contact-Discovery Enumeration

A. Offline: a stolen database (A5)

One core of the test laptop (AMD Ryzen 7 8840HS, Node.js 22.13.1) computed 540,238 SHA-256 hashes per second over 13-character E.164 strings. The North American Numbering Plan has at most 10^{10} ten-digit numbers, so a lookup table of unkeyed hashes for all of them takes about $10^{10}/540,238 \approx 18,500$ s, or 5.1 hours on one core, and a single area code (10^7 numbers) about 19 seconds. Unkeyed phone hashes therefore give essentially no confidentiality, consistent with [16]. Kibhi stores $\text{HMAC}_K(\text{SHA256}(n))$; without K the same attack requires guessing a 256-bit key.

B. Online: the matching API (A4)

The matching API must say whether a hash belongs to a user, so it is an oracle however hashes are stored. One account could submit 5,000 phone hashes per request and 20 requests per hour: 2.4 million numbers per day. The Android client sends a whole address book in one request, so a person needs only a few requests a day; we added a cap of six per account per day. Table VII shows the effect.

An adversary can divide the work across many accounts. Each must verify an email address, and sign-ups are rate limited per IP address (300 per hour), which raises the cost of a Sybil attack without preventing it; private set intersection [17] would remove the oracle and is left as future work.

TABLE VIII
Mapping to COPPA and the Age Appropriate Design Code (AADC).

Requirement	Status	Mechanism
COPPA: no collection from under-13s	Met	Neutral age screen, 30-day retry block, deletion on report
AADC 3: age appropriate application	Partial	Self-declared age; no estimation
AADC 7: high-privacy default settings	Met	Teen accounts private by default
AADC 8: data minimisation	Met	Date of birth never sent to clients
AADC 10: geolocation off by default	Met	Location discovery unavailable to teens
AADC 11: parental controls	Not met	None yet
AADC 12: profiling off by default	Partial	Not suggested to strangers; feed personalization applies to all users
AADC 15: online tools	Met	Report and block on every surface

TABLE IX
Checklist for testing age-based contact protections.

Check	Adv.
T1 Enumerate every route that creates contact; confirm each calls the policy	A1, A3
T2 Exercise each route over HTTP as a modified client, with controls that must succeed	A3
T3 Count a denial only if it is the policy's answer, not an authentication error	A3
T4 For each age-based refusal, identify an adult cover reason any user can trigger	A6
T5 Compare status, headers, and body of teen and cover refusals byte for byte	A6
T6 Compare their latency, interleaved, with a rank test; re-test after every fix	A6
T7 Measure discovery oracles per account per day and the cost of a full sweep	A4
T8 Store only keyed hashes of low-entropy identifiers; keep the key out of the database	A5

IX. Regulatory Mapping

Table VIII maps the design to COPPA and to the relevant standards of the U.K. Age Appropriate Design Code. We report shortfalls as well as matches.

X. A Testing Checklist for Teen Protections

The checks that found the defects above are cheap to run and not specific to Kibhi. Table IX lists them as a checklist that any platform with age-based protections can apply; each item names the adversary it addresses. The constant-work pattern that closed the timing channel is shown in Fig. 3: every lookup that any refusal path needs is issued together, before the decision, so the work done is independent of the recipient's age.

XI. Discussion and Limitations

Lessons. Four lessons generalize beyond Kibhi. (i) Enforce contact rules at the server, at the exact points where contact is created. (ii) Every rule that treats a protected

```

function RefuseOrAllow(s, t)
  (b, τ, k, u) ← await all {blocked(s, t), teen(t),
    known(t, s), requestUsage(s, t)}
  if b or (τ ∧ ¬k) or optedOut(t): return Cover
  if ¬k ∧ overLimit(u): return LimitReached
  return Allow

```

Fig. 3. Constant-work refusal: all lookups any refusal path needs are issued together, and every age-based refusal returns the same Cover response as an adult who opted out.

group differently is a candidate side channel; give each refusal an adult cover and test that the two are identical. (iii) Test timing as well as content: a constant-work fix must cover every path, and fixing one path can move the leak to another, as round 2 showed. (iv) Keyed hashing protects stored identifiers, not the online matching oracle; that requires rate limits or cryptographic protocols.

Self-declared age. The architecture is only as accurate as the declared date of birth. A teen who claims to be an adult receives adult defaults. Stronger age assurance would narrow this gap at the cost of collecting more sensitive data.

Structural channels. Private-by-default profiles and absence from suggestions (C4, C5) remain weak signals because adults can produce them too; their strength depends on how many adults do, which we did not measure.

Communities. Shared spaces can contain adults a teen does not know (Section V).

Measurement setting. Timing was measured from a client on the same machine as the application server, which reached a cloud database over the internet; a remote attacker would see more jitter. The suites ran against a development database with synthetic accounts, not production traffic.

XII. Ethics

No real user's data was accessed or affected. All experiments used throwaway accounts on a development database, which were deleted afterwards, and the platform is operated by the author. Each defect found was fixed in the platform's code base, and the fixes are being deployed to production before this article's public release.

XIII. Conclusion

Server-side enforcement gave Kibhi one set of teen contact rules for every client, at a cost of one or two database round trips per decision. Testing those rules as an attacker would found that the protections themselves revealed which accounts belonged to minors, first through wording and then through timing, and that a partial timing fix merely moved the leak. Refusal indistinguishability, the taxonomy of channels, and the open attack and timing suites are offered so that other platforms can test their own teen protections the same way.

Acknowledgment

In accordance with IEEE policy, the author discloses that an AI system (Claude, Anthropic) was used to help draft the text of all sections, to write the evaluation scripts in docs/papers/eval, to implement the fixes described in Sections VII and VIII, and to format tables and figures. The author designed the system and the study, reviewed every change, verified every reported number against the result files, and takes full responsibility for the content.

References

- [1] Kibhi, “Kibhi: short video and messaging,” 2026. [Online]. Available: <https://kibhi.com>
- [2] Meta, “Teen Accounts expand to Facebook and Messenger with new protections,” Meta Newsroom, Apr. 2025.
- [3] European Commission, “Guidelines on measures to ensure a high level of privacy, safety and security for minors online, pursuant to Article 28(4) of Regulation (EU) 2022/2065,” Jul. 14, 2025.
- [4] Ofcom, “Protection of Children Codes of Practice,” Online Safety Act 2023, in force Jul. 25, 2025. [Online]. Available: <https://www.ofcom.org.uk/online-safety/protecting-children>
- [5] eSafety Commissioner, “Social media age restrictions,” Australian Government, Dec. 2025. [Online]. Available: <https://www.esafety.gov.au/young-people/social-media-age-restrictions>
- [6] National Center for Missing & Exploited Children, “2023 CyberTipline Report,” 2024. [Online]. Available: <https://www.missingkids.org/cybertiplinedata>
- [7] S. Perez, “TikTok to launch parental controls globally, disable direct messaging for users under 16,” TechCrunch, Apr. 16, 2020.
- [8] Meta, “Introducing Instagram Teen Accounts: Built-in protections for teens, peace of mind for parents,” Meta Newsroom, Sep. 17, 2024.
- [9] U.S. Federal Trade Commission, “Children’s Online Privacy Protection Rule,” 16 C.F.R. pt. 312.
- [10] Information Commissioner’s Office, “Age appropriate design: A code of practice for online services,” Wilmslow, U.K., 2020.
- [11] Online Safety Act 2023, c. 50, U.K. Parliament, 2023.
- [12] Regulation (EU) 2022/2065 of the European Parliament and of the Council on a Single Market for Digital Services (Digital Services Act), Off. J. Eur. Union, L 277, Oct. 27, 2022, Art. 28.
- [13] A. Bortz and D. Boneh, “Exposing private information by timing web applications,” in Proc. 16th Int. Conf. World Wide Web (WWW), 2007, pp. 621–628.
- [14] S. A. Crosby, D. S. Wallach, and R. H. Riedi, “Opportunities and limits of remote timing attacks,” ACM Trans. Inf. Syst. Secur., vol. 12, no. 3, Art. 17, 2009.
- [15] OWASP Foundation, “Testing for account enumeration and guessable user account,” OWASP Web Security Testing Guide, v4.2.
- [16] C. Hagen, C. Weinert, C. Sendner, A. Dmitrienko, and T. Schneider, “All the numbers are US: Large-scale abuse of contact discovery in mobile messengers,” in Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS), 2021.
- [17] D. Kales, C. Rechberger, T. Schneider, M. Senker, and C. Weinert, “Mobile private contact discovery at scale,” in Proc. 28th USENIX Secur. Symp., 2019.
- [18] Twitter, Inc., “An incident impacting some accounts and private information on Twitter,” Twitter Privacy Center, Aug. 5, 2022.
- [19] H. Krawczyk, M. Bellare, and R. Canetti, “HMAC: Keyed-hashing for message authentication,” IETF RFC 2104, Feb. 1997.

University, Hamden, CT, USA, in 2026. Tauro holds the CompTIA Security+ certification and previously worked in information technology roles at the National Railways of Zimbabwe and at Zimbabwe’s Ministry of Lands, Agriculture, Fisheries, Water and Rural Development. Tauro designed, built, and operates Kibhi, a short-video and messaging platform for users aged 13 and over. Research interests include trust and safety engineering, child online safety, privacy, and application security.