STIR/SHAKEN: A Cocktail of Cryptographic Clumsiness

Joshua Brown
University of Michigan

Matthew Hardeman
Low Latency Communications

Paul Grubbs
University of Michigan

Abstract

Scams and fraudulent robocalls are rampant in the telephone system. Fraudsters often employ *call spoofing* to impersonate a known caller to the victim, making it difficult to detect and block the fraudster's calls. In an effort to combat call spoofing, the U.S. government in 2019 mandated the use of a new suite of protocols called STIR/SHAKEN.

In STIR/SHAKEN, providers include a cryptographically-signed assertion about the caller's identity and other metadata in call headers. This serves both to authentically convey this metadata between providers during call routing and to provide reportable, verifiable evidence of bad behavior by providers that originate fraudulent calls.

This paper asks whether STIR/SHAKEN achieves its security goals, and at what cost to the telephone system. On security we find serious structural weaknesses: a year of STIR/SHAKEN certificates observed in production call traffic shows a PKI riddled with noncompliance—nearly half of valid certificates violate the specifications—a provider can prevent later verification of a STIR/SHAKEN assertion by taking its certificate offline, and a header-malleability flaw, which we verified in test calls placed with the help of a real telephone provider, causes a spoofed caller ID to be displayed as verified on some phones. The cost, meanwhile, has been steep: STIR/SHAKEN deployment has created systemic new harms to the privacy of all phone users.

These new privacy harms take two forms. First, signing call metadata renders it cryptographically non-repudiable: for the first time in the phone system's 150-year history, there is publicly verifiable proof that a call took place. Second, STIR/SHAKEN's complexity pushes providers to outsource signing and verification—roughly half of the 29 providers we surveyed do so, most to a handful of dominant vendors—causing widespread new leakage of sensitive call metadata to off-path third parties. (One such off-path third party, Neustar, was acquired in 2021 by TransUnion, one of the three nationwide U.S. credit bureaus. Neustar authenticates over 30 billion calls per month—and TransUnion sells a "Phone Behavior Intelligence" product advertising "insight into billions of call transactions" obtained through its "deep carrier relationships.") These structural harms sit atop implementation failures: our tests also found implementations that completely break the federally-mandated *67 caller-privacy mechanism.

Finally, we propose solutions to the problems we identify. To fix the PKI, STIR/SHAKEN should adopt certificate transparency; a public certificate log also stops providers from taking certificates offline to defeat later verification. Our remaining solutions are cryptographic, and share a prerequisite STIR/SHAKEN cannot currently meet: the origin provider must learn the destination provider's public key. We thus design a backwards-compatible key discovery mechanism that uses SIP error messages. We then attack the privacy harms directly: replacing STIR/SHAKEN's signatures with asymmetric message franking (AMF) makes call metadata deniable again, and issuing AMFs *blindly* keeps it hidden from outsourced signers. We design the first blind signing protocol for the AMF construction of Tyagi et al., along with new security definitions capturing requirements novel to the setting, such as *verifier-hiding* blindness. To our knowledge, we thereby initiate the formal study of blind signing for designated-verifier signatures; we believe this is of independent interest.

1 Introduction

Fraudulent scams and robocalls are rampant in the telephone network. They cause immense financial losses to users, undermine trust in phone calls, and degrade the network's ability to deliver legitimate calls, as callees increasingly ignore calls from unknown numbers. According to the FCC, fraudulent calls cost Americans billions of dollars a year [23] and have persisted for decades, often targeting the most vulnerable

populations including immigrants, the elderly, and people with disabilities. In 2021 alone, over 60 million American consumers lost an estimated $30 billion. During the COVID-19 pandemic, contact tracing efforts were severely hampered by phone subscribers ignoring calls from unknown numbers, assuming the calls to be yet another scam [71].

After a series of failed attempts [29], [28], [34] to combat this problem, the U.S. government mandated the use by US-based carriers of a new suite of protocols called STIR/SHAKEN. In STIR/SHAKEN, when a carrier originates a phone call into the US telephone system, they include with it an attestation of their confidence that the calling party has the right to use the caller's phone number. Concretely, they do this by signing—with a certificate obtained from a STIR/SHAKEN-specific certificate authority—the call metadata, including caller and callee numbers. This signature is carried in-band in a custom extension field of the Session Initiation Protocol (SIP) message that sets up the call. By signing the call, the provider is staking their continued ability to participate in the phone system on the legitimacy of that call. Should the call be fraudulent, the signed metadata is reported to a "traceback group", who verifies it as part of tracing the call back to the originating provider which may give rise to financial liabilities or imperil their operating license. (Financial penalties have been given out as recently as this year [38].) According to the FCC's database, over 5,000 telecoms implement STIR/SHAKEN [36] and an estimated 42% of calls received use STIR/SHAKEN attestation as of April 2025 [80].

STIR/SHAKEN is, at its core, a system that uses cryptography for abuse detection and reporting. We observe that, despite its setting-specific features and motivation, it bears a remarkable similarity to systems like DomainKeys Identified Mail (DKIM), which provides sender authentication for emails, or even Facebook's message franking [33, 49], in which users report claimed abusive messages along with a verifiable cryptographic tag to a central authority for investigation.

Armed with this simple, but very consequential, observation, we initiate the formal academic study of the STIR/SHAKEN system. We seek to answer the following questions. First, what security and privacy properties does STIR/SHAKEN seek to provide for telecoms and for end-users of the phone system? Does it in fact provide those guarantees today? Are there other important security or privacy properties it lacks? If so, what are the consequences for STIR/SHAKEN and the phone system more broadly?

Our results. In this work, we employ a uniquely broad set of methodological tools to answer these questions. We begin by giving the first reasonably rigorous description of the threat model and security and privacy goals of STIR/SHAKEN (§3). Then, we ask whether STIR/SHAKEN meets its security and privacy goals in this threat model. The answer is no—we identify a number of issues with both the design of STIR/SHAKEN and its current implementation, including a not-previously-observed malleability issue with the signed metadata, the result of which is that in some cases, malicious callers *can still undetectably impersonate other parties*. We also continue to investigate an issue highlighted but not fully explored in prior work [17] that reported signatures can be made unverifiable to the traceback authority by simply preventing them from downloading the provider's public key. We perform a series of measurements of the STIR/SHAKEN ecosystem with the help of a real telecom provider, and find that exploiting these and other issues in STIR/SHAKEN is nearly trivial, and that a huge fraction of STIR/SHAKEN certificates—even CA certs—are malformed (§7). We also find that some providers fail to satisfy the sole privacy goal of STIR/SHAKEN, namely compliance with the federally-mandated *67 caller-anonymity mechanism (§7).

We next investigate whether STIR/SHAKEN *lacks* privacy properties that are considered important in other systems that use cryptography for abuse prevention. We identify two key issues in the way STIR/SHAKEN uses cryptography: first, signing call metadata renders it non-repudiable. This means that for the first time in the 150-year history of the phone system, there is publicly-verifiable cryptographic proof that a call was placed. Second, because of the complexity of managing STIR/SHAKEN identity keys, STIR/SHAKEN allows providers to outsource their STIR/SHAKEN signing and verifying to third parties. This means that STIR/SHAKEN creates new leakage of sensitive call metadata to off-path third parties.

To argue that these issues are practically as well as theoretically important, we conduct a survey (§6) of 29 providers. We find that data retention policies at telecoms suggest that publicly verifiable STIR/SHAKEN metadata is stored for long durations after a call has concluded, increasing the value of leaked logs. (Telecom providers are breached with some regularity [86].) Roughly half of the providers we surveyed use third-party providers, with a small number of providers (such as TransNexus and Neustar) providing a disproportionately large amount of third-party STIR/SHAKEN services. This means that a few companies are now in a

position to ingest a huge fraction of the metadata of *all* US phone calls. One of them, Neustar—which authenticates over 30 billion calls per month—is owned by TransUnion, one of the three nationwide U.S. credit bureaus; TransUnion markets a "Phone Behavior Intelligence" product built on "insight into billions of call transactions" [57]. Worse, the sensitivity of the ingested metadata is set to grow: the Rich Call Data extension to STIR/SHAKEN signs a caller-supplied "call reason," so an outsourced signer could learn not just that a subscriber received a call from a medical practice, but that the stated reason was, say, an upcoming oncology appointment. A system deployed to prevent one kind of abuse has thus built the infrastructure for another.

Finally, we give solutions to the problems we identified in STIR/SHAKEN (§8). We argue that STIR/SHAKEN absolutely requires a certificate transparency system like the one in use in the web PKI [56]. To facilitate improvements to the cryptographic parts of STIR/SHAKEN, we design a new backwards-compatible mechanism, based on SIP 603 error messages, for STIR/SHAKEN key discovery. Finally, we design and analyze a new cryptographic protocol for STIR/SHAKEN (§8.2). Our scheme is based on asymmetric message franking (AMF), a deniable signature primitive custom-built for abuse reporting settings. AMFs resolve the deniability issues with STIR/SHAKEN, but it is not known how to blindly issue AMFs; we need this to hide metadata from third-party signers. Filling this gap turns out to require new security definitions, a new protocol, and a delicate security analysis; we overview each in turn.

Blind asymmetric message franking. We design BAMF, the first blind signing protocol for AMFs. Surprisingly, in doing so we appear to have initiated the formal study of blind signing for *designated-verifier* signatures, a setting that raises definitional questions with no analogue in the blind-signature literature (§8.3). One is what blindness should mean: in our setting the signer must not learn *to whom* it signs—a third-party signer already knows the origin provider, so learning the receiver key would reveal both endpoint providers of a call, potentially harming privacy. We formalize this as *verifier-hiding blindness*, which strengthens standard blindness by letting the adversarial signer choose two candidate pairs of receiver and judge identity keys, not just two messages, between which it must distinguish. Another is what unforgeability should mean: designated-verifier signatures are verified with a *secret*, and AMFs have two designated verifiers—the receiver and the judge—with *different* verification procedures, so even stating one-more unforgeability (dvOMUNF) requires choosing which secret validates the adversary's forgeries; this is one of several points of genuine definitional leeway we identify and resolve. Finally, because verification itself needs the designated verifier's secret, a provider that outsources its key management needs the third party's help to verify—blindly. We identify this new primitive, *blind verification*, give a lightweight candidate protocol for AMFs, and leave its formal study to future work.

Our construction (§9) lifts blind Schnorr signing to the AMF setting. An AMF signature is a Fiat-Shamir proof of an AND of two OR clauses over four discrete-log statements, assembled with the CDS OR-proof technique [30]. The key observation is that the sender's secret enters exactly one of the four statements. We therefore run the standard blind Schnorr protocol for that statement alone, while the user completes the other three locally—one from an honest witness it samples itself (the judge clause) and two from simulated Schnorr transcripts, exactly as in ordinary AMF signing. What remains is to make the blinding commute with the proof composition: the user folds all commitments into a single Fiat-Shamir challenge and applies the CDS challenge split twice to that one challenge—once per OR clause—so that the blind-Schnorr blinding shifts the sender statement's share of the challenge without disturbing the clauses completed locally.

Finally, we analyze BAMF (§9.1). We prove that it satisfies *perfect* verifier-hiding blindness, even when the adversarial signer supplies malformed identity keys; because the signing interaction touches only the sender clause, the analysis is not much more involved than the perfect-blindness proof for plain blind Schnorr [42]. Unforgeability is another matter, even in the *sequential* setting we target, where the adversary cannot interleave signing sessions. We prove sequential dvOMUNF for algebraic adversaries by reduction to the one-more discrete logarithm problem, following the blueprint of Kastner, Loss, and Xu [54] for blind Schnorr. The heart of such a proof is a bad-event analysis: enumerate the algebraic constraints that the verification equations impose on the representations of the adversary's forgeries, then upper-bound the probability that the adversary satisfies all of them while denying the reduction a usable discrete logarithm. For blind Schnorr, the hardest case of this analysis reduces to the 1-ROS problem, whose constraints are linear and involve a single adversarial group element per forgery. Our analogue is substantially more complex: the constraints are quadratic, in many more unknowns—each forgery carries nine adversarial group elements, appearing in

six verification equations, along with two adversarially chosen CDS challenge splits—and they interact, since some group elements appear in *both* OR clauses, coupling the clauses' constraints, while some clauses admit degenerate assignments that yield trivial wins. To bound the bad event we show that, for each OR clause of a forgery, the constraints can be distilled into a small system of what we call *obstruction polynomials*, of which the forgery's random-oracle output must be a common root for the bad event to occur; counting roots completes the bound. In the course of this analysis we also uncovered a flaw in the original AMF scheme [85]: its verification fails to rule out a degenerate edge case—signature elements set to the group identity—that admits trivial forgeries and breaks a step of the original sender-binding proof. Two additional verification checks, which we adopt, repair it. We prove nothing about *concurrent* signing sessions, and BAMF should not be used with them; we leave concurrently secure blind AMF signing as an interesting open problem.

AI usage statement. We used AI tools in this work in a few ways. The AI tool Claude was used to edit human-written text throughout the manuscript. It also wrote initial drafts of Section 11 from human-written notes. Claude and ChatGPT were used to draft, refine, patch, and revise the security analyses of BAMF. The authors of this work take responsibility for all its contents and claims.

2 Background & Preliminaries

2.1 The Telephone System

The telephone system is a synchronous voice-based communication system. Calls placed by end-users are routed through a network of providers (i.e., telephone companies) often traversing multiple intermediary providers before arriving at the recipient. There are over 5,000 providers in the United States alone [36]. The network they collectively operate is known as the Public Switched Telephone Network (PSTN). New PSTN features have a unique design constraint: they must remain compatible with devices and infrastructure that can be over a century old. In the United States, the Federal Communications Commission (FCC) regulates the PSTN.

In the PSTN, calls happen in two phases. First, a *signalling* phase creates the necessary connection between the caller and recipient through their respective providers. After signalling, the voice data is sent in the *media transmission* phase. Our work only deals with signalling, so we will not refer to media transmission further.

The PSTN contains both legacy telephony and modern IP telephony infrastructure. Legacy infrastructure uses signalling System 7 (SS7) for signalling over time-division multiplexing (TDM) [19]. IP telephony infrastructure uses Voice-over-IP (VoIP) on the (IP-based) internet and performs signalling with the Session Initiation Protocol (SIP) [69]. A single call can traverse both IP and legacy infrastructure during routing. Protocol translation (SIP $\leftrightarrow$ SS7) occurs at each such boundary. Unlike in the modern internet, where HTTPS is nearly ubiquitous, calls in the PSTN are almost never encrypted in transit, but direct peering between VoIP providers means that VoIP calls do not necessarily traverse the public internet during signalling [9].

Session Initiation Protocol (SIP). The first signalling message sent in the text-based SIP protocol is the `INVITE` message. Its first line, called the request line, specifies the destination location of the call. Similarly, the `Contact` header specifies the caller's location, where SIP responses should be sent. The `From` and `To` header specifies the caller and callee's identity, respectively [69]. (The separation between identity and location in SIP is analogous to the distinction between domain names and IP addresses on the internet.)

Caller ID information, introduced in the early 1980s [1], is transmitted by the origin provider generally in the `From` header [69]. Caller ID provides no means of authenticating the accuracy of the transmitted number and can be trivially spoofed by the calling party's provider. Fraudsters exploit this vulnerability by spoofing caller IDs to impersonate trusted entities (e.g. banks, government agencies, and local business), thereby increasing the likelihood that recipients will answer their call. This lack of authentication is a major enabler of fraudulent activity on the telephone network [4].

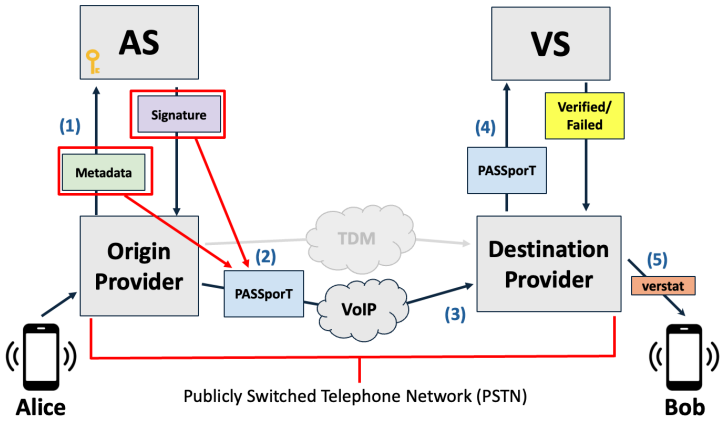

Figure 1: STIR/SHAKEN Architecture

3 STIR/SHAKEN Overview

STIR/SHAKEN is a system with two goals: (1) preventing telephone fraud and (2) providing a trustworthy caller identity to the call recipient. To achieve these goals, STIR/SHAKEN allows providers to make cryptographically verifiable claims about the origin of a call, the service provider's relationship with the caller, and the identity of the caller, while staking their ability to operate on the veracity of such claims. Should the call be reported and later determined to be fraudulent by investigators, the government can easily and provably identify the provider that made a claim about the call's provenance and hold them accountable through legal fines or license revocation. STIR/SHAKEN is currently deployed independently in the United States and Canada, and is being considered for further deployment in other countries [31].

STIR/SHAKEN has two main standardization bodies. The underlying protocols, applicable to any VoIP context, are developed by IETF's STIR working group [51]. ATIS, a consortium of telecommunications companies, adapts the IETF's protocols for use in the PSTN.

Entities in STIR/SHAKEN. In the United States, the STIR/SHAKEN ecosystem is governed by the **Federal Communications Commission** (FCC). The FCC works with an industry group called the **governance authority** (GA) to set STIR/SHAKEN policies and security requirements around the issuance and use of SHAKEN certificates. The **policy administrator** (PA) implements policies from the FCC and GA and is responsible for approving providers and **certificate authorities** (CAs) for participation in STIR/SHAKEN. The PA also manages and publishes a list of approved CAs, forming the STIR/SHAKEN root of trust, and a centralized **certificate revocation list** (CRLs) for the national SHAKEN PKI [8]. CAs issue certificates to approved telecoms. **Providers** use these certificates to sign claims about the call metadata, producing a **PASSporT**. **Authentication services** (AS, top-left of Figure 1) manage the provider's keys and perform the actual signing procedure. **Verification services** (VS, top-right of Figure 1) handle verification of PASSporTs and provide a success or failure indicator to providers. **End-users** place and receive phone calls and are recipients of caller identity information informed by STIR/SHAKEN attestations in the PASSporT. The **industry traceback group** (ITG) is a functional role whose holder is selected by (and assigned authority by) the FCC to investigate claims of illegal calls.

The STIR/SHAKEN protocol. See Figure 1 for a depiction of a typical STIR/SHAKEN-enabled call flow. STIR/SHAKEN operates over VoIP based SIP. When the end-user Alice places a call to Bob, the *origin provider*—her provider in the figure—requests, from the AS, a signature over the call metadata and an attestation level (label 1 in the figure). The origin provider uses the signature returned from the AS to form a JSON Web Token (JWT), called a PASSporT in STIR/SHAKEN parlance, and inserts it into the `Identity` header in the `INVITE` message (label 2). The call metadata in the PASSporT's payload includes, at a minimum, the origin telephone number, the destination telephone number, and a timestamp of the call but can also be expanded to include a variety of other information about the caller and the reason for the

call [20]. There are three levels of attestation in STIR/SHAKEN, A, B, and C, indicating a progressively weaker relationship between the origin provider and the caller [40], [74]. The PASSporT also includes a URL in the `x5u` header that is used to download the certificate required to verify the PASSporT. PASSporTs are always signed using ECDSA with P-256 and SHA-256 (i.e., `alg` is ES256 in the JWT standard). PASSporTs are valid for a limited time period, defined by local (the validator's) policy [40]. The IETF recommendation for PASSporT freshness is 60 seconds [63].

The call is routed through the PSTN arriving at the *destination provider*—Bob's provider in the figure (label 3). The destination provider extracts the PASSporT from the `INVITE`, ensures that the caller and callee identity in the PASSporT matches the `INVITE` headers, and forwards it to the VS for verification (label 4). The VS fetches the certificate from the `x5u` URL, verifies the signature on the PASSporT, and returns a success or failure indicator to the destination provider. The destination provider may send a `verstat` header with the verification result and the other PASSporT claims to Bob's device for display in the UI (label 5 in Figure 1).

Unless the call is reported, STIR/SHAKEN stops here. Various jurisdictions and providers have different mechanisms for both end-users and destination providers to report fraudulent calls. If Alice's call is reported, a manual "traceback" process might be initiated by the ITG who works with providers to determine whether a call is actually fraudulent, and if so, who was responsible for letting the call be placed [47]. The traceback process is where the evidentiary aspect of STIR/SHAKEN comes into play: ITG uses signed PASSporTs, which is verified as part of investigation, to identify the origin provider [22]. ITG investigations often involve other information, which is not relevant for this paper.

STIR/SHAKEN providers can either use in-house ASs and VSs or delegate these roles to off-premise third parties. Third-party ASs & VSs are widespread in practice (§6). An FCC report [10, 48] indicates that multiple third-party AS patterns are increasingly utilized by small carriers to fulfill their STIR/SHAKEN obligations. They coin the terms "Hosted SHAKEN", to reflect a third-party signing service invoked by the originating carrier, and "Carrier SHAKEN", which describes an arrangement in which a downline carrier possesses the private key of the originating carrier and signs calls on their behalf. The downline carrier may optionally use a further (off-path) third party to perform the actual signing operations. The report acknowledges similar setups for verification.

The FCC has discussed extending STIR/SHAKEN to SMS messages [26], referencing an RFC that proposes including message hashes in PASSporTs [65].

STIR/SHAKEN over legacy infrastructure. STIR/SHAKEN only operates over VoIP based SIP as legacy TDM infrastructure *cannot* carry SIP headers, and in particular PASSporTs. Changing TDM protocols and hardware is not feasible. Currently, when a call traverses an IP-legacy boundary PASSporTs are silently dropped. Based on industry statistics [81], these drops are surprisingly frequent and the primary reason some calls are received without STIR/SHAKEN.

To address compatibility with legacy infrastructure, the FCC is considering enforcing variants of two different solutions: in-band and out-of-band [11].

The in-band solution relies on unused data fields within SS7 to transmit the attestation level unauthenticated between TDM sections of the call path [6]. Due to the agreements required by individual providers to ensure trust and compatibility and the changes required to aspects of the VoIP/legacy border infrastructure, we do not believe that this solution is likely to be widely adopted [12].

The other proposed solution is called **Out-of-Band STIR/SHAKEN** (OOBSS) [19]. Before the call enters a legacy section of the call path, an authenticated provider uploads the PASSporT, via HTTPS, to a PA approved third party "drop-box" called a **Call Placement Service** (CPS). The PASSporT is indexed by the origin and destination phone number. At some point after the call traverses the legacy section of the call path, a SIP supported provider queries the CPS for the PASSporT using the origin and destination phone numbers transmitted in the signalling data. The provider then restores the fetched PASSporT into the call's SIP headers. To maximize the likelihood that the PASSporT can be recovered, the ATIS-1000096 standard dictates PASSporTs should be fanned out to *every* authorized national CPS [19]. While this solution is not yet widely used, there is at least one deployed production implementation of a redundant CPS service mesh by TransNexus [79]. Since the FCC will likely soon mandate the use of a TDM compatibility solution, it seems likely that adoption of some version of OOBSS will increase.

3.1 STIR/SHAKEN Threat Model

No detailed threat model has ever been given for STIR/SHAKEN so we present what we've pieced together from industry documents, using intuition and common sense to fill in gaps where needed.

The GA and PA are assumed to be fully trusted—in particular, the PA must correctly vet providers and CAs to participate in the STIR/SHAKEN ecosystem. (If the PA maliciously grants admission to a rogue CA, it can impersonate any provider and forge fraudulent attestations.) The ITG is also assumed to carry out their investigations honestly—they do not collude with any provider to fabricate evidence that implicates another provider, nor do they maliciously *ignore* evidence of misbehavior.

STIR/SHAKEN also relies on all CAs to act honestly. This means the CAs correctly verify that a provider requesting a certificate for a certain identity, in fact has the right—granted by the PA—to represent that identity in STIR/SHAKEN. CAs must also issue well-formed (per the STIR/SHAKEN specifications) certificates. Even more so than the PA, STIR/SHAKEN relies on CAs to act honestly and not be compromised—even a single malicious CA can undermine the authenticity of the entire system, by issuing certificates for any identity.

The originating provider may attempt to circumvent the STIR/SHAKEN reporting process by denying that they signed a call or by framing another provider. To do this, the originating provider may collude with other providers or the end-user caller. However, the origin provider is assumed not to forge calls or impersonate their own customers (i.e., sign records of a call that never happened). In the case of a third-party AS, the origin provider is assumed not to hold its own private signing key but may attempt to recover it during interactions with the AS.

The AS is assumed to correctly authenticate the origin provider before properly signing requests with the intended key. It is also assumed not to reveal its signing key to any party, including the origin provider (except where these parties are the same entity). The AS does not forge PASSporTs. The AS can see all call metadata included in the origin provider's request.

The VS is assumed to correctly verify PASSporTs as standardized and to accurately convey the verification result to the destination provider. The VS can see all call metadata included in the destination provider's request.

The destination provider is trusted to faithfully evaluate the results of the VS's response and convey relevant identity information and/or confidence levels to the callee. The destination provider is also trusted to respect caller ID privacy requests. In the United States, callers can request privacy by prepending `*67` to a dialed number. Their provider inserts a `Privacy` header in the `INVITE` message to indicate this request. When a destination provider receives a call with this header, it is required to not transmit any identifying information about the caller to the callee, even when STIR/SHAKEN is used [35, 2, 3].

The threat model permits the use of third-party ASs and/or VSs. For our model, we define "third-party AS/VS" as any AS/VS operated by a party who would not otherwise be exposed to the PASSporT or call metadata if not for their role in STIR/SHAKEN. In other words, third-party ASs/VSs are not naturally on the call path outside of STIR/SHAKEN. The threat model explicitly allows for AS/VS collusion, which is necessary because this occurs in practice when the origin provider's third-party AS and the destination provider's third-party VS are operated by the same company.

The destination provider or VS may report evidence of call fraud to the ITG when requested by a user report or detected by call analytics. But the destination provider can attempt to generate a fake PASSporT, in collusion with other providers, to frame an honest provider for misbehavior.

Intermediary providers have access to all call metadata, including the PASSporT. They are trusted to route the call correctly, but may try to modify call metadata, add or remove headers, or drop the PASSporT entirely. While intermediary providers are allowed to retain PASSporTs for an indefinite period of time and may even leak it to third parties, they may not replay PASSporTs to place their own fraudulent calls or leak the PASSporT to parties that will. This is a requirement that stems from routing mechanisms in the PSTN.

End-users are not assumed to act honestly. They may collude with the origin providers to place fraudulent calls or to lie about their identity or any other claims made in the PASSporT. In this case, STIR/SHAKEN seeks to guarantee that the destination provider has reportable evidence that this collusion occurred.

4 Flaws in STIR/SHAKEN

We evaluated the security and privacy of STIR/SHAKEN against the design goals and trust assumptions described in §3.1. Our analysis revealed a range of issues, some in the design of STIR/SHAKEN and some in how it has been deployed. We group them into three: flaws in the entirely new PKI that STIR/SHAKEN establishes (§4.1), a failure to bind the caller identity displayed to the callee to the identity that was actually validated (§4.2), and failures to honor caller anonymity requests (§4.3).

4.1 PKI Flaws

Many of the issues we identify stem from the entirely new PKI established by STIR/SHAKEN. This is where STIR/SHAKEN's evidentiary guarantee ultimately rests: a PASSporT is useful as evidence only if the certificate that validates it remains available, can be invalidated when its holder is expelled from the ecosystem, and was correctly issued in the first place. We show that the first two of these properties fail by design, and the third fails in practice.

Post-call traceability. A central design goal of STIR/SHAKEN is post-call non-repudiation: the origin provider must not be able to later deny having attested to the claims it made about a call. Each PASSporT contains a URL in the `x5u` header pointing to the certificate used to verify it; this URL can be hosted by any entity, including the originating telecom (§7). Adei et al. [17] observed in passing that a malicious provider can frustrate traceback by signing PASSporTs whose `x5u` URL either never resolves or resolves only after a long delay. We observe that `x5u` abuse can lead to a qualitatively different and more severe problem: a malicious provider can serve its certificate normally while the call is placed and verified, then simply remove it (i.e., return a 404) afterward. Though the PASSporT remains available, its signature can no longer be verified. While an intermediary or the destination provider may cache the certificate, this is not required behavior, and the cache is unlikely to survive if the call is investigated long after certificate expiration. The origin provider thus *retroactively* repudiates its attestation at no real-time cost—defeating exactly the evidentiary guarantee STIR/SHAKEN exists to provide. A useful analogy is to the "sender-binding" goal of message franking, described in [49], which requires that a malicious sender cannot prevent reporting of a message the receiver has accepted. Taking down the certificate is thus a kind of attack on STIR/SHAKEN's sender binding. Our measurements (§7) show that a significant fraction of URLs in observed PASSporTs already fail to return the signing certificate, or return a different certificate than the one needed to verify a PASSporT—so a deliberate takedown would currently be hard to distinguish from the retrieval failures already endemic in the STIR/SHAKEN PKI.

Certificate revocation failures. STIR/SHAKEN relies on a centralized CRL managed by the PA and polled by verifiers approximately every 24 hours [40], [27], [75]. Certificates may be revoked for several reasons: by request of the originating telecom, by request of the issuing certificate authority (CA) directed to and approved by the PA, upon action of the GA, or following regulatory action by the FCC.

The latter case is especially critical to the primary design goal of STIR/SHAKEN: when a telecom is removed from the STIR/SHAKEN ecosystem for misconduct, they must no longer be able to generate valid PASSporTs. To enforce this, the PA must revoke all certificates issued to that entity prior to their expulsion. However, STIR/SHAKEN lacks a standardized mechanism for the PA to definitively determine what certificates exist and were issued to a specific telecom. The PA does not maintain a list of issued certificates. Instead, when a telecom is removed from the network, the PA must conduct an ad hoc, manual collection effort by contacting each CA and requesting a list of certificates issued to the telecom [76].

This process is inherently fragile. The PA has no means to verify the completeness of the list returned by the CAs. If a CA fails to respond, responds incompletely, or responds with delay, the PA may fail to revoke one or more valid certificates associated with the offending telecom. As a result, certificate revocation in STIR/SHAKEN is not only error-prone and fundamentally unreliable, but also may be extremely time-consuming.

Certificate mis-issuance & retrieval errors. Through our empirical analysis of STIR/SHAKEN certificates (§7), we found widespread certificate mis-issuance and certificate retrieval problems in the STIR/SHAKEN PKI. Of the 9568 unique PASSporT URLs, 36.34% violated the STIR/SHAKEN specifications, including a staggering 89.27% of URLs hosted by providers. A number of these URLs returned a 404, an empty

response, timed out, or returned an entirely different certificate than the one used to sign the PASSporT. Of the 6960 unique and valid telecom certificates, 46.34% contained at least one violation of the STIR/SHAKEN specifications. Of the 42 unique and valid CA certificates, 45.24% contained at least one violation of the STIR/SHAKEN specifications.

4.2 Spoofed Caller ID

STIR/SHAKEN's authenticity guarantee reaches the callee only if the caller identity *displayed* to them is the identity that was validated. The threat model of §3.1 trusts the destination provider to convey identity information and confidence levels to the callee, but neither it nor the STIR/SHAKEN standards require that the displayed identity is the one STIR/SHAKEN validated. We show that this gap admits an attack in which a spoofed caller ID is displayed alongside a positive verification indicator.

A SIP `INVITE` may carry the calling party's number in more than one header, most importantly `From` and `P-Asserted-Identity`. When the two disagree, the specifications require the PASSporT to be validated against `P-Asserted-Identity` [40]. A malicious caller can exploit this as follows.

1. The caller places a call through its own provider, populating `P-Asserted-Identity` with a number it is genuinely authorized to use and `From` with the number it wishes to impersonate.

2. The origin provider signs a PASSporT over the `P-Asserted-Identity` value. Because that number really does belong to the caller, the provider can attest to it at the highest level; it does not check the `From` header or its binding to the signed number.

3. The call traverses the PSTN and reaches the destination provider, whose VS verifies the PASSporT successfully. The destination provider sets `verstat` accordingly and forwards both identity headers—but not the PASSporT itself—to the callee's device.

4. The device must choose which header to display as the caller ID. If it chooses `From`, the callee sees the impersonated number next to an indicator reporting successful STIR/SHAKEN validation.

Nothing in this attack requires a dishonest provider or a forged signature: every cryptographic check succeeds, and the attestation is truthful about the number it covers. The failure is that the number it covers is not the number the callee reads.

During testing we placed a call with disparate numbers in the `From` and `P-Asserted-Identity` headers, and the destination provider behaved as described above: it presented the `P-Asserted-Identity` field with `verstat` parameters indicating successful STIR/SHAKEN verification, and also presented the differing `From` number. On the enterprise SIP handsets we tested, the `From` number is what the user is shown, and the documentation of major IP phone vendors likewise describes `From` as the default display field [90, 44, 25]. We did not test major mobile handsets, which make these display decisions in baseband firmware we could not instrument, so we cannot say whether the attack reaches them. In accordance with responsible disclosure practices, we reported this issue to the affected vendors.

Two changes would close the gap. First, STIR/SHAKEN specifications should require that the field displayed as caller ID be exactly the field validated against the PASSporT. Second, if a device or destination provider is unable to establish that the caller ID was validated against a PASSporT, it must not show a positive verification.

4.3 Breaking Caller Anonymity

Caller anonymity is the one privacy protection STIR/SHAKEN is obliged to respect: the threat model of §3.1 trusts the destination provider to honor caller-ID privacy requests, and federal regulation requires it [35].

When a call where the caller has requested privacy is received by the callee's telecom, the receiving telecom is legally required to withhold all caller identity information, most notably the caller's phone number, from the callee.

STIR/SHAKEN introduces a novel vector that can inadvertently leak caller information to the callee. Specifically, to indicate the authentication result of the PASSporT, the destination provider may forward the `verstat` parameter alongside the authenticated information which likely includes the caller's phone number.

In our testing, we found that Vitelity, a large provider of consumer and business-to-business VoIP services, is leaking the caller's phone number (alongside the `verstat` parameter) in the `P-Asserted-Identity` header even when the caller has requested anonymity. This header was only included in the SIP `INVITE` message when STIR/SHAKEN validation was requested in the provider's configuration panel. Furthermore, as detailed in Section 5, RCD introduces even more sensitive metadata that may inadvertently be leaked to the callee.

5 Privacy In STIR/SHAKEN

Other than anonymous calling (described in §3.1), that STIR/SHAKEN inherits from the broader telephone system, STIR/SHAKEN lacks any significant privacy goals. We've identified a number of alarming privacy issues that severely affect the privacy of everyday users of the telephone network. STIR/SHAKEN introduces new, severe privacy risks to users that did not exist prior to its deployment. We therefore believe that STIR/SHAKEN must address these privacy issues, and we present solutions to them in §8.

Metadata leakage. STIR/SHAKEN stores a tremendous amount of call metadata in PASSporTs. At a minimum, PASSporTs necessarily include the caller and callee phone numbers. Features like Rich Call Data (RCD) drastically expand the amount and type of information that can be embedded in the PASSporT to include names, addresses, photos, and other identifiable information about the caller. RCD also allows the caller to indicate the purpose of the call, alluding to the call content, a feature already offered by a number of providers today [78, 84, 82]. In fact, in the example presented in the STIR/SHAKEN specification the call purpose indicates that the call is from a medical professional reminding the patient of an upcoming appointment [20]. Needless to say, data stored in PASSporTs can be highly sensitive and, since recipients use this information to decide whether to answer a call, there is a strong incentive for callers to include as much information as possible.

Unfortunately, STIR/SHAKEN also leaks the PASSporT, including all of this metadata, to both on and off path third parties. PASSporTs are sent in the clear as part of the SIP `INVITE` message and are thus accessible to all intermediary providers. If the origin or destination provider uses a third party AS or VS, respectively, then the PASSporT is also sent to these off-path third parties during the authentication and verification procedures. Based on our survey (§6), nearly half of telecoms use off-prem third parties for their authentication and verification services. Furthermore, if OOBSS is utilized, the PASSporT will be sent to at least one CPS but might be sent to the entire national network of CPSs.

SIP `INVITE` headers, which include the PASSporT, are also stored and logged by many telecoms for weeks or even up to a year after a call has concluded. Based on our survey data, over 80% of telecoms store some call metadata after a call has concluded with many explicitly indicating SIP `INVITE` headers as part of the stored metadata. Additionally, an estimated 20% of telecoms store some call metadata indefinitely. By significantly expanding the volume and type of metadata previously stored in call logs, STIR/SHAKEN greatly increases the value of these logs to attackers.

This is far from a theoretical risk. The telecom industry has a long history of data breaches and leaks. For example, in 2024, hackers stole 3 terabytes of call records from a South Korean telecom provider and targeted a number of other telecoms in a variety of countries as well [73]. More recently, earlier this year, Chinese hackers were reported to have compromised at least 9 different U.S. carriers, accessing call records among other data [86].

The caller/callee pair is also leaked due to the indexing scheme used in OOBSS. Since OOBSS is still in development and consideration, we leave solutions to this problem for future work.

We therefore believe that **metadata confidentiality** between the originating provider and the destination provider must be a critical privacy goal of STIR/SHAKEN. Neither intermediaries nor third-party authentication and verification services should have access to the metadata in PASSporTs. In addition, given the limited number of companies offering third-party STIR/SHAKEN services in this ecosystem (see §6), it is possible that the authentication service and the verification service are operated by the same company. For additional protection, we consider the case where the third-party authentication service and the destination provider collude (see §3.1). A call signed by the authentication service should not be **linkable** to the call verification performed by the verification service even if these two parties collude. We, however, do not consider network traffic analysis to be in scope for this work.

Deniability. STIR/SHAKEN introduces a significant deniability problem to the telephone system. STIR/SHAKEN PASSporTs are cryptographically signed, non-repudiable, and publicly-verifiable, providing proof that a particular call was placed with the associated metadata. Anyone in possession of a PASSporT can convince any third party that the particular call was placed. For the first time in the history of the telephone system, calls contain publicly-verifiable cryptographic evidence of the call.

And, as explored earlier in the section, a number of different on and off path third-parties have access to a PASSporT for a given call. PASSporTs also contain a significant amount of metadata beyond just the caller and callee phone numbers. If RCD contains "call reason", STIR/SHAKEN will even produce cryptographic proof that a call was placed for a specific purpose.

Furthermore, the deniability problem in STIR/SHAKEN is exacerbated by the data retention policies of telecoms. The inclusion of cryptographic proof of call data fundamentally changes the value of call records. Attackers who steal call records including PASSporTs can publicly prove the authenticity of the stolen call data.

The deniability problem in STIR/SHAKEN is worth comparing and contrasting with another widely-recognized deniability issue: DKIM signatures for email. Similar to STIR/SHAKEN, DKIM is a system that exists to prevent spam and email spoofing. DKIM, however, involves signing email headers and contents, instead of just metadata. Since DKIM signatures allow anyone to verify an email was sent, they have historically been utilized to verify leaked or hacked information. One of the most notable examples was in 2020 when Wikileaks used DKIM signatures to authenticate Hunter Biden's hacked emails [45], [89]. While PASSporTs only include signatures over call metadata, not call content (other than call purpose in the case of RCD), their existence is similarly cryptographic evidence that the originating caller placed a call to a destination callee with the associated metadata. That said, if STIR/SHAKEN is extended to SMS as proposed, then the deniability problem will be even more severe as PASSporT signatures would be bound to message content as well.

STIR/SHAKEN must achieve **deniability**, ensuring that only the destination provider for the particular call and the investigators can verify PASSporTs are genuine.

An example worth highlighting. To illustrate the impact and severity of privacy issues in STIR/SHAKEN, we draw readers to the example of Neustar. Neustar is not a telecom. Neustar does, however, operate one of the largest third-party STIR/SHAKEN services in the U.S. authenticating over 30 billion calls per month and providing STIR/SHAKEN solutions to over 100 carriers as of 2021 [62]. That year, Neustar was acquired by TransUnion, one of the three nationwide credit bureaus in the U.S. [83].

Under the copyrighted motto "Information for Good", TransUnion and Neustar together offer a product they call "Phone Behavior Intelligence" which is advertised as offering "insight into billions of call transactions" through their "deep carrier relationships" [57]. In their whitepaper, Neustar boasts the granularity of their analytics, offering detailed data on the "best time of day and week to call [individual] consumers" based on their individualized "phone usage" patterns. In their own words, " if a consumer consistently uses his phone Tuesdays between 10 a.m. and 12 p.m., he may be more likely to respond to a text sent to his primary phone." Neustar also provides internal statistics comparing their determination of "contactability score" to actual customer responses rates [61]. We leave it to the reader to draw their own conclusions from this example.

6 Survey

To better understand the telephone ecosystem and how STIR/SHAKEN is deployed, we conducted a qualitative and quantitative survey of 29 telecoms. In this section, we describe our survey methodology, our results, and conclusions. Most importantly, our survey results revealed some unique qualities of the telecommunication system that have informed our construction design and threat model. Our full methodology and further results are described in Appendix A.

Methodology. The survey was taken between February and March 2024 via phone. Contact information came from an FCC database. Telecoms self-reported their size relative to the industry with 81% small, 15% medium, and 4% large (of the 26 respondents who reported a size). We drew our sample from the 2,796 U.S. telecoms listed in the database as having a complete STIR/SHAKEN implementation, of which we contacted

approximately 331 and received 29 responses. We acknowledge a bias towards small providers and the limited response rate due to the survey being conducted via phone and the voluntary nature of participation. While our survey is not statistically rigorous, further discussions with industry experts suggests our survey is likely representative of industry overall.

6.1 Results

STIR/SHAKEN implementations. To understand how STIR/SHAKEN is used in the wild, we asked a series of questions about telecoms' STIR/SHAKEN implementation. We began by asking whether telecoms use a third-party or on-premises (on-prem) STIR/SHAKEN implementation for signing and verifying PASSporTs. If the respondent indicated they used an on-prem implementation, we asked whether they built their implementation in-house or purchased it from a vendor. If the respondent did not build their own implementation, we asked which vendor they purchased their on-prem solution from or use as an off-premise service. Our survey revealed that a plurality, nearly half (46.4%), of telecoms use off-prem third party STIR/SHAKEN implementations. A surprisingly large number of telecoms, 21.4%, reported that they had built or were planning to build their own STIR/SHAKEN implementation. 17.9% of telecoms reported using an on-prem implementation purchased from a vendor. The remaining participants did not know or provided an ambiguous answer. Respondents indicated they use a variety of vendors for third party (off and on-premise solutions) the most popular of which was TransNexus: of the 16 respondents who named a specific vendor, 5 (31%) named TransNexus or its ClearIP product.

Data retention policies. As explained in our introduction, there have been many reports of telecoms being compromised and their data being stolen. We wanted to understand what data about calls telecoms typically retain and for how long after the call has concluded. Overall, telecoms collect a variety of information for billing, regulatory, and fraud detection purposes.

An extremely large majority 83.3% of telecoms indicated that they retain call metadata for sometime after the call has concluded. Additionally, 20.8% indicated that they retain some call metadata indefinitely. When asked about the specific types of data and duration of their retention, telecoms reported a variety of policies. Among the telecoms that specifically described retaining SIP headers, the reported durations ranged from 7 days to a year, and other basic metadata such as the caller and callee numbers, call timestamp, and call duration was retained anywhere from 30 days to 10 years. Most of the indefinite data retention was of billing records but sometimes included SIP information.

Disproportionate burden on smaller telecoms. We asked telecoms about their experiences with STIR/SHAKEN. Overall, 56% of respondents agreed or strongly agreed that STIR/SHAKEN is a burden on their operations, compared to 16% who disagreed or strongly disagreed. Among small telecoms specifically, 55% called it a burden and 15% did not. In open responses, many cited the high cost of third-party solutions and the disproportionate compliance burden on smaller providers. One small carrier claimed, "big companies" use STIR/SHAKEN to "squash small companies." Another telecom said the cost of proper STIR/SHAKEN implementations makes it difficult for good actors to keep up with the bad ones, who "always see a way out of it." A third carrier called third-party certificate and signing services "ridiculously priced." A medium-sized carrier reported that STIR/SHAKEN costs 50% of their [revenue] and warned that it will "kill all the small companies."

Perceived effectiveness of STIR/SHAKEN. Opinions on STIR/SHAKEN's effectiveness were mixed, and negative on balance. 37.5% of telecoms agreed or strongly agreed that STIR/SHAKEN reduces robocalls, compared to 45.8% who disagreed or strongly disagreed. The remaining 16.7% were neutral.

Many respondents criticized false spam labeling caused by large carriers' poor STIR/SHAKEN implementations, particularly affecting calls from small carriers. One company stated explicitly that STIR/SHAKEN "doesn't help at all", another labeled it "a waste", and one exclaimed "I hate STIR/SHAKEN." According to one company's internal statistics, "it has stopped zero calls" with robocallers adapting to "more creative" tactics. Others were cautiously positive, calling STIR/SHAKEN a "very effective tool if utilized properly" and another noting that it "seems to be working." Some believed it curbs "80% of [the] lowest level spam" but that the remaining use "more sophisticated" methods.

Inquiries by fraudsters. Several telecoms reported receiving inquiries from bad actors seeking to route

calls through their legitimate networks to obtain valid STIR/SHAKEN attestations. While only 17.4% of respondents reported such inquiries, those that did described large volumes—one received "hundreds" in 2023 and "dozens in the last 90 days." Telecoms noted frequent emails and calls, often from abroad, and described using onboarding checks and call monitoring to detect fraudulent customers. Telecoms described annoyance with the inquiries.

7 Measurements & Tests

Certificate analytics. We conducted an analysis on 9728 `x5u` URLs which were observed in SHAKEN PASSporTs received by numerous production telecoms. Those telecoms contributed the `x5u` URLs between October 2023 and October 2024 to a SHAKEN ecosystem data repository. All analytics were performed on a GitHub Actions compute unit on June 2nd, 2025.

Of the 9728 unique URLs, 8617 returned a certificate, with the rest returning a 404, an empty or non-certificate response, or timing out.

STIR/SHAKEN specifications place requirements on how certificate hosting endpoints should behave (such as the specific headers they must return) [40], [8]. We assessed compliance with the standards using ZLint with a custom rule set of 8 lints. Of the 9728 URLs, 160 were skipped as duplicates, leaving 9568 assessed against the rule set, of which 36.34% contained one or more violations of the relevant specifications. We also compared the 494 unique URLs hosted by providers with the 9074 hosted by CAs and found that 89.27% and 33.46% violated the specifications, respectively. The average response time for all of the endpoints was 131ms.

We analyzed and assessed compliance of the certificates themselves as well. The 9728 URLs returned 8457 unique telecom certificates signed by 47 unique CA certificates. Of the unique telecom certificates, 1479 were not valid during the collection period. Upon examination of some of the out-of-validity certificates, some of these certificates have apparent issuance dates subsequent to the end of the data collection period. This indicates that some `x5u` URLs are being reused to serve subsequent SHAKEN certificates. Further, 18 telecom certificates and 5 CA certificates did not have a valid trust chain.

We excluded these certificates from further analysis resulting in 6960 trusted telecom certificates and 42 trusted CA certificates remaining in our testing set. The average validity duration of telecom certificates and CA certificates was 64 days and 5590 days, respectively. Among the telecom certificates, 678 unique telecoms were represented with an average of 12.71 certificates per telecom. Additionally, against 33 lints, the telecom and CA certificates were found to violate relevant specifications 46.34% and 45.24% of the time, respectively.

Two particularly pernicious error types were observed in the examined certificates: errors in the coding of the data structure of the TNAuthList [64] and failures to populate the TNAuthList with correct and standardized data [8]. We observed that one CA had at least two certificates in force during the period examined which had data coding errors as to the TNAuthList. Two other CAs violated the data content standards of the TNAuthList SPC data field by including lower-case letters. The TNAuthList is the single most significant subject identifying field in a SHAKEN certificate. It is intended to be an exact match to industry databases indexing telecommunications service providers. Errors in this field could plausibly lead to ambiguities or inability of relying parties to definitively associate a certificate with its corresponding service provider.

Response to non-conforming PASSporTs. SIP response code 603 has been widely utilized to indicate rejection of a call for numerous causes. We test to examine whether such call rejection responses could be a useful transport for destination key discovery. With the assistance of a telecom, a number of test calls were placed with PASSporTs containing additional data elements in the passport alongside an incorrect (and unroutable) destination telephone number. Additionally, a number of calls were placed with otherwise valid passports signed by a key not-corresponding with the SHAKEN certificate subject public key. We wanted to test whether telecoms commonly return SIP 603 error responses or permit such calls to proceed with non-conforming PASSporTs or PASSporTs with invalid signatures. We also wished to determine whether a newly described standard for SIP "603+" call rejections would provide a reliable reverse-transport for significant amounts of arbitrary data from the terminating carrier back to the originating carrier in the

event of a rejected call. We placed calls with such PASSporTs toward 6 large telecoms, receiving 603 error responses from none. In none of these cases did the terminating carriers assess the calls as validated, but they also did not outright reject the calls.

Seeking 603+. Because none of the large carriers targeted rejected the calls, we conducted further test calls in which the telecom assisting us placed calls to telephone numbers which would hairpin back to a different voice switch of the same telecom after having traversed an indeterminate number of intermediate carriers. These calls were placed with a chosen set of 3 first-intermediate hops toward the destination. The test system at the telecom was configured to assess the STIR/SHAKEN PASSporTs received with these calls. If the validation result was invalid or mismatched in any regard, the test system responded with a "603+" response structured as described in the relevant standard [7] and with varying "Reason" header value lengths. While the FCC has mandated future adoption and support of the "603+" response mechanism [13], our test results demonstrate that industry support is presently very poor. Across numerous test calls and an indeterminate number of call paths (more than 6 confirmed), only the 603 header itself was returned to the call origin. In a majority of cases, the special "603 Network Blocked" syntax was altered to the standard "603 Decline" by intermediate carriers. In no case was the Reason header described in the standard [7] returned to the origin telecom. The expected behavior, in light of the FCC's requirements, is that the SIP 603 response's Reason header should be transmitted in full and unedited back to the call origin.

Privacy requests & STIR/SHAKEN. The FCC mandates that providers do not pass information about a caller to the recipient when the caller requests privacy [35]. We wanted to test whether integrations or implementations of STIR/SHAKEN at telecoms affected compliance with this privacy requirement.

We requested privacy on phone calls to six retail service providers. Despite these privacy requests, we identified a provider in which calling party number information was exposed to the called party, directly attributable to this provider's STIR/SHAKEN implementation. Vitelity (a subunit of Sinch, a multi-billion dollar service provider), has a feature which allows a subscriber to choose whether to receive "verstat" data — a summary assessment of whether the call was successfully validated with STIR/SHAKEN. In calls to a test phone number at Vitelity, while the verstat feature was enabled, the P-Asserted-Identity header, populated with the caller's number, was transmitted to the end user, even when calling party privacy was requested. The relevant STIR/SHAKEN standard makes clear that it is possible to deliver verstat information to a user while respecting call privacy requests and describes the proper mechanism to achieve this [40]. At Vitelity, we observed that if the verstat feature was disabled, Vitelity correctly honored the privacy request and prevented the end user from receiving the calling party number. As verstat is inherently a STIR/SHAKEN related feature, we regard that Vitelity's STIR/SHAKEN implementation resulted in Vitelity likely failing to comply with its obligations under federal regulation [35]. In accordance with responsible disclosure practices, we reported the observed behavior and corresponding privacy concerns to Vitelity via email in August 2025.

8 Solutions & Recommendations

Certificate transparency. A number of the issues we've identified with the STIR/SHAKEN PKI can be addressed by introducing certificate transparency. Certificate transparency is a system for monitoring and auditing the issuance of digital certificates through a public append-only log. It has already been successfully deployed in other PKIs such as the web. Once implemented, references to certificates (such as the URL to `x5u` field) should be replaced with references to indexes in the certificate transparency log. Today, `x5u` is a URL chosen entirely by the signing provider and dereferenced by the VS, presenting an opportunity for server-side request forgery (SSRF), in which an attacker-controlled URL causes a server to issue malicious requests to internal-only services. SSRF is a widespread category of attacks considered one of the top web application security risks [87, 88]. Replacing `x5u` with an index into a certificate transparency log removes this vector entirely: the VS would only ever query a small, fixed set of vetted log servers, never a destination chosen by the signer. Certificate transparency would allow for the detection of misissued and malformed certificates, a widespread problem in the STIR/SHAKEN PKI. It would also allow the PA to rapidly determine which certificates need to be revoked when a provider is expelled from the STIR/SHAKEN ecosystem. Additionally, since certificate transparency logs are public and append only, origin providers would no longer be able to use ephemeral URLs to prevent post-call investigation. Logs would also make certificate retrieval far more

robust, because the STIR/SHAKEN certificate corpus is small enough to prefetch in its entirety. Across a year of observed traffic we saw only 8457 unique telecom certificates and 47 CA certificates (§7), a population orders of magnitude smaller than the web PKI's. Rather than dereferencing a signer-supplied URL on the critical path of every call, a VS could periodically pull the full set from the logs and verify against a local cache, refreshing often enough to keep pace with the 64-day average validity period we observed. Verification would then never fail because a certificate host was slow, unreachable, or had removed the certificate—a real failure mode today, as 11.42% of the `x5u` URLs we tested returned no certificate at all. Finally, certificate transparency would improve the overall robustness of the STIR/SHAKEN PKI.

Encrypting PASSporTs. The STIR/SHAKEN PASSporT includes highly sensitive metadata about the call and is accessible to a number of parties for a particular call. We propose that the origin provider encrypts the PASSporT to the public key of the destination provider providing end-to-end confidentiality. This prevents both middle providers and CPSs from learning the metadata in the PASSporT. The key difficulty in doing this is that there is currently no mechanism for the origin provider to reliably discover the key of the destination provider

8.1 Key Discovery

Two of the solutions we propose—encrypting passports in transit and using asymmetric message franking (§8.2)—depend on the ability of the origin provider to discover the public key of the destination provider. The STIR/SHAKEN ecosystem does not currently provide a mechanism for such key discovery. This difficulty has several structural sources. First, in the United States, telephone numbers are portable: the carrier that originally received a number block may no longer serve that number. Routing therefore depends on a per-call dip against the Local Routing Number (LRN) database [50, 37], which returns a switch identifier—not a STIR/SHAKEN signing identity. Moreover, in many call paths the LRN dip is performed by an intermediate transit carrier rather than the origin, so even the carrier-level identity of the terminating provider may not be known to the origin at the point of signing. Second, no voice equivalent exists of the NetNumber-administered number-to-provider registry used for SMS [60]; the STIR/SHAKEN ecosystem provides no comparable directory. Third, certificate transparency alone does not close this gap: even given the terminating carrier's identity, large carriers operate multiple STIR/SHAKEN certificates for different infrastructural components (see §7), so a CT log lets one enumerate a carrier's certificates but does not identify which one signs (and verifies) calls for any particular destination number.

Because of this, we need to design a backwards compatible per-call key discovery mechanism. We propose a flexible key discovery mechanism, transparent to middle providers, and which discourages MITM attacks while retaining compatibility with existing STIR/SHAKEN specifications. We provide a brief overview of this scheme and leave the details to the relevant standardization bodies. We also note that this scheme can be replaced if the STIR/SHAKEN ecosystem adopts alternative key discovery mechanisms that allow origin providers to non-interactively determine the destination provider's public key, such as with a realtime destination phone number database similar to that used for SMS routing. [60]

SIP 603 errors. The key discovery mechanism relies on the SIP 603 error response. When a call is blocked, providers can optionally return a SIP 603 error response to the originator of the call which includes a text field of unbounded size for specifying the reason for the block. Intermediary providers are required to pass this error response back to the originator verbatim.

Probing request. The key discovery mechanism begins with a probing request to determine support for the new scheme and to request the destination provider's public key. To craft a probing request, the origin provider constructs a PASSporT with fixed values for the origin and destination phone number and any relevant key discovery information for the origin provider's public key. The discovery information can include the certificate itself, a URL pointing to the certificate, an index in a certificate transparency log, or some other information to conduct key discovery out of band. Finally, the origin provider adds a "privacy-requested" field to the PASSporT. The PASSporT can either be signed with an ephemeral (self-signed) key or the signature can be replaced with bogus data, thereby invalidating the PASSporT.

Probing response. Once a supporting destination provider receives the probing request, it checks for the "privacy-requested" field. If the field is present, the destination provider constructs a probing response with

an AMF-enabled PASSporT containing a "privacy-response" header field, the destination phone number, and any relevant key discovery information. The probing response PASSporT is signed by the destination provider, designated to the public key specified in the probing request, and sent back to the origin provider in the SIP 603 error response.

Backwards compatibility. Destination providers that do not support the probing request, due to the invalid PASSporT, will either return a SIP 603 error response without a "privacy-response" field or allow the call to proceed as normal (see §7). Since both the probing request is sent as a PASSporT and the probing response is sent as a SIP 603 error response, the key discovery mechanism is completely transparent to middle providers and compatible with existing specifications (see §7), [40], [7].

MITM mitigation and deniability. When sending the probing response, the destination provider is effectively making a claim of ownership over the destination phone number. Should a malicious middle provider attempt to intercept the probing response and send their own keys instead of the destination provider's, the origin provider will have reportable evidence of the attack. In many instances, the origin provider can know to a great degree of confidence the set of legitimate service provider codes which may properly assert responsibility for the given destination telephone number. Automated or partially automated testing could retrospectively identify attestations by improper third parties, and severe consequences can be visited upon those third parties by the regulatory regime. There is therefore a strong disincentive, virtually identical to the existing STIR/SHAKEN scheme's incentives against improper attestation, for performing this type of attack.

We claim that this key discovery mechanism need not alter the deniability properties of STIR/SHAKEN. The probing request is an invalid PASSporT and provides no cryptographic evidence; though the probing response is signed this signature could be done using asymmetric message franking (§8.2) to make it deniable.

8.2 Near-term cryptographic improvements

Above, we highlight two weaknesses that stem from the way STIR/SHAKEN uses signatures: they make traffic non-repudiable (§4), and if providers delegate signing and verification to third party AS/VS, they must send the message to be signed—containing sensitive call metadata—to the third parties (§4).

We first synthesize the security and privacy properties STIR/SHAKEN ought to provide. These are largely goals for the protocol as a whole rather than for any particular signature scheme, and stating them up front lets us be frank about how far the most easily deployable improvements fall short. We then survey the design space of changes against these goals and single out those that offer the most privacy for the least deployment effort. The same goals frame the first-principles redesign we begin in §8.3. This discussion is independent of whether PASSporT encryption (§8) is used; the two are orthogonal.

Design goals for STIR/SHAKEN cryptography. STIR/SHAKEN's security and privacy depend as much on how its cryptography is deployed—who signs, who verifies, and which of those roles are outsourced to third parties—as on the primitive itself. Our goals are best stated against this whole picture: some are demands on the signature primitive, others on how it is deployed.

First, the signature scheme used in STIR/SHAKEN must provide **authenticity**: the destination provider and the industry traceback group (ITG) must be able to verify that a given origin provider signed a message; equivalently, the signature must be unforgeable. STIR/SHAKEN must also provide **deniability** (§5): because a signed PASSporT crosses trust boundaries and may be retained or leaked, no party other than the destination provider and the ITG should be able to use it to convince anyone that a call was placed.

The third goal is a demand on deployment. It should be possible for providers to have third parties manage their STIR/SHAKEN secret keys without revealing call metadata to those third parties; we call this **third-party blindness**. Intuitively, third-party blindness is achieved if managing a provider's keys only lets a third party know the timing of STIR/SHAKEN events—e.g., signing or verifying a PASSporT—and the STIR/SHAKEN provider identity of their customer that initiated it.

To satisfy third-party blindness, the PASSporT metadata—above all the pair of origin and destination telephone numbers [59]—clearly must be hidden from the third party. More subtly, we argue that the third party should also not learn the pair of origin and destination *provider identities* for a call. Some leakage here is unavoidable: through its participation the third party necessarily learns *one* endpoint's identity—that of its own customer—and preventing it from learning that too would require prohibitively expensive machinery

	D1	D2	D3	D4	D5
S1 Status quo	DEN	DEN AS-B	DEN VS-B	DEN AS-B VS-B	DEN AS-B VS-B UNL
S2 Commit + sig	DEN	DEN	DEN VS-B	DEN VS-B	DEN VS-B UNL
S3 Blind sig	DEN	DEN	DEN VS-B	DEN VS-B	DEN VS-B
S4 Commit + AMF	✓	AS-B	VS-B	AS-B VS-B	AS-B VS-B UNL
S5 BAMF	✓	✓	VS-B	VS-B	VS-B

Table 1: Design goals achieved by candidate schemes (rows) across AS/VS deployment models (columns). Deployments: D1 both on-prem; D2 third-party AS, on-prem VS; D3 on-prem AS, third-party VS; D4 third-party AS and VS, distinct parties; D5 a single third party as both AS and VS. Goals: DEN deniability; AS-B AS-blindness; VS-B VS-blindness; UNL unlinkability (applicable only to D5). Each cell lists the applicable goals the scheme *fails*; ✓denotes that all applicable goals are met. BAMF (S5) is the only scheme that meets every applicable goal when signing is outsourced but verification is kept local (D2).

such as general-purpose MPC. What third-party blindness demands is the achievable weaker guarantee: that a single third party never learns *both* identities for the same call.

The reason to hide the provider-identity pair is not that any single pair is strongly identifying on its own—a third party that observes one signing request in isolation learns little. The concern is *aggregation*. An outsourced AS sits between many origin providers and, over the lifetime of each relationship, records for every one of them the destination provider of every call it signs. The result is a longitudinal, provider-level interaction graph—who calls whom, when, how often, and at what volume—that exposes inter-provider business relationships, calling-campaign structure, and traffic concentration, none of which depend on recovering the underlying telephone numbers. Our survey (§6) shows that a small number of vendors provide outsourced STIR/SHAKEN for a large fraction of providers, so in practice a single such graph would span much of the network.

We note a subtle asymmetry relevant when we wish to hide provider identities: for a standard signature, the signer need not learn the intended recipient of the message it signs, whereas a verifier must know the sender's public key. (For designated-verifier signatures, discussed below, even the signer must learn the recipient.) This asymmetry comes into play in our comparison of solutions below.

The design space. Which of these goals a change actually achieves depends on how a provider's authentication and verification services (AS and VS) are deployed—each may be run in-house or handed to a third party—so we lay schemes and deployments out jointly in Table 1. We refine third-party blindness into three sub-goals: *AS-blindness* (AS-B)—a third-party AS learns neither PASSporT contents nor the destination provider identity; *VS-blindness* (VS-B)—a third-party VS learns neither PASSporT contents nor the origin provider identity; and *unlinkability* (UNL)—when a single third party serves as both AS and VS, it cannot link a call's signing and verification interactions. Deniability (DEN) does not depend on AS/VS outsourcing. The candidate schemes run from today's status quo (S1, an ordinary signature on the cleartext PASSporT), through commit-then-sign/prehashing with an ordinary signature (S2) and a plain blind signature (S3), to two schemes built on *asymmetric message franking* (AMF)—a primitive we introduce below—namely commit-then-AMF-sign (S4) and the blind-AMF scheme BAMF (S5), which we develop beginning in §8.3.

The best near-term improvement: prehashing. The change with the best ratio of privacy gained to deployment effort is also the simplest. Rather than sign the PASSporT in the clear, the origin provider commits to it—for instance by signing a salted hash—and sends the opening to the destination provider alongside the signed commitment (scheme S2). This needs no new cryptographic primitive and no ecosystem-wide coordination, yet it removes the PASSporT contents from a third-party AS or VS's view entirely; and

because the AS now signs an opaque commitment, it hides the destination provider's identity from the AS as well. An additional benefit is that, because it is signature-scheme-agnostic, it is easy to deploy in parallel with a migration to post-quantum signatures.

We regard this as the improvement most worth pursuing in the near term, but it has serious limitations: first, it is not deniable—the ordinary signature on the commitment, together with the opening, is publicly verifiable evidence that the call took place, so S2 fails DEN in Table 1.

Second, prehashing fails our provider-identity-hiding privacy goal if the VS is outsourced: the receiver must tell the outsourced VS the sender identity so that it can use the right key to verify the signature. Thus, prehashing fails VS-B in deployments D3, D4, and D5. Relatedly, since the hash itself does not change between signing and verification, if both AS and VS are outsourced to the same party (deployment D5), prehashing fails to achieve unlinkability there.

The reader may be wondering why we recommend prehashing vs. other possibilities. One alternative is using a blind signature scheme that actually provides cryptographic unlinkability. This would be better than prehashing for privacy; in particular it would prevent linking signing and verification interactions in scenario D5. However, it would not resolve the provider-identity-hiding issue with outsourced verification, since the receiver must still tell the VS the sender identity. Ultimately, though, practical considerations prevent us from recommending blind signatures as a near-term solution: since STIR/SHAKEN only allows ECDSA signatures, a blind signing protocol for ECDSA would be necessary. ECDSA is notoriously difficult to adapt to the blind signing setting; state-of-the-art blind ECDSA signing protocols require heavy machinery like additively homomorphic encryption [67]. (That said, since STIR/SHAKEN will need to transition to a post-quantum signature soon anyway, it would be good for privacy if they included efficient blind signing as a selection criteria in this process.)

Another alternative recommendation would be integrating some deniability protection into STIR/SHAKEN in the near term. We do believe that deniability is a crucial privacy goal for STIR/SHAKEN, but because of a lack of standardized, deployed mechanisms for achieving it, deniability seems infeasible to add to STIR/SHAKEN in the near term. The main option for retrofitting deniability into STIR/SHAKEN today is to release expired signing keys, as has been suggested for DKIM [46]. This would add substantial complexity on top of the ongoing STIR/SHAKEN rollout, which is itself proceeding imperfectly; further, absent carefully immunizing the ITG reporting pathway against forged call logs, releasing old signing keys could undermine the main security goal of STIR/SHAKEN. Alternatively, deniability could be achieved via a new authentication primitive with the right deniability properties. There is one interesting candidate from prior work; we turn to it next.

Asymmetric message franking. Introduced by [85], AMFs are a type of designated-verifier signature that has two separate designated verifiers with different forgery abilities. One designated verifier is the *receiver*—the intended verifier of the AMF signature in the common case—and the other designated verifier is the *judge*, to whom AMF signatures can be reported if the associated message is deemed abusive.

In STIR/SHAKEN, the destination provider can play the role of the AMF receiver, and the ITG can play the role of the judge. (This would necessitate the ITG having a STIR/SHAKEN identity, which it does not currently, but this would not be difficult to do with existing STIR/SHAKEN PKI.) The original AMF construction, depicted in Figure 4, is built as a boolean composition of simple discrete-log proofs and has several properties that make it easy to deploy for STIR/SHAKEN. It is fast and easy to implement from building blocks (hashing and elliptic curve groups) present in mature cryptography libraries. Since it uses Schnorr-style proofs, it can even be implemented based on implementations of standardized Schnorr-like signatures (e.g., EdDSA [53]). Relatedly, it can use standardized elliptic-curve groups, such as P-256.

AMFs would allow adding deniability to STIR/SHAKEN. As shown by [85], AMFs are unforgeable—meeting our authenticity goal—yet strongly deniable: given only the public keys, anyone can forge a signature that looks convincing to a party without the secret keys, so a signature is not transferable evidence, while a holder of the secret keys can still tell genuine signatures from forgeries. When signing is outsourced to a third-party AS (deployment D2), commit-then-AMF-sign (S4) therefore improves on S2 by adding deniability. It still falls short of third-party blindness: AMF signing takes the receiver's public key as an explicit input (Figure 4), so a third-party AS learns the destination provider on every request. Closing this last leak requires signing an AMF *without* the signer ever seeing the receiver key—blind signing for AMFs—for which no protocol is known. Two blind designated-verifier signature constructions do exist [24, 70], but

their guarantees are unclear, and—more basically—the definitional question of *what* security blind signing for a designated-verifier signature should provide has, to our knowledge, never been asked. This is the problem we take up next. (Interestingly, since verifying an AMF requires the verifier to have their secret key, outsourced verification needs some kind of blinding as well.) In contrast to the discussion above, our study of blind signing for AMFs is a more first-principles study of what new cryptography *could* provide. Near-term deployment of our new protocol below would be difficult for a number of reasons, but as a feasibility result we believe it is important for expanding the horizon of what is seen as possible in settings like STIR/SHAKEN.

8.3 Blind signing for AMFs

Our next major contribution, developed in §9, is BAMF: a blind signing protocol for AMFs in which the AS computes an AMF signature under the origin provider's signing key while learning neither the PASSporT nor the receiver public key. This is scheme S5 in Table 1, the unique candidate meeting every applicable goal when signing is outsourced (D2). That the signer cannot tell which receiver it is signing for is a new property, which we formalize as *verifier-hiding blindness* below. Note that our definition will require hiding both the receiver and judge (in AMF terminology) but in STIR/SHAKEN we imagine there will be a single judge identity used for all AMF signatures.

The remaining deployments outsource verification as well, and there VS-B fails for every scheme, BAMF included, because whoever verifies must operate on the sender and judge public keys and thereby learns the origin identity; we do not attempt to protect this case in general. We observe only that verification's dependence on the receiver secret is confined to the single check $R = E_R^{\mathsf{sk}_R}$ (Figure 4): masking R and E_R by a common random exponent lets a provider delegate even that step to the party managing its receiver key without revealing them, so a provider outsourcing only its secrets—not its public-key operations—can still keep the origin identity hidden. We sketch this *blind-verification* step in §9 and leave a formal treatment to future work.

We first present the two security definitions we need to analyze BAMF, one for unforgeability and one for blindness. Both are new and specific to the designated-verifier setting, and they form an auxiliary contribution of independent interest: to our knowledge, ours is the first work to identify the need for *verifier-hiding* blindness—blindness that hides the identity of the designated verifier from the signer. We call the two notions *designated-verifier one-more unforgeability* (dvOMUNF) and *verifier-hiding blindness* (vhBLIND), and use these names below.

Designated-verifier one-more unforgeability. In Figure 2 we give pseudocode for the dvOMUNF game. This game is an extension of the standard (sequential) one-more unforgeability definition for blind signatures. The extension is necessary because for AMFs and other kinds of designated-verifier signatures, verification is not public as in a normal signature—it uses the designated verifier's secret.

This fact necessitates making a few design choices in the definition: first, since AMFs have two designated verifiers, which of the two verification procedures should the definition use? Second, since the adversary can no longer run verification itself, should the adversary get a verification oracle? Finally, as described above, we also want to enable blind verification—should the adversary get to *blindly* sign and verify at the same time? Ultimately, our definition errs on the side of simplicity: it does not give a verification oracle (blind or otherwise) to the adversary; further, it uses the receiver's secret to verify, as this is the most common case in practice.

A scheme meeting dvOMUNF satisfies our authenticity goal: a valid signature cannot be produced except through a signing session with the party holding the key. We ask for the *one-more* form of this guarantee, rather than ordinary unforgeability, for a specific reason. Because the signer never sees the message, forgery cannot be defined in terms of a fresh message; dvOMUNF instead bounds the number of valid signatures an adversary can produce by the number of signing sessions it completes. This is exactly the guarantee that matters under the partial compromise we consider most realistic for STIR/SHAKEN: an attacker who breaches a provider's signing service at low privilege can drive it—issuing $\mathtt{S}_1$ and $\mathtt{S}_2$ queries—but cannot exfiltrate the signing key, which sits behind higher-privilege controls or in an HSM. Against such an attacker, dvOMUNF guarantees both that the key cannot be extracted and that no more valid signatures can be minted than sessions were actually run, so a breach of the signing service cannot escalate into an unbounded signing capability.

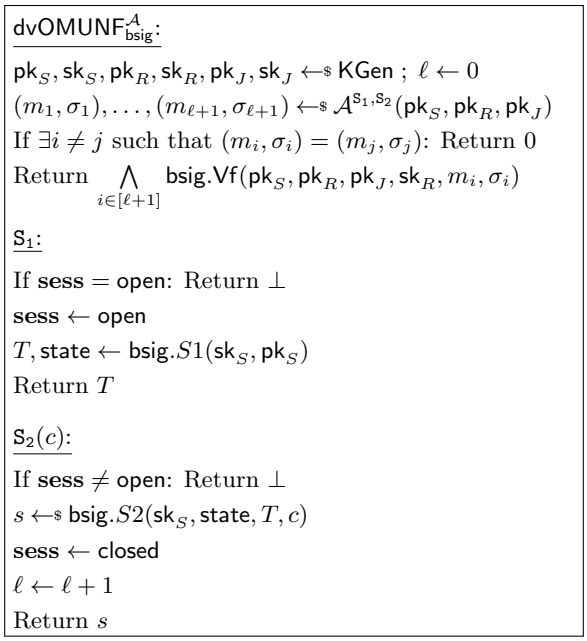

Figure 2: Sequential one-more unforgeability security game for a blind signature bsig with secret sk_R input to Vf.

Verifier-hiding blindness. Our novel verifier-hiding blindness notion for blind signatures is described in Figure 3. The standard blindness game does not guarantee that the identity of the designated verifier is hidden from the signer. As we described above, this is needed to meet our third-party blindness design goal. We include this requirement in vhBLIND by allowing $\mathcal{A}$ to specify two pairs of identity keys along with the two messages in the init oracle. Based on the game bit b it passes the right pair of identity keys to U_1 and U_2. The remainder of the game is unchanged from the standard blindness notion. Note that we do not include in vhBLIND any checks about the well-formedness of the identity keys. This means that $\mathcal{A}$ could output malformed keys to try to break blindness [58]. We argue this attack is largely prevented in our application—STIR/SHAKEN CAs can simply refuse to issue signatures for malformed public keys—and does not need to be made explicit in our definition. We leave a separate definition for blindness in blind verification to future work.

9 Blind AMFs

We begin by recalling the AMF construction (see Figure 4 for Sign and Vf pseudocode). Let $\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J$ be the sender, receiver, and judge public keys respectively. AMFs are signatures of knowledge for the relation $\mathcal{R}_{AMF}$:

$$\mathcal{R}_{AMF} = \{((G, \mathsf{pk}_S, \mathsf{pk}_J, J, R, E_J); (t, u, v, w)) : (\mathsf{pk}_S = G^t \vee J = G^u) \wedge ((J = \mathsf{pk}_J^v \wedge E_J = G^v) \vee R = G^w)\} .$$

We use a modified version of Camenisch-Stadler notation for relations—the relation $\mathcal{R}$ is

$$\{(\text{public inputs}; \text{private witnesses}) : \text{constraints between inputs and witnesses}\}$$

Intuitively, the left side of conjunction $(\mathsf{pk}_S = G^t \vee J = G^u)$ is a designated-verifier signature to the judge, which allows the judge that holds sk_J to verify that the message was signed by holder of sk_S. The right-hand side of the conjunction lets the receiver verify that the judge will be able to verify the signature as coming from the sender. This guarantee is the *sender-binding* property, and it in fact requires a check that the original construction of [85] omits: that $E_R \neq 1$. Without it, a malicious sender can set $E_R = R = 1$, so that the receiver's designated-verifier check $R = E_R^{\mathsf{sk}_R}$ holds for *every* sk_R—since $1 = 1^{\mathsf{sk}_R}$—and so passes without

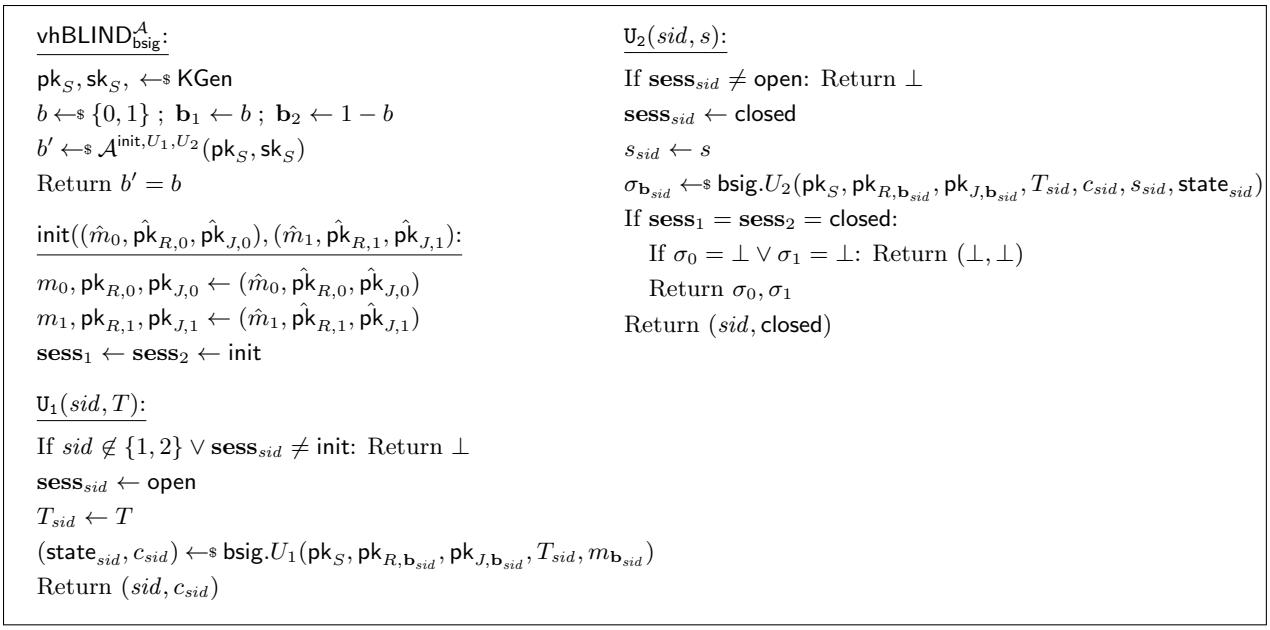

Figure 3: Verifier-hiding blindness game for a blind signing protocol bsig for a DV signature.

the sender knowing sk_R. The sender can then satisfy the right-hand clause of $\mathcal{R}_{AMF}$ through its trivial disjunct $R = G^0$ while simulating the judge disjunct ($J = \mathsf{pk}_J^v \wedge E_J = G^v$) altogether, producing a signature the receiver accepts but the judge cannot verify—a sender-binding violation. Requiring $E_R \neq 1$ rejects this degenerate witness and forces the sender to prove the judge disjunct honestly. The gap surfaces in their own sender-binding proof, whose reduction recovers the receiver secret as $\mathsf{sk}_R = w/\beta$ with $\beta = \log_G E_R$—undefined exactly when $E_R = 1$. We additionally reject the symmetric degenerate case $E_J = 1$, a check likewise absent from [85]; it is not required for sender-binding, but we retain it in verification as a well-formedness safeguard.

Blind verification. We include only a brief and informal discussion of blind verification for AMFs, and leave most of the details for future work. Looking at the AMF verification procedure in Figure 4, we can see that the receiver secret key sk_R is only used to check that the equation $R = E_R^{\mathsf{sk}_R}$ holds for the group elements R, E_R in the AMF signature. The destination provider can have their third party check this relation without revealing R, E_R by blinding them both with a common random value $a \leftarrow\!\!\$\ \mathbb{Z}_p^*$ and then letting the third party perform the check with sk_R as normal. If the equation holds, it will also hold on the blinded group elements; if it does not hold, no value of a will cause the third party to output true. It is not difficult to see this satisfies perfect blindness—as long as the signature verifies the VS's view is independent of the elements R, E_R. Informally, it also seems like this does not give any advantage in forging signatures for the destination provider's key pk_R, by reduction to the gap discrete-logarithm assumption defined in [85], but we leave a formal answer to this question to future work. The rest of the signature verification can be done with just the public key pk_S of the origin provider—obtainable from the `x5u` value in the PASSporT—and the ITG public key pk_J. As discussed above, if the public parts of verification are outsourced, this breaks provider-identity-hiding. Fixing this seems to require heavy techniques like MPC; we leave this as a question for future work.

Blind signing for AMFs. We describe our protocol BAMF for blind AMF signing in Figure 5. The basic underlying observation is that the sender secret key sk_S is only needed in one of the four clauses of the AMF relation; the other three can be proven using either the actual witnesses or the simulated transcript of the OR-proof construction. Since the clause involving the sender secret is just a proof of knowledge of sk_S, we use the standard blind Schnorr signing protocol to produce the proof for that clause, and generate the other three as normal. We will show that this protocol satisfies a natural lifting of the standard sequential one-more unforgeability notion for blind signatures; we will also show it satisfies a new, stronger form of

$\underline{\mathsf{Sign}(\mathsf{sk}_S, \mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, m)}$:

$(\alpha, \beta) \leftarrow\!\!\$\ \mathbb{Z}_p$
$J \leftarrow \mathsf{pk}_J^{\alpha}\ ;\ E_J \leftarrow G^{\alpha}$
$R \leftarrow \mathsf{pk}_R^{\beta}\ ;\ E_R \leftarrow G^{\beta}$
$\widetilde{A}_u, \tilde{c}_u, \tilde{z}_u \leftarrow\!\!\$\ \mathsf{Sim}(J)$
$a_t \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ A_t \leftarrow G^{a_t}$
$\widetilde{A}_w, \tilde{c}_w, \tilde{z}_w \leftarrow\!\!\$\ \mathsf{Sim}(R)$
$a_v \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ A_{v,1} \leftarrow G^{a_v}\ ;\ A_{v,2} \leftarrow \mathsf{pk}_J^{a_v}$
$L \leftarrow (J, R, E_J, E_R, A_t, \widetilde{A}_u, A_{v,1}, A_{v,2}, \widetilde{A}_w)$
$c \leftarrow H(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, L, m)$
$c_t \leftarrow c - \tilde{c}_u$
$z_t \leftarrow a_t + c_t \mathsf{sk}_S$
$c_v \leftarrow c - \tilde{c}_w$
$z_v \leftarrow a_v + c_v \alpha$
Return $L, (c_t, z_t, \tilde{z}_u), (c_v, z_v, \tilde{z}_w)$

$\underline{\mathsf{Vf}(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, \mathsf{sk}_R, m, \sigma)}$:

$L, (c_t, z_t, z_u), (c_v, z_v, z_w) \leftarrow \sigma$
$(J, R, E_J, E_R, A_t, A_u, A_{v,1}, A_{v,2}, A_w) \leftarrow L$
$b_0 \leftarrow R = E_R^{\mathsf{sk}_R} \wedge E_R \neq 1 \wedge E_J \neq 1$
$c \leftarrow H(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, L, m)$
$c_u \leftarrow c - c_t$
$c_w \leftarrow c - c_v$
$b_1 \leftarrow G^{z_t} = A_t \cdot \mathsf{pk}_S^{c_t}$
$b_2 \leftarrow G^{z_u} = A_u \cdot J^{c_u}$
$b_3 \leftarrow G^{z_v} = A_{v,1} \cdot E_J^{c_v} \wedge \mathsf{pk}_J^{z_v} = A_{v,2} \cdot J^{c_v}$
$b_4 \leftarrow G^{z_w} = A_w \cdot R^{c_w}$
Return $\bigwedge_{i=0}^{4} b_i$

Figure 4: AMF signing and receiver verification algorithms. Note that [85] did not include the checks $E_R \neq 1$ and $E_J \neq 1$. The procedure Sim is the standard Schnorr transcript simulator—concretely, on input a group element X, $\mathsf{Sim}(X)$ generates $(c, z) \leftarrow\!\!\$\ \mathbb{Z}_p$ then outputs the triple $(G^z X^{-c}, c, z)$.

blindness for blind signing that also guarantees "verifier hiding", as needed for STIR/SHAKEN.

9.1 Security Analysis of BAMF

We prove that BAMF satisfies our two new security definitions. We begin with verifier-hiding blindness.

Theorem 1. *The* BAMF *blind signing protocol in Figure 5 satisfies perfect VH-blindness as defined in Figure 3—that is, for all adversaries* $\mathcal{A}$, $\Pr[\mathsf{vhBLIND}^{\mathcal{A}}_{\mathsf{BAMF}} = 1] = \frac{1}{2}$.

Proof. First we show that $\Pr[\mathsf{vhBLIND}^{\mathcal{A}}_{\mathsf{BAMF}} = 1] = \frac{1}{2}$ if for the bit b' output by $\mathcal{A}$ in vhBLIND, $\Pr[b' = 1 | b = 1] = \Pr[b' = 1 | b = 0]$, and likewise for the case $b' = 0$.

Let $E_{b'}$ for a fixed bit b' be the event that $\mathcal{A}$ outputs bit b' in game vhBLIND. Let E_{win} be the event that $\mathcal{A}$ wins vhBLIND, i.e., that $\mathsf{vhBLIND}^{\mathcal{A}}_{\mathsf{BAMF}} = 1$. We assume without loss of generality that $\mathcal{A}$'s output b' is always a bit, so that conditioned on either value of b the events E_0 and E_1 partition the probability space. (Any output outside $\{0, 1\}$ loses the game and can be replaced by a fixed bit, say 0.)

We know that $1 = \Pr[E_{win}] + \Pr[\neg E_{win}]$. We use the law of total probability on the bit b sampled in vhBLIND to write

$$\Pr[\neg E_{win}] = \Pr[E_1 | b = 0] \Pr[b = 0] + \Pr[E_0 | b = 1] \Pr[b = 1] \ .$$

Since $\Pr[b = 0] = \Pr[b = 1] = \frac{1}{2}$, the LHS just equals

$$\frac{1}{2}(\Pr[E_1 | b = 0] + \Pr[E_0 | b = 1]) \ .$$

But by the same argument applied to $\Pr[E_{win}]$, we can rewrite everything as

$$1 = \Pr[E_{win}] + \Pr[\neg E_{win}] = \frac{1}{2}(\Pr[E_1 | b = 1] + \Pr[E_0 | b = 0] + \Pr[E_1 | b = 0] + \Pr[E_0 | b = 1])$$

Now, as long as $\Pr[E_1 | b = 1] = \Pr[E_1 | b = 0]$ and likewise for E_0, we can replace each "lose" probability (the two terms coming from $\Pr[\neg E_{win}]$, with mismatched bits) by its "win" counterpart (the corresponding term from $\Pr[E_{win}]$, with matching bits) in the equation above, to get

$$1 = \frac{1}{2}(2 \Pr[E_1 | b = 1] + 2 \Pr[E_0 | b = 0]) \ .$$

$\underline{\mathsf{BAMF}.S1(\mathsf{sk}_S, \mathsf{pk}_S)}$:
$t \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ T \leftarrow G^t$
Return (t, T)

$\underline{\mathsf{BAMF}.S2(\mathsf{sk}_S, \mathsf{pk}_S, \mathsf{state}, T, c)}$:
$t \leftarrow \mathsf{state}$
$s \leftarrow t + c\mathsf{sk}_S$
Return s

$\underline{\mathsf{BAMF}.U2(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, T', m, \mathsf{state}, c, s)}$:
$(\gamma, \delta, c_t, c_\perp, K, M, N, L, T) \leftarrow \mathsf{state}$
If $G^s \neq T \cdot \mathsf{pk}_S^c$: Return $\perp$
$c_v \leftarrow c_\perp - \tilde{c}_w$
$z_v \leftarrow a_v + c_v\alpha$
Return $L, (c_t, s + \gamma, \tilde{z}_u), (c_v, z_v, \tilde{z}_w)$

$\underline{\mathsf{BAMF}.U1(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, T, m)}$:
$\gamma, \delta \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ T' \leftarrow T \cdot G^\gamma \cdot \mathsf{pk}_S^\delta$
$(\alpha, \beta) \leftarrow\!\!\$\ \mathbb{Z}_p$
$J \leftarrow \mathsf{pk}_J^\alpha\ ;\ E_J \leftarrow G^\alpha\ ;\ R \leftarrow \mathsf{pk}_R^\beta\ ;\ E_R \leftarrow G^\beta$
$\widetilde{A}_u, \tilde{c}_u, \tilde{z}_u \leftarrow\!\!\$\ \mathsf{Sim}(J)$
$\widetilde{A}_w, \tilde{c}_w, \tilde{z}_w \leftarrow\!\!\$\ \mathsf{Sim}(R)$
$a_v \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ A_{v,1} \leftarrow G^{a_v}\ ;\ A_{v,2} \leftarrow \mathsf{pk}_J^{a_v}$
$L \leftarrow (J, R, E_J, E_R, T', \widetilde{A}_u, A_{v,1}, A_{v,2}, \widetilde{A}_w)$
$K \leftarrow (\alpha, \beta, a_v)$
$M \leftarrow (\widetilde{A}_u, \tilde{c}_u, \tilde{z}_u)$
$N \leftarrow (\widetilde{A}_w, \tilde{c}_w, \tilde{z}_w)$
$c_\perp \leftarrow H(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, L, m)$
$c_t \leftarrow c_\perp - \tilde{c}_u$
$c \leftarrow c_t + \delta$
Return $((\gamma, \delta, c_t, c_\perp, K, M, N, L, T), c)$

Figure 5: Our AMF blind signing protocol. The procedure Sim is as defined in Figure 4.

So we get that $1 = \Pr[E_1|b = 1] + \Pr[E_0|b = 0]$; however the RHS here is just $2\Pr[E_{win}]$, so rewriting we can get

$$\frac{1}{2} = \Pr[E_{win}] = \Pr[\mathsf{vhBLIND}^{\mathcal{A}}_{\mathsf{BAMF}} = 1]$$

as desired. To summarize and rephrase, we've shown that as long as the probability $\mathcal{A}$ outputs 1 in game $\mathsf{vhBLIND}$ is the same for the case $b = 0$ and $b = 1$, and likewise for outputting 0, $\mathcal{A}$ has exactly a $1/2$ probability of winning the verifier-hiding blindness game. Because the 0-case is symmetric, we focus on the 1-case, and restate the new goal to prove:

$$\Pr[E_1|b = 0] = \Pr[E_1|b = 1]\ . \tag{1}$$

Next we will explain why this is a useful simplification, but first we need to specify two restrictions on $\mathcal{A}$. First, we restrict $\mathcal{A}$'s randomness to be fixed; second, we restrict that $\mathcal{A}$ always receives valid signatures (not $(\perp, \perp)$) as output from its second call to $\mathtt{U}_2$.

These restrictions are standard and justified in [77], but we restate the argument here for clarity. (Experienced readers can safely skip to the next paragraph.) First, we justify fixing $\mathcal{A}$'s randomness. Observe that since $\mathcal{A}$'s randomness is independent of the challenge bit b, we can apply the law of total probability to the adversary's randomness (call it ρ) to get that $\Pr[E_1|b = 0] = \sum_\rho \Pr[\rho]\Pr[E_1|\rho, b = 0]$, and likewise for $b = 1$. Thus, if we can prove $\Pr[E_1|\rho, b = 0] = \Pr[E_1|\rho, b = 1]$ *for every* ρ—in words, this is Equation 1 conditioned on a fixed value of ρ—the equation must also hold when ρ is uniform random, by averaging. Next, we justify $\mathcal{A}$ always receiving non-$\perp$ signatures. Let V be the event that the game outputs non-$\perp$ signatures. Thus $\Pr[E_1|\rho, b = 0] = \Pr[E_1 \wedge V|\rho, b = 0] + \Pr[E_1 \wedge \neg V|\rho, b = 0]$ and likewise for $b = 1$. The right-hand term measures the probability of $\mathcal{A}$ outputting 1 when the game outputs $(\perp, \perp)$ from $\mathtt{U}_2$. If $\neg V$ happens, $\mathcal{A}$'s view depends on b only through the two challenges c_1, c_2; it does *not* see the final signatures. Both challenges are blinded by a uniformly random element of $\mathbb{Z}_p$, δ_1 and δ_2; without the final signatures these are evidently independent of everything else and so convey no information about b, thus $\Pr[E_1 \wedge \neg V|\rho, b = 0] = \Pr[E_1 \wedge \neg V|\rho, b = 1]$.

With these restrictions in place we can proceed to the main part of our argument, namely proving that $\Pr[E_1 \wedge V|\rho, b = 0] = \Pr[E_1 \wedge V|\rho, b = 1]$. For clarity, since we will be using game hops, we will slightly relabel E_1 as $E_1^{G_0}$, making explicit that this is the event $\mathcal{A}$ outputs 1 in game G_0. Our starting game G_0 is in Figure 6; it is identical to $\mathsf{vhBLIND}^{\mathcal{A}}_{\mathsf{BAMF}}$, except it inlines the BAMF procedures, samples the per-session randomness in advance, and renames the post-init session state from init to open-pending.

Now we will reframe our goal slightly: we will argue about the distribution of $\mathcal{A}$'s view in $\mathbf{G}_0$, namely the transcript of $\mathcal{A}$'s oracle inputs and outputs. This transcript includes the sender public key $\mathsf{pk}_S, \mathsf{sk}_S$, its inputs

$\underline{\mathbf{G}_0 = \mathsf{vhBLIND}^{\mathcal{A}}_{\mathsf{BAMF}}:}$

$\mathsf{sk}_S \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ \mathsf{pk}_S \leftarrow G^{\mathsf{sk}_S}$

$b \leftarrow\!\!\$\ \{0,1\}\ ;\ \mathbf{b}_1 \leftarrow b\ ;\ \mathbf{b}_2 \leftarrow 1-b$

$\mathbf{sess}_1 \leftarrow \mathbf{sess}_2 \leftarrow \mathsf{init}$

Sample fresh per-session randomness for $sid \in \{1,2\}$:

$\gamma_{sid}, \delta_{sid}, \alpha_{sid}, \beta_{sid}, a_{v,sid}, \tilde{c}_{u,sid}, \tilde{z}_{u,sid}, \tilde{c}_{w,sid}, \tilde{z}_{w,sid} \leftarrow\!\!\$\ \mathbb{Z}_p$

$b' \leftarrow\!\!\$\ \mathcal{A}^{\mathsf{init},\mathtt{U}_1,\mathtt{U}_2,H}(\mathsf{pk}_S, \mathsf{sk}_S)$

Return $b' = b$

$\underline{\mathsf{init}((\hat{m}_0, \hat{\mathsf{pk}}_{R,0}, \hat{\mathsf{pk}}_{J,0}), (\hat{m}_1, \hat{\mathsf{pk}}_{R,1}, \hat{\mathsf{pk}}_{J,1})):}$

$(m_0, \mathsf{pk}_{R,0}, \mathsf{pk}_{J,0}) \leftarrow (\hat{m}_0, \hat{\mathsf{pk}}_{R,0}, \hat{\mathsf{pk}}_{J,0})$

$(m_1, \mathsf{pk}_{R,1}, \mathsf{pk}_{J,1}) \leftarrow (\hat{m}_1, \hat{\mathsf{pk}}_{R,1}, \hat{\mathsf{pk}}_{J,1})$

$\mathbf{sess}_1 \leftarrow \mathbf{sess}_2 \leftarrow \mathsf{open}$-pending

$\underline{\mathtt{U}_1(sid, T_{sid}):}$

If $sid \notin \{1,2\} \vee \mathbf{sess}_{sid} \neq \mathsf{open}$-pending: Return $\perp$

$\mathbf{sess}_{sid} \leftarrow \mathsf{open}$

$T'_{sid} \leftarrow T_{sid} \cdot G^{\gamma_{sid}} \cdot \mathsf{pk}_S^{\delta_{sid}}$

$J_{sid} \leftarrow \mathsf{pk}_{J,\mathbf{b}_{sid}}^{\alpha_{sid}}\ ;\ E_{J,sid} \leftarrow G^{\alpha_{sid}}$

$R_{sid} \leftarrow \mathsf{pk}_{R,\mathbf{b}_{sid}}^{\beta_{sid}}\ ;\ E_{R,sid} \leftarrow G^{\beta_{sid}}$

$\widetilde{A}_{u,sid} \leftarrow G^{\tilde{z}_{u,sid}} \cdot J_{sid}^{-\tilde{c}_{u,sid}}$

$\widetilde{A}_{w,sid} \leftarrow G^{\tilde{z}_{w,sid}} \cdot R_{sid}^{-\tilde{c}_{w,sid}}$

$A_{v,1,sid} \leftarrow G^{a_{v,sid}}\ ;\ A_{v,2,sid} \leftarrow \mathsf{pk}_{J,\mathbf{b}_{sid}}^{a_{v,sid}}$

$L_{sid} \leftarrow (J_{sid}, R_{sid}, E_{J,sid}, E_{R,sid}, \widetilde{A}_{u,sid}, T'_{sid}, \widetilde{A}_{w,sid}, A_{v,1,sid}, A_{v,2,sid})$

$c_{\perp,sid} \leftarrow H(\mathsf{pk}_S, \mathsf{pk}_{R,\mathbf{b}_{sid}}, \mathsf{pk}_{J,\mathbf{b}_{sid}}, L_{sid}, m_{\mathbf{b}_{sid}})$

$c_{t,sid} \leftarrow c_{\perp,sid} - \tilde{c}_{u,sid}\ ;\ c_{v,sid} \leftarrow c_{\perp,sid} - \tilde{c}_{w,sid}$

$c_{sid} \leftarrow c_{t,sid} + \delta_{sid}$

Save $\mathsf{state}_{sid} \leftarrow (sid, \mathbf{b}_{sid}, T_{sid}, L_{sid}, c_{sid}, c_{t,sid}, c_{v,sid})$

Return (sid, c_{sid})

$\underline{\mathtt{U}_2(sid, s):}$

If $\mathbf{sess}_{sid} \neq \mathsf{open}$: Return $\perp$

$\mathbf{sess}_{sid} \leftarrow \mathsf{closed}\ ;\ s_{sid} \leftarrow s$

$(sid, \mathbf{b}_{sid}, T_{sid}, L_{sid}, c_{sid}, c_{t,sid}, c_{v,sid}) \leftarrow \mathsf{state}_{sid}$

If $G^{s_{sid}} \neq T_{sid} \cdot \mathsf{pk}_S^{c_{sid}}$: $\sigma_{\mathbf{b}_{sid}} \leftarrow \perp$

Else:

$z_{t,sid} \leftarrow s_{sid} + \gamma_{sid}$

$z_{v,sid} \leftarrow a_{v,sid} + c_{v,sid} \cdot \alpha_{sid}$

$\sigma_{\mathbf{b}_{sid}} \leftarrow (L_{sid}, (c_{t,sid}, z_{t,sid}, \tilde{z}_{u,sid}), (c_{v,sid}, z_{v,sid}, \tilde{z}_{w,sid}))$

If $\mathbf{sess}_1 = \mathbf{sess}_2 = \mathsf{closed}$:

If $\sigma_0 = \perp \vee \sigma_1 = \perp$: Return $(\perp, \perp)$

Return (σ_0, σ_1)

Return (sid, closed)

$\underline{H(X, Y, Z, \vec{L}, m):}$

If $\mathcal{T}[X, Y, Z, \vec{L}, m] = \perp$: $\mathcal{T}[X, Y, Z, \vec{L}, m] \leftarrow\!\!\$\ \mathbb{Z}_p$

Return $\mathcal{T}[X, Y, Z, \vec{L}, m]$

Figure 6: Game $\mathbf{G}_0$. The verifier-hiding blindness game with the BAMF procedures from Figure 5 inlined and per-session randomness sampled at the start instead of in $\mathtt{U}_1$.

$((\hat{m}_0, \hat{\mathsf{pk}}_{R,0}, \hat{\mathsf{pk}}_{J,0}), (\hat{m}_1, \hat{\mathsf{pk}}_{R,1}, \hat{\mathsf{pk}}_{J,1}))$ to oracle init, the per-session $\mathtt{U}_1$ input-output pairs $(T_1, c_1; T_2, c_2)$ and $\mathtt{U}_2$ inputs (s_1, s_2) and the signatures (σ_0, σ_1), and all the RO input-output pairs it queried.

Let the random variable denoting $\mathcal{A}$'s transcript in $\mathbf{G}_0$ be $\Delta_{\mathbf{G}_0}$, and let π denote an arbitrary fixed transcript. Since the transcript is the only thing the adversary sees that depends on b, if we let $\mathcal{S}$ denote the set of transcripts that causes $\mathcal{A}$ to output 1 in $\mathbf{G}_0$,

$$\Pr[E_1^{\mathbf{G}_0} \wedge V | \rho, b = 0] = \sum_{\pi \in \mathcal{S}} \Pr[\Delta_{\mathbf{G}_0} = \pi \wedge V | \rho, b = 0]$$

by an (exact) union bound and likewise for the $b = 1$ case. Thus, if we can argue that for any fixed π, $\Pr[\Delta_{\mathbf{G}_0} = \pi \wedge V | \rho, b = 0] = \Pr[\Delta_{\mathbf{G}_0} = \pi \wedge V | \rho, b = 1]$, we will be done.

We move to the game G_1, depicted in Figure 7, which is identical to G_0 except for three key changes: first, the validity check is removed from $\mathtt{U}_2$; second, T'_{sid} is computed by sampling a random element χ_{sid} of $\mathbb{Z}_p$ to be its discrete log, then setting $T'_{sid} \leftarrow G^{\chi_{sid}}$; third, the signature element $z_{t,sid}$ is computed in $\mathtt{U}_2$ as $\chi_{sid} + c_{t,sid}\mathsf{sk}_S$.

We argue that for any transcript π, $\Pr[\Delta_{\mathbf{G}_0} = \pi \wedge V | \rho, b = 0] = \Pr[\Delta_{\mathbf{G}_1} = \pi \wedge V | \rho, b = 0]$. It is intuitively clear that removing the validity check does not change the distribution—we are in the part of the probability space where the event V must happen, so the validity check never fails. That the other two changes do not affect the distribution is, a priori, much more surprising. To see why, first write $T'_{sid} \leftarrow T_{sid} G^{\gamma_{sid}} \mathsf{pk}_S^{\delta_{sid}}$; this is how T'_{sid} is computed in G_0. Also in G_0, the validity condition gives us that $G^{s_{sid}} = T_{sid}\mathsf{pk}_S^{c_{sid}}$. Taking discrete logs and rewriting, this gives us that in G_0, $\gamma_{sid} = \mathsf{dlog}_G(T'_{sid}) - \mathsf{dlog}_G(T_{sid}) - \mathsf{sk}_S\delta_{sid}$ and $s_{sid} = \mathsf{dlog}_G(T_{sid}) + c_{sid}\mathsf{sk}_S$. But then the value $z_{t,sid}$, which in $\mathbf{G}_0$ is computed as $s_{sid} + \gamma_{sid}$, could also be computed (by combining the previous two equations) as

$$\begin{aligned} z_{t,sid} &= [\mathsf{dlog}_G(T_{sid}) + c_{sid}\mathsf{sk}_S] + \mathsf{dlog}_G(T'_{sid}) - \mathsf{dlog}_G(T_{sid}) - \mathsf{sk}_S\delta_{sid} && (2) \\ &= \mathsf{dlog}_G(T'_{sid}) + \mathsf{sk}_S(c_{sid} - \delta_{sid})\ . && (3) \end{aligned}$$

But in both G_0 and G_1, $c_{sid} - \delta_{sid} = c_{t,sid}$, so we can rewrite the second line's RHS as $\mathsf{dlog}_G(T'_{sid}) + c_{t,sid}\mathsf{sk}_S$. Since $\chi_{sid} = \mathsf{dlog}_G(T'_{sid})$, the computation of $z_{t,sid}$ in G_1 is, in fact, a rewriting of the same computation in G_0. Since T'_{sid} is uniform random (conditioned on fixed values of $\mathsf{sk}_S, \delta_{sid}, T_{sid}$) in $\mathbf{G}_0$, changing how it is sampled does not change its distribution in G_1. Because of this, $\Pr[\Delta_{\mathbf{G}_0} = \pi \wedge V | \rho, b = 0] = \Pr[\Delta_{\mathbf{G}_1} = \pi \wedge V | \rho, b = 0]$, as claimed.

Here we draw the reader's attention to a fairly surprising fact: in game $\mathbf{G}_1$ the outputs of $\mathtt{U}_1$ and $\mathtt{U}_2$ do not depend at all on $\mathcal{A}$'s inputs to those oracles! This means in particular that the oracle outputs—for $\mathtt{U}_1$ the challenge c_{sid} and for $\mathtt{U}_2$ the signatures—can be computed as soon as the adversary queries the init oracle. To make this explicit, we introduce game G_2 (with pseudocode in Figure 8). In $\mathbf{G}_2$, the only occurrence of the challenge bit b is in $\mathtt{U}_1$, where it selects which of the two precomputed challenges c_0, c_1 is returned. The precomputed values themselves are exactly those that $\mathbf{G}_1$ computes: as observed above, in $\mathbf{G}_1$ the $\mathtt{U}_1$ and $\mathtt{U}_2$ outputs depend only on the per-session randomness and on the triple a session serves, never on $\mathcal{A}$'s inputs T_{sid}, s_{sid}, so they may be computed eagerly in init. Game $\mathbf{G}_2$ does exactly this, indexing the randomness and the resulting values by the triple $i \in \{0, 1\}$ that they pertain to. (Moving the game's H queries into init changes no value, as H is a consistent random oracle.) The signatures (σ_0, σ_1) are released by $\mathtt{U}_2$ in a fixed order, so each σ_i depends only on the triple $(m_i, \mathsf{pk}_{R,i}, \mathsf{pk}_{J,i})$. Thus $\mathbf{G}_1$ and $\mathbf{G}_2$ induce the same transcript distribution: for any fixed π, $\Pr[\Delta_{\mathbf{G}_1} = \pi \wedge V | \rho, b = 0] = \Pr[\Delta_{\mathbf{G}_2} = \pi \wedge V | \rho, b = 0]$, and likewise for $b = 1$.

Now we must argue we can swap the $b = 0$ case with $b = 1$ in $\mathbf{G}_2$. To do this we use another game hop, to a game $\mathbf{G}_3$ that we do not draw out: $\mathbf{G}_3$ is identical to $\mathbf{G}_2$ except that $\mathtt{U}_1$ returns $c_{1-\mathbf{b}_{sid}}$ in place of $c_{\mathbf{b}_{sid}}$, so the $\mathtt{U}_1$ call for session *sid* in effect returns the *other* session's challenge.

We first claim $\mathbf{G}_2$ and $\mathbf{G}_3$ induce the same transcript distribution at $b = 0$—that is, swapping the two challenges has no effect. Fix every random value in the game except δ_0, δ_1. Each challenge is $c_i = c_{t,i} + \delta_i$, and δ_i is a fresh uniform value that appears nowhere else in the game; in particular, by the substitution made in $\mathbf{G}_1$, it does not appear in σ_i. Hence, with everything else fixed, c_0 and c_1 are independent and uniform over $\mathbb{Z}_p$, and independent of (σ_0, σ_1). The two sessions therefore receive a pair of independent uniform challenges whether $\mathtt{U}_1$ hands out (c_0, c_1) (as in $\mathbf{G}_2$) or (c_1, c_0) (as in $\mathbf{G}_3$); and since $\mathcal{A}$'s remaining messages and random-oracle queries are deterministic functions of the values it receives, its entire transcript is identically distributed in the two games. Thus $\Pr[\Delta_{\mathbf{G}_2} = \pi \wedge V | \rho, b = 0] = \Pr[\Delta_{\mathbf{G}_3} = \pi \wedge V | \rho, b = 0]$.

$\underline{\mathbf{G}_1}$:

$\mathsf{sk}_S \leftarrow\!\!\$\ \mathbb{Z}_p$; $\mathsf{pk}_S \leftarrow G^{\mathsf{sk}_S}$

$b \leftarrow\!\!\$\ \{0,1\}$; $\mathbf{b}_1 \leftarrow b$; $\mathbf{b}_2 \leftarrow 1-b$

$\mathbf{sess}_1 \leftarrow \mathbf{sess}_2 \leftarrow \mathsf{init}$

Sample fresh per-session randomness for $sid \in \{1,2\}$:

$\chi_{sid} \leftarrow\!\!\$\ \mathbb{Z}_p$; $T'_{sid} \leftarrow G^{\chi_{sid}}$

$\delta_{sid}, \alpha_{sid}, \beta_{sid}, a_{v,sid}, \tilde{c}_{u,sid}, \tilde{z}_{u,sid}, \tilde{c}_{w,sid}, \tilde{z}_{w,sid} \leftarrow\!\!\$\ \mathbb{Z}_p$

$b' \leftarrow\!\!\$\ \mathcal{A}^{\mathsf{init},\mathtt{U}_1,\mathtt{U}_2,H}(\mathsf{pk}_S, \mathsf{sk}_S)$

Return $b' = b$

$\underline{\mathsf{init}((\hat{m}_0, \hat{\mathsf{pk}}_{R,0}, \hat{\mathsf{pk}}_{J,0}), (\hat{m}_1, \hat{\mathsf{pk}}_{R,1}, \hat{\mathsf{pk}}_{J,1}))}$:

$(m_0, \mathsf{pk}_{R,0}, \mathsf{pk}_{J,0}) \leftarrow (\hat{m}_0, \hat{\mathsf{pk}}_{R,0}, \hat{\mathsf{pk}}_{J,0})$

$(m_1, \mathsf{pk}_{R,1}, \mathsf{pk}_{J,1}) \leftarrow (\hat{m}_1, \hat{\mathsf{pk}}_{R,1}, \hat{\mathsf{pk}}_{J,1})$

$\mathbf{sess}_1 \leftarrow \mathbf{sess}_2 \leftarrow$ open-pending

$\underline{\mathtt{U}_1(sid, T_{sid})}$:

If $sid \notin \{1,2\} \vee \mathbf{sess}_{sid} \neq$ open-pending: Return $\perp$

$\mathbf{sess}_{sid} \leftarrow \mathsf{open}$

$J_{sid} \leftarrow \mathsf{pk}_{J,\mathbf{b}_{sid}}^{\alpha_{sid}}$; $E_{J,sid} \leftarrow G^{\alpha_{sid}}$

$R_{sid} \leftarrow \mathsf{pk}_{R,\mathbf{b}_{sid}}^{\beta_{sid}}$; $E_{R,sid} \leftarrow G^{\beta_{sid}}$

$\widetilde{A}_{u,sid} \leftarrow G^{\tilde{z}_{u,sid}} \cdot J_{sid}^{-\tilde{c}_{u,sid}}$

$\widetilde{A}_{w,sid} \leftarrow G^{\tilde{z}_{w,sid}} \cdot R_{sid}^{-\tilde{c}_{w,sid}}$

$A_{v,1,sid} \leftarrow G^{a_{v,sid}}$; $A_{v,2,sid} \leftarrow \mathsf{pk}_{J,\mathbf{b}_{sid}}^{a_{v,sid}}$

$L_{sid} \leftarrow (J_{sid}, R_{sid}, E_{J,sid}, E_{R,sid}, \widetilde{A}_{u,sid}, T'_{sid}, \widetilde{A}_{w,sid}, A_{v,1,sid}, A_{v,2,sid})$

$c_{\perp,sid} \leftarrow H(\mathsf{pk}_S, \mathsf{pk}_{R,\mathbf{b}_{sid}}, \mathsf{pk}_{J,\mathbf{b}_{sid}}, L_{sid}, m_{\mathbf{b}_{sid}})$

$c_{t,sid} \leftarrow c_{\perp,sid} - \tilde{c}_{u,sid}$; $c_{v,sid} \leftarrow c_{\perp,sid} - \tilde{c}_{w,sid}$

$c_{sid} \leftarrow c_{t,sid} + \delta_{sid}$

Save $\mathsf{state}_{sid} \leftarrow (sid, \mathbf{b}_{sid}, T_{sid}, L_{sid}, c_{sid}, c_{t,sid}, c_{v,sid})$

Return (sid, c_{sid})

$\underline{\mathtt{U}_2(sid, s_{sid})}$:

If $\mathbf{sess}_{sid} \neq$ open: Return $\perp$

$\mathbf{sess}_{sid} \leftarrow$ closed

$(sid, \mathbf{b}_{sid}, T_{sid}, L_{sid}, c_{sid}, c_{t,sid}, c_{v,sid}) \leftarrow \mathsf{state}_{sid}$

$z_{t,sid} \leftarrow \chi_{sid} + c_{t,sid}\mathsf{sk}_S$

$z_{v,sid} \leftarrow a_{v,sid} + c_{v,sid} \cdot \alpha_{sid}$

$\sigma_{\mathbf{b}_{sid}} \leftarrow (L_{sid}, (c_{t,sid}, z_{t,sid}, \tilde{z}_{u,sid}), (c_{v,sid}, z_{v,sid}, \tilde{z}_{w,sid}))$

If $\mathbf{sess}_1 = \mathbf{sess}_2 =$ closed:

Return (σ_0, σ_1)

Return $(sid,$ closed$)$

$\underline{H(X, Y, Z, \vec{L}, m)}$:

If $\mathcal{T}[X, Y, Z, \vec{L}, m] = \perp$: $\mathcal{T}[X, Y, Z, \vec{L}, m] \leftarrow\!\!\$\ \mathbb{Z}_p$

Return $\mathcal{T}[X, Y, Z, \vec{L}, m]$

Figure 7: Game $\mathbf{G}_1$ used in the proof of Theorem 1.

$\underline{\mathbf{G}_2}$:

$\mathsf{sk}_S \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ \mathsf{pk}_S \leftarrow G^{\mathsf{sk}_S}$

$b \leftarrow\!\!\$\ \{0,1\}\ ;\ \mathbf{b}_1 \leftarrow b\ ;\ \mathbf{b}_2 \leftarrow 1-b$

$\mathbf{sess}_1 \leftarrow \mathbf{sess}_2 \leftarrow \mathsf{init}$

For $i \in \{0,1\}$:

  $\chi_i \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ T'_i \leftarrow G^{\chi_i}$

  $\delta_i, \alpha_i, \beta_i, a_{v,i}, \tilde{c}_{u,i}, \tilde{z}_{u,i}, \tilde{c}_{w,i}, \tilde{z}_{w,i} \leftarrow\!\!\$\ \mathbb{Z}_p$

$b' \leftarrow\!\!\$\ \mathcal{A}^{\mathsf{init},\mathtt{U}_1,\mathtt{U}_2,H}(\mathsf{pk}_S, \mathsf{sk}_S)$

Return $b' = b$

$\underline{\mathsf{init}((\hat{m}_0, \hat{\mathsf{pk}}_{R,0}, \hat{\mathsf{pk}}_{J,0}), (\hat{m}_1, \hat{\mathsf{pk}}_{R,1}, \hat{\mathsf{pk}}_{J,1}))}$:

$(m_0, \mathsf{pk}_{R,0}, \mathsf{pk}_{J,0}) \leftarrow (\hat{m}_0, \hat{\mathsf{pk}}_{R,0}, \hat{\mathsf{pk}}_{J,0})$

$(m_1, \mathsf{pk}_{R,1}, \mathsf{pk}_{J,1}) \leftarrow (\hat{m}_1, \hat{\mathsf{pk}}_{R,1}, \hat{\mathsf{pk}}_{J,1})$

For $i \in \{0,1\}$:

  $J_i \leftarrow \mathsf{pk}_{J,i}^{\alpha_i}\ ;\ E_{J,i} \leftarrow G^{\alpha_i}$

  $R_i \leftarrow \mathsf{pk}_{R,i}^{\beta_i}\ ;\ E_{R,i} \leftarrow G^{\beta_i}$

  $\widetilde{A}_{u,i} \leftarrow G^{\tilde{z}_{u,i}} \cdot J_i^{-\tilde{c}_{u,i}}$

  $\widetilde{A}_{w,i} \leftarrow G^{\tilde{z}_{w,i}} \cdot R_i^{-\tilde{c}_{w,i}}$

  $A_{v,1,i} \leftarrow G^{a_{v,i}}\ ;\ A_{v,2,i} \leftarrow \mathsf{pk}_{J,i}^{a_{v,i}}$

  $L_i \leftarrow (J_i, R_i, E_{J,i}, E_{R,i}, \widetilde{A}_{u,i}, T'_i, \widetilde{A}_{w,i}, A_{v,1,i}, A_{v,2,i})$

  $c_{\perp,i} \leftarrow H(\mathsf{pk}_S, \mathsf{pk}_{R,i}, \mathsf{pk}_{J,i}, L_i, m_i)$

  $c_{t,i} \leftarrow c_{\perp,i} - \tilde{c}_{u,i}\ ;\ c_{v,i} \leftarrow c_{\perp,i} - \tilde{c}_{w,i}$

  $c_i \leftarrow c_{t,i} + \delta_i$

  $z_{t,i} \leftarrow \chi_i + c_{t,i}\mathsf{sk}_S$

  $z_{v,i} \leftarrow a_{v,i} + c_{v,i} \cdot \alpha_i$

  $\sigma_i \leftarrow (L_i, (c_{t,i}, z_{t,i}, \tilde{z}_{u,i}), (c_{v,i}, z_{v,i}, \tilde{z}_{w,i}))$

$\mathbf{sess}_1 \leftarrow \mathbf{sess}_2 \leftarrow \mathsf{open\text{-}pending}$

$\underline{\mathtt{U}_1(sid, T_{sid})}$:

If $sid \notin \{1,2\} \vee \mathbf{sess}_{sid} \neq \mathsf{open}$-pending: Return $\perp$

$\mathbf{sess}_{sid} \leftarrow \mathsf{open}$

Return $(sid, c_{\mathbf{b}_{sid}})$

$\underline{\mathtt{U}_2(sid, s_{sid})}$:

If $\mathbf{sess}_{sid} \neq \mathsf{open}$: Return $\perp$

$\mathbf{sess}_{sid} \leftarrow \mathsf{closed}$

If $\mathbf{sess}_1 = \mathbf{sess}_2 = \mathsf{closed}$:

  Return (σ_0, σ_1)

Return (sid, closed)

$\underline{H(X, Y, Z, \vec{L}, m)}$:

If $\mathcal{T}[X, Y, Z, \vec{L}, m] = \perp$: $\mathcal{T}[X, Y, Z, \vec{L}, m] \leftarrow\!\!\$\ \mathbb{Z}_p$

Return $\mathcal{T}[X, Y, Z, \vec{L}, m]$

Figure 8: Game $\mathbf{G}_2$ used in the proof of Theorem 1.

Next we observe that $\mathbf{G}_3$ at $b = 0$ is the *same game* as $\mathbf{G}_2$ at $b = 1$. Under $b = 0$ we have $\mathbf{b}_1 = 0$ and $\mathbf{b}_2 = 1$, so $\mathbf{G}_3$ returns $c_{1-\mathbf{b}_1} = c_1$ to session 1 and $c_{1-\mathbf{b}_2} = c_0$ to session 2; under $b = 1$ we have $\mathbf{b}_1 = 1$ and $\mathbf{b}_2 = 0$, so $\mathbf{G}_2$ returns $c_{\mathbf{b}_1} = c_1$ to session 1 and $c_{\mathbf{b}_2} = c_0$ to session 2—the identical wiring. Since the precomputed values $c_0, c_1, \sigma_0, \sigma_1$ are produced in init independently of b, the two games are identical, and so $\Pr[\Delta_{\mathbf{G}_3} = \pi \wedge V | \rho, b = 0] = \Pr[\Delta_{\mathbf{G}_2} = \pi \wedge V | \rho, b = 1]$.

Chaining these two equalities gives $\Pr[\Delta_{\mathbf{G}_2} = \pi \wedge V | \rho, b = 0] = \Pr[\Delta_{\mathbf{G}_2} = \pi \wedge V | \rho, b = 1]$ for every π. The hops establishing $\Delta_{\mathbf{G}_0} \equiv \Delta_{\mathbf{G}_1} \equiv \Delta_{\mathbf{G}_2}$ hold at each fixed value of b, so this equality transfers back to $\mathbf{G}_0$: $\Pr[\Delta_{\mathbf{G}_0} = \pi \wedge V | \rho, b = 0] = \Pr[\Delta_{\mathbf{G}_0} = \pi \wedge V | \rho, b = 1]$. Summing over the transcripts π that cause $\mathcal{A}$ to output 1 recovers $\Pr[E_1 \wedge V | \rho, b = 0] = \Pr[E_1 \wedge V | \rho, b = 1]$; combined with the equality for the $\neg V$ case established above and averaging over ρ, we conclude $\Pr[E_1 | b = 0] = \Pr[E_1 | b = 1]$. We remark that the argument makes no well-formedness assumption on the adversarially chosen keys $\mathsf{pk}_{R,i}, \mathsf{pk}_{J,i}$, treating them as arbitrary group elements, so perfect VH-blindness holds even for malformed identity keys. □

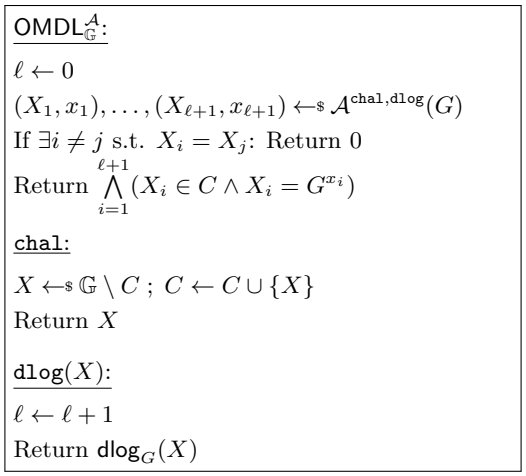

Figure 9: One-more discrete logarithm (OMDL) game for adversary $\mathcal{A}$ and group $\mathbb{G}$ of order p generated by G. The function $\mathsf{dlog}_G(X)$ returns the base-G discrete log of X. We depart from [54] in one respect: `chal` samples *without replacement* ($X \leftarrow\!\!\$\ \mathbb{G} \setminus C$), so the challenges are distinct by construction. This rules out an edge case their set-based formulation leaves open—a repeated challenge shrinks C below $\ell+1$ and precludes any winning output—and lets our reduction avoid an additive $\binom{\ell+1}{2}/p$ collision term that is an artifact of the game rather than a property of BAMF. (Formulations that issue an indexed challenge list, e.g. [42], sidestep the issue the same way.)

Unforgeability. We next turn to the sequential one-more unforgeability of BAMF. Our result is a reduction to the one-more discrete logarithm assumption. We use the concrete OMDL assumption from [54], depicted in Figure 9. Our result is in the algebraic group model [41], a setting common in prior work on discrete-logarithm-based blind signatures.

Theorem 2. *Let $\mathbb{G}$ be a group of prime order p. Let $\mathcal{A}$ be an algebraic adversary in game* dvOMUNF *in Figure 2—instantiated over $\mathbb{G}$ with protocol* BAMF *in Figure 5—that closes at most ℓ signing sessions and makes at most q_h random oracle queries, and that queries H on each of its output forgeries. Then we give a concrete OMDL adversary $\mathcal{B}$ (Figure 11) and concrete DL adversaries $\mathcal{C}$ (Figure 12) and $\mathcal{D}$ (Figure 13) such that*

$$\mathsf{Adv}^{\mathsf{dvomunf}}_{\mathsf{BAMF},\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{omdl}}_{\mathcal{B},\mathbb{G}} + \mathsf{Adv}^{\mathsf{dl}}_{\mathcal{C},\mathbb{G}} + \mathsf{Adv}^{\mathsf{dl}}_{\mathcal{D},\mathbb{G}} + \frac{3q_h^2 + q_h}{p}\ .$$

Proof overview. We begin with a brief, high-level overview of the proof. The overall approach follows the analysis of sequential OMUNF of the plain blind Schnorr protocol in [54]. The main goal of the proof is to show that an algebraic adversary that wins sequential OMUNF must have extracted either the sender's, receiver's, or judge's secret key. The security proof *must* argue that any of these keys can be extracted because knowledge of any one of them allows creation of verifying AMF signatures—this is by design and how the AMF scheme achieves deniability. These three cases correspond to our three reductions $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$. Since the adversary only gets a (blind) signing oracle for the sender secret key, only the sender key reduction $\mathcal{B}$ plays the one-more discrete logarithm game; the other two reductions are to plain discrete log. Each reduction works by getting a group element challenge, using it as the public key whose corresponding secret key is the scalar they want to extract, then simulating the rest of $\mathcal{A}$'s interactions with the BAMF sequential OMUNF game. The sender secret reduction $\mathcal{B}$ uses the same strategy as [54]—it calls its challenge oracle to get S_1 outputs T_i, and on input c_i to oracle S_2 calls its DL oracle on group element $T_i \mathsf{pk}_S^{c_i}$. The judge and receiver secret reductions $\mathcal{C}$ and $\mathcal{D}$ know the sender secret, so they run the exact code of the OMUNF oracles for BAMF. When $\mathcal{A}$ outputs its forgeries, it also outputs the representations of all the group elements in the signatures. At this point, we want to be able to argue that if all $\mathcal{A}$'s forgeries pass verification, each reduction can extract its desired discrete log from the representations. This is almost true, but there is a subtle failure case: even though the signatures pass verification, the corresponding representations can be "degenerate" and contain no information about secrets. The main difficulty of the proof is giving an upper-bound on this

failure case. This turns out to be substantially more difficult than the corresponding argument in [54], for several reasons. First, an AMF signature is a Boolean composition—an AND of two ORs—of Schnorr-style knowledge statements of the form $G^z = A \cdot Y^c$, rather than a single such statement. Second, some of the group elements standing in the "key" position Y of these statements are adversarial, whereas in plain blind Schnorr the only adversarial group element is the forgery's commitment. Worse still, there is an adversarial group element (J) that actually appears in *both* OR-clauses; this couples their representations together in a way that bars a clean modular analysis. Third, because AMF signatures use CDS OR-proofs, the per-clause challenges are adversarially chosen subject only to an additive constraint, rather than being fixed directly by the random-oracle output of the "special" query. Together, these mean that the hardest case of [54]—two special queries falling in one open session, from which they derive an instance of the 1-ROS problem—here yields nonlinear (quadratic) constraints on the adversary's choices in place of their linear ones.

A further consequence of the OR structure is that it is not a priori clear that AMF signatures have the unique-response property: an adversary might hope to turn a single "special" query into two distinct forgeries by splitting the same RO output in two different ways. Blind Schnorr rules this out essentially for free, since there the forgery challenge is the RO output itself and Schnorr responses are unique—z is determined by the commitment T, the public key, and the challenge c.

Another wrinkle is that AMF verification contains the non-Schnorr-like statement $R = E_R^{\mathsf{sk}_R}$. Since E_R's representation may itself involve $\mathsf{pk}_R = G^{\mathsf{sk}_R}$, the induced constraint is quadratic in sk_R; moreover, this equation contains no challenge at all, so it is not clear how to argue non-degeneracy through the choice of a challenge.

A final complication is that, unlike in plain blind Schnorr, it is trivial for an adversary to output representations that block extraction of any *single* key. Consequently, the natural modular approach of splitting the bad event into three per-key sub-events and applying a union bound is destined to fail: it can only ever give the trivial upper bound of 1 on the bad-event probability.

Ultimately, our proof tames this complexity using a couple of ideas. First, we observe that the technical core of the bad-event analysis can be stripped down to a single case, which we call the "hard case"; in a precise sense, it describes the most restricted adversary that can still exploit all of these subtleties. Notably, in this case $\ell = 1$, so the adversary outputs only two forgeries. With just two forgeries it is tractable to write all the constraints on the output representations as one linear system over a short vector of key monomials, with the property that the "bad" event corresponds to a designated subset of the entries being zero. We then work through the consequences of these entries being zero, and show that for each OR clause in a forgery we can derive a small system of "obstruction polynomials" with the property that the RO output for that forgery must be a common root in order for the bad event to happen. With these obstruction polynomials in hand, we complete the bad-event accounting for the hard case by root counting, together with one additional argument showing that the verification checks $E_J, E_R \neq 1$ prevent the obstruction polynomials from being trivial (either identically zero or constant).

Once this hard-case analysis is done, we briefly explain how to lift it to the full proof, relaxing our restrictions. The main idea is that for arbitrary ℓ, the hard-case analysis translates verbatim to any pair of forgery queries falling inside one session, and the bad-event probability then follows by a union bound over such pairs.

Proof. Our proof begins with game $\mathbf{G}_0$, with pseudocode in Figure 10. That game is a rewriting of $\mathsf{dvOMUNF}$ that substitutes the concrete BAMF blind signing and (receiver) verification procedures. Since $\mathcal{A}$ is an algebraic adversary, when it outputs an element X of $\mathbb{G}$ it must also output with it a representation—concretely, a vector of scalars $\vec{x}$ in $\mathbb{Z}_p$—of X in terms of all the group elements it is given by the $\mathsf{dvOMUNF}$ game. These elements include the generator G, three public keys $\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J$, and the group elements returned from $\mathtt{S}_1$. As argued by [54], we can assume wlog that $\mathcal{A}$ opens $\ell + 1$ sessions, and (since we are in the sequential setting) closes the first ℓ of them. Thus, the adversary is given $\ell+1$ total group elements from $\mathtt{S}_1$—call them $T_1, \ldots, T_{\ell+1}$. By the theorem's hypothesis, $\mathcal{A}$ calls $H(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, L^{(i)}, m_i)$ on every forgery (m_i, σ_i) it outputs, where $L^{(i)}$ is the vector of group elements for σ_i. (Call $c^{(i)}$ the output of this RO query; we call this the *special query* of forgery i, and take it to be the first H-query on $(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, L^{(i)}, m_i)$, since repeats return the cached value. The hypothesis is standard and without real loss of generality: an unrestricted adversary can be wrapped to make any missing queries itself just before producing its output, increasing q_h by at most $\ell+1$.) Throughout, whenever we speak of the AGM representation of a group element

$\underline{\mathbf{G}_0 := \mathsf{dvOMUNF}^{\mathcal{A}}_{\mathsf{BAMF}}}$:

$(\mathsf{pk}_S, \mathsf{sk}_S, \mathsf{pk}_R, \mathsf{sk}_R, \mathsf{pk}_J, \mathsf{sk}_J) \leftarrow\!\!\$\ \mathsf{KGen}\ ;\ \ell \leftarrow 0\ ;\ b \leftarrow 1$
$(m_1, \sigma_1), \ldots, (m_{\ell+1}, \sigma_{\ell+1}) \leftarrow\!\!\$\ \mathcal{A}^{\mathsf{S}_1, \mathsf{S}_2}(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J)$
$\forall i : (L^{(i)}, (c_t^{(i)}, z_t^{(i)}, z_u^{(i)}), (c_v^{(i)}, z_v^{(i)}, z_w^{(i)})) \leftarrow \sigma_i$
$(J^{(i)}, R^{(i)}, E_J^{(i)}, E_R^{(i)}, A_t^{(i)}, A_u^{(i)}, A_{v,1}^{(i)}, A_{v,2}^{(i)}, A_w^{(i)}) \leftarrow L^{(i)}$
If $\exists i \neq j$ such that $(m_i, \sigma_i) = (m_j, \sigma_j)$: Return 0
For $i = 1$ to $\ell + 1$:
  $c^{(i)} \leftarrow H(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, L^{(i)}, m_i)$
  $c_u^{(i)} \leftarrow c^{(i)} - c_t^{(i)}$
  $c_w^{(i)} \leftarrow c^{(i)} - c_v^{(i)}$
  $b \overset{\wedge}{\leftarrow} R^{(i)} = \left(E_R^{(i)}\right)^{\mathsf{sk}_R} \wedge E_R^{(i)} \neq 1$
  $b \overset{\wedge}{\leftarrow} G^{z_t^{(i)}} = A_t^{(i)} \cdot \mathsf{pk}_S^{c_t^{(i)}}$
  $b \overset{\wedge}{\leftarrow} G^{z_u^{(i)}} = A_u^{(i)} \cdot \left(J^{(i)}\right)^{c_u^{(i)}}$
  $b \overset{\wedge}{\leftarrow} G^{z_v^{(i)}} = A_{v,1}^{(i)} \cdot \left(E_J^{(i)}\right)^{c_v^{(i)}} \wedge E_J^{(i)} \neq 1$
  $b \overset{\wedge}{\leftarrow} \mathsf{pk}_J^{z_v^{(i)}} = A_{v,2}^{(i)} \cdot \left(J^{(i)}\right)^{c_v^{(i)}}$
  $b \overset{\wedge}{\leftarrow} G^{z_w^{(i)}} = A_w^{(i)} \cdot \left(R^{(i)}\right)^{c_w^{(i)}}$
Return b

$\underline{\mathsf{S}_1}$:

If $\mathsf{sess} = \mathsf{open}$: Return $\perp$
$\mathsf{sess} \leftarrow \mathsf{open}$
$t_{\ell+1} \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ T_{\ell+1} \leftarrow G^{t_{\ell+1}}$
$\mathsf{state} \leftarrow t_{\ell+1}$
Return $T_{\ell+1}$

$\underline{\mathsf{S}_2(c)}$:

If $\mathsf{sess} \neq \mathsf{open}$: Return $\perp$
$c_{\ell+1} \leftarrow c\ ;\ s_{\ell+1} \leftarrow \mathsf{state} + c\,\mathsf{sk}_S$
$\mathsf{sess} \leftarrow \mathsf{closed}$
$\ell \leftarrow \ell + 1$
Return s_ℓ

$\underline{H(A, B, C, \vec{L}, m)}$:

if $\mathcal{T}[A, B, C, \vec{L}, m] = \perp$:
  $x \leftarrow\!\!\$\ \mathbb{Z}_p$
  $\mathcal{T}[A, B, C, \vec{L}, m] \leftarrow x$
Return $\mathcal{T}[A, B, C, \vec{L}, m]$

Figure 10: Game $\mathsf{dvOMUNF}$ with concrete BAMF procedures substituted in for bsig procedures. The notation $\overset{\wedge}{\leftarrow}$ is the equivalent of AND-equals assignment of the left-hand side.

of forgery i, we mean the representation $\mathcal{A}$ supplies at that forgery's special query (which each reduction can recover from the random-oracle table via $(L^{(i)}, m_i)$); the AGM does not constrain how this relates to any other representation $\mathcal{A}$ might give the same element, and both the bad-event analysis and the reductions use only these query-time representations. Throughout the proof we use two indexing conventions: a superscript (i) indexes the $\ell + 1$ forgeries, while an integer subscript indexes signing sessions—so T_j, t_j, s_j, and the session challenge c_j all refer to the jth session. The adversary $\mathcal{A}$ outputs $\ell + 1$ signatures; each contains nine $\mathbb{G}$ elements and six scalars in $\mathbb{Z}_p$. These elements are $(J^{(i)}, R^{(i)}, E_J^{(i)}, E_R^{(i)}, A_t^{(i)}, A_u^{(i)}, A_{v,1}^{(i)}, A_{v,2}^{(i)}, A_w^{(i)})$ for the ith forgery σ_i. The ith forgery's six scalars are $(c_t^{(i)}, z_t^{(i)}, z_u^{(i)})$, $(c_v^{(i)}, z_v^{(i)}, z_w^{(i)})$. There are six equations checked during verification; each gives a relationship between the AGM representations of $\mathcal{A}$'s output group elements. (For $\mathcal{A}$ to win, *all* $6(\ell + 1)$ equations must hold.)

Building intuition with a special case. We begin with an argument for a special case of the theorem. The restrictions for this special case are, first, that $\ell = 1$, so the adversary only participates in one session and outputs two forgeries. The second restriction is that this adversary is *required* to make both RO queries for its forgeries *during the open session.* Third, the adversary is not allowed to use the last (unclosed) session's T_2 in any representation. Fourth and finally, the adversary makes exactly two random-oracle queries—the special queries of its two forgeries—and these queries are distinct. (We show below that the distinctness restriction excludes nothing.) Though we have restricted the adversary's behavior here, this restriction leaves intact the most nontrivial parts of the analysis. Thus, below we will call this the "hard case".

Let $i \in \{1, 2\}$ index the two forgeries. For every group element X appearing in forgery i, write its AGM representation as

$$X^{(i)} = G^{\gamma_X^{(i)}} \mathsf{pk}_S^{\kappa_{s,X}^{(i)}} \mathsf{pk}_R^{\kappa_{r,X}^{(i)}} \mathsf{pk}_J^{\kappa_{j,X}^{(i)}} T_1^{\tau_X^{(i)}} .$$

After the first session is closed, the element T_1 has a representation in terms of G and pk_S (namely $G^{s_1} = T_1 \mathsf{pk}_S^{c_1} \Rightarrow T_1 = G^{s_1} \mathsf{pk}_S^{-c_1}$), so we may redistribute the τ terms and rewrite the discrete log of X in the basis $(G, \mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J)$. We collect the four resulting coefficients into a single bracket notation: for a coordinate $c \in \{1, S, R, J\}$ (with $1 \mapsto G$, $S \mapsto \mathsf{pk}_S$, $R \mapsto \mathsf{pk}_R$, $J \mapsto \mathsf{pk}_J$), write $[X]_c$ for the coefficient of that

basis element, so that

$$\mathsf{dlog}_G X^{(i)} = [X^{(i)}]_1 + [X^{(i)}]_S\,\mathsf{sk}_S + [X^{(i)}]_R\,\mathsf{sk}_R + [X^{(i)}]_J\,\mathsf{sk}_J,$$

where

$$[X^{(i)}]_1 = \gamma_X^{(i)} + s_1\tau_X^{(i)}, \quad [X^{(i)}]_S = \kappa_{s,X}^{(i)} - c_1\tau_X^{(i)}, \quad [X^{(i)}]_R = \kappa_{r,X}^{(i)}, \quad [X^{(i)}]_J = \kappa_{j,X}^{(i)}.$$

Only the G- and pk_S-coordinates absorb the τ term; the pk_R- and pk_J-coordinates are the untouched AGM coefficients. This bracket is the notation we use throughout: the entries of the explainability matrix below and every quantity in the obstruction analysis are built from the $[\cdot]_c$.

Note two important facts: first, the game cannot perform this rewriting until the first session is closed; second, the exact representation of T_1 in terms of G and pk_S is *chosen by* $\mathcal{A}$ adaptively, as a function of the H outputs it gets for the queried forgeries, because it chooses the S_2 input c_1 and (by correctness) also its output s_1. This adaptive choice is the reason why this case is the hard one: since $\mathcal{A}$'s behavior is arbitrary, we must deal with arbitrarily bad choices of c_1 when arguing about a reduction's ability to win a discrete-log game: an arbitrarily bad choice could allow $\mathcal{A}$ to win without enabling a discrete log to be extracted from its forgeries. Thus, the main technical challenge of the remainder of the "hard case" is arguing that even an arbitrarily bad choice of c_1 cannot prevent extraction of all three secret keys. (Relatedly, the *lack* of adaptivity is exactly why we can more or less ignore the group element T_2—it is given as part of the last session, which is never closed, so the adversary never gets to make an adaptive choice based on it.)

Now define the monomial vector $\mathbf{m} := (1, \mathsf{sk}_S, \mathsf{sk}_R, \mathsf{sk}_J, \mathsf{sk}_S\mathsf{sk}_R, \mathsf{sk}_R^2, \mathsf{sk}_R\mathsf{sk}_J)^\top$. Intuitively, the first four entries correspond to the "normal" representation of an element in terms of $G, \mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J$; the last three entries come from cross-products that appear when analyzing extraction from OR clauses (because the OR clauses' challenges are correlated)

The explainability matrix E_{hc} is the 12 x 7 matrix over $\mathbb{Z}_p$ whose rows are the coefficients of the twelve verification constraints against $\mathbf{m}$. (Twelve because in this case the adversary outputs two forgeries, and each adds six rows to the matrix.) A winning hard-case transcript satisfies $\mathsf{E}_{\mathrm{hc}}\mathbf{m} = 0$. For each forgery, the first five rows are Schnorr-like verification equations of the form $B^z = A \cdot Y^q$ with $B \in \{G, \mathsf{pk}_J\}$; for these define

$$\begin{aligned} L(A, Y, q, B, z) &:= z \cdot \mathbf{1}[B = G] - [A]_1 - q[Y]_1 \\ \chi_S(A, Y, q) &:= [A]_S + q[Y]_S \\ \chi_R(A, Y, q) &:= [A]_R + q[Y]_R \\ \chi_J(A, Y, q, B, z) &:= [A]_J + q[Y]_J - z \cdot \mathbf{1}[B = \mathsf{pk}_J]\ . \end{aligned}$$

Each equation's matrix row is $\mathsf{lin}(A, Y, q, B, z) := (-L, \chi_S, \chi_R, \chi_J, 0, 0, 0)$. For the sixth equation, the receiver check $R = E_R^{\mathsf{sk}_R}$, define

$$\mathsf{rec}(R, E_R) := ([R]_1,\ [R]_S,\ [R]_R - [E_R]_1,\ [R]_J,\ -[E_R]_S,\ -[E_R]_R,\ -[E_R]_J)\ .$$

Then the full hard-case matrix is, with $c_u^{(i)} = c^{(i)} - c_t^{(i)}$ and $c_w^{(i)} = c^{(i)} - c_v^{(i)}$ as in Figure 10,

$$\mathsf{E}_{\mathrm{hc}} := \begin{pmatrix} \mathsf{lin}(A_t^{(1)}, \mathsf{pk}_S, c_t^{(1)}, G, z_t^{(1)}) \\ \mathsf{lin}(A_u^{(1)}, J^{(1)}, c_u^{(1)}, G, z_u^{(1)}) \\ \mathsf{lin}(A_{v,1}^{(1)}, E_J^{(1)}, c_v^{(1)}, G, z_v^{(1)}) \\ \mathsf{lin}(A_{v,2}^{(1)}, J^{(1)}, c_v^{(1)}, \mathsf{pk}_J, z_v^{(1)}) \\ \mathsf{lin}(A_w^{(1)}, R^{(1)}, c_w^{(1)}, G, z_w^{(1)}) \\ \mathsf{rec}(R^{(1)}, E_R^{(1)}) \\ \mathsf{lin}(A_t^{(2)}, \mathsf{pk}_S, c_t^{(2)}, G, z_t^{(2)}) \\ \mathsf{lin}(A_u^{(2)}, J^{(2)}, c_u^{(2)}, G, z_u^{(2)}) \\ \mathsf{lin}(A_{v,1}^{(2)}, E_J^{(2)}, c_v^{(2)}, G, z_v^{(2)}) \\ \mathsf{lin}(A_{v,2}^{(2)}, J^{(2)}, c_v^{(2)}, \mathsf{pk}_J, z_v^{(2)}) \\ \mathsf{lin}(A_w^{(2)}, R^{(2)}, c_w^{(2)}, G, z_w^{(2)}) \\ \mathsf{rec}(R^{(2)}, E_R^{(2)}) \end{pmatrix},$$

which expands entrywise to

$$\begin{pmatrix}
\bar{\gamma}_{A_t^{(1)}} - z_t^{(1)} & \bar{\kappa}_{s,A_t^{(1)}} + c_t^{(1)} & \kappa_{r,A_t^{(1)}} & \kappa_{j,A_t^{(1)}} & 0 & 0 & 0 \\
\bar{\gamma}_{A_u^{(1)}} + c_u^{(1)}\bar{\gamma}_{J^{(1)}} - z_u^{(1)} & \bar{\kappa}_{s,A_u^{(1)}} + c_u^{(1)}\bar{\kappa}_{s,J^{(1)}} & \kappa_{r,A_u^{(1)}} + c_u^{(1)}\kappa_{r,J^{(1)}} & \kappa_{j,A_u^{(1)}} + c_u^{(1)}\kappa_{j,J^{(1)}} & 0 & 0 & 0 \\
\bar{\gamma}_{A_{v,1}^{(1)}} + c_v^{(1)}\bar{\gamma}_{E_J^{(1)}} - z_v^{(1)} & \bar{\kappa}_{s,A_{v,1}^{(1)}} + c_v^{(1)}\bar{\kappa}_{s,E_J^{(1)}} & \kappa_{r,A_{v,1}^{(1)}} + c_v^{(1)}\kappa_{r,E_J^{(1)}} & \kappa_{j,A_{v,1}^{(1)}} + c_v^{(1)}\kappa_{j,E_J^{(1)}} & 0 & 0 & 0 \\
\bar{\gamma}_{A_{v,2}^{(1)}} + c_v^{(1)}\bar{\gamma}_{J^{(1)}} & \bar{\kappa}_{s,A_{v,2}^{(1)}} + c_v^{(1)}\bar{\kappa}_{s,J^{(1)}} & \kappa_{r,A_{v,2}^{(1)}} + c_v^{(1)}\kappa_{r,J^{(1)}} & \kappa_{j,A_{v,2}^{(1)}} + c_v^{(1)}\kappa_{j,J^{(1)}} - z_v^{(1)} & 0 & 0 & 0 \\
\bar{\gamma}_{A_w^{(1)}} + c_w^{(1)}\bar{\gamma}_{R^{(1)}} - z_w^{(1)} & \bar{\kappa}_{s,A_w^{(1)}} + c_w^{(1)}\bar{\kappa}_{s,R^{(1)}} & \kappa_{r,A_w^{(1)}} + c_w^{(1)}\kappa_{r,R^{(1)}} & \kappa_{j,A_w^{(1)}} + c_w^{(1)}\kappa_{j,R^{(1)}} & 0 & 0 & 0 \\
\bar{\gamma}_{R^{(1)}} & \bar{\kappa}_{s,R^{(1)}} & \kappa_{r,R^{(1)}} - \bar{\gamma}_{E_R^{(1)}} & \kappa_{j,R^{(1)}} & -\bar{\kappa}_{s,E_R^{(1)}} & -\kappa_{r,E_R^{(1)}} & -\kappa_{j,E_R^{(1)}} \\
\bar{\gamma}_{A_t^{(2)}} - z_t^{(2)} & \bar{\kappa}_{s,A_t^{(2)}} + c_t^{(2)} & \kappa_{r,A_t^{(2)}} & \kappa_{j,A_t^{(2)}} & 0 & 0 & 0 \\
\bar{\gamma}_{A_u^{(2)}} + c_u^{(2)}\bar{\gamma}_{J^{(2)}} - z_u^{(2)} & \bar{\kappa}_{s,A_u^{(2)}} + c_u^{(2)}\bar{\kappa}_{s,J^{(2)}} & \kappa_{r,A_u^{(2)}} + c_u^{(2)}\kappa_{r,J^{(2)}} & \kappa_{j,A_u^{(2)}} + c_u^{(2)}\kappa_{j,J^{(2)}} & 0 & 0 & 0 \\
\bar{\gamma}_{A_{v,1}^{(2)}} + c_v^{(2)}\bar{\gamma}_{E_J^{(2)}} - z_v^{(2)} & \bar{\kappa}_{s,A_{v,1}^{(2)}} + c_v^{(2)}\bar{\kappa}_{s,E_J^{(2)}} & \kappa_{r,A_{v,1}^{(2)}} + c_v^{(2)}\kappa_{r,E_J^{(2)}} & \kappa_{j,A_{v,1}^{(2)}} + c_v^{(2)}\kappa_{j,E_J^{(2)}} & 0 & 0 & 0 \\
\bar{\gamma}_{A_{v,2}^{(2)}} + c_v^{(2)}\bar{\gamma}_{J^{(2)}} & \bar{\kappa}_{s,A_{v,2}^{(2)}} + c_v^{(2)}\bar{\kappa}_{s,J^{(2)}} & \kappa_{r,A_{v,2}^{(2)}} + c_v^{(2)}\kappa_{r,J^{(2)}} & \kappa_{j,A_{v,2}^{(2)}} + c_v^{(2)}\kappa_{j,J^{(2)}} - z_v^{(2)} & 0 & 0 & 0 \\
\bar{\gamma}_{A_w^{(2)}} + c_w^{(2)}\bar{\gamma}_{R^{(2)}} - z_w^{(2)} & \bar{\kappa}_{s,A_w^{(2)}} + c_w^{(2)}\bar{\kappa}_{s,R^{(2)}} & \kappa_{r,A_w^{(2)}} + c_w^{(2)}\kappa_{r,R^{(2)}} & \kappa_{j,A_w^{(2)}} + c_w^{(2)}\kappa_{j,R^{(2)}} & 0 & 0 & 0 \\
\bar{\gamma}_{R^{(2)}} & \bar{\kappa}_{s,R^{(2)}} & \kappa_{r,R^{(2)}} - \bar{\gamma}_{E_R^{(2)}} & \kappa_{j,R^{(2)}} & -\bar{\kappa}_{s,E_R^{(2)}} & -\kappa_{r,E_R^{(2)}} & -\kappa_{j,E_R^{(2)}}
\end{pmatrix},$$

or, in the bracket notation $[X]_c$ used in the derivations below—with $[X]_1 = \bar{\gamma}_X = \gamma_X + s_1\tau_X$, $[X]_S = \bar{\kappa}_{s,X} = \kappa_{s,X} - c_1\tau_X$, $[X]_R = \kappa_{r,X}$, and $[X]_J = \kappa_{j,X}$—equivalently

$$\begin{pmatrix}
[A_t^{(1)}]_1 - z_t^{(1)} & [A_t^{(1)}]_S + c_t^{(1)} & [A_t^{(1)}]_R & [A_t^{(1)}]_J & 0 & 0 & 0 \\
[A_u^{(1)}]_1 + c_u^{(1)}[J^{(1)}]_1 - z_u^{(1)} & [A_u^{(1)}]_S + c_u^{(1)}[J^{(1)}]_S & [A_u^{(1)}]_R + c_u^{(1)}[J^{(1)}]_R & [A_u^{(1)}]_J + c_u^{(1)}[J^{(1)}]_J & 0 & 0 & 0 \\
[A_{v,1}^{(1)}]_1 + c_v^{(1)}[E_J^{(1)}]_1 - z_v^{(1)} & [A_{v,1}^{(1)}]_S + c_v^{(1)}[E_J^{(1)}]_S & [A_{v,1}^{(1)}]_R + c_v^{(1)}[E_J^{(1)}]_R & [A_{v,1}^{(1)}]_J + c_v^{(1)}[E_J^{(1)}]_J & 0 & 0 & 0 \\
[A_{v,2}^{(1)}]_1 + c_v^{(1)}[J^{(1)}]_1 & [A_{v,2}^{(1)}]_S + c_v^{(1)}[J^{(1)}]_S & [A_{v,2}^{(1)}]_R + c_v^{(1)}[J^{(1)}]_R & [A_{v,2}^{(1)}]_J + c_v^{(1)}[J^{(1)}]_J - z_v^{(1)} & 0 & 0 & 0 \\
[A_w^{(1)}]_1 + c_w^{(1)}[R^{(1)}]_1 - z_w^{(1)} & [A_w^{(1)}]_S + c_w^{(1)}[R^{(1)}]_S & [A_w^{(1)}]_R + c_w^{(1)}[R^{(1)}]_R & [A_w^{(1)}]_J + c_w^{(1)}[R^{(1)}]_J & 0 & 0 & 0 \\
[R^{(1)}]_1 & [R^{(1)}]_S & [R^{(1)}]_R - [E_R^{(1)}]_1 & [R^{(1)}]_J & -[E_R^{(1)}]_S & -[E_R^{(1)}]_R & -[E_R^{(1)}]_J \\
[A_t^{(2)}]_1 - z_t^{(2)} & [A_t^{(2)}]_S + c_t^{(2)} & [A_t^{(2)}]_R & [A_t^{(2)}]_J & 0 & 0 & 0 \\
[A_u^{(2)}]_1 + c_u^{(2)}[J^{(2)}]_1 - z_u^{(2)} & [A_u^{(2)}]_S + c_u^{(2)}[J^{(2)}]_S & [A_u^{(2)}]_R + c_u^{(2)}[J^{(2)}]_R & [A_u^{(2)}]_J + c_u^{(2)}[J^{(2)}]_J & 0 & 0 & 0 \\
[A_{v,1}^{(2)}]_1 + c_v^{(2)}[E_J^{(2)}]_1 - z_v^{(2)} & [A_{v,1}^{(2)}]_S + c_v^{(2)}[E_J^{(2)}]_S & [A_{v,1}^{(2)}]_R + c_v^{(2)}[E_J^{(2)}]_R & [A_{v,1}^{(2)}]_J + c_v^{(2)}[E_J^{(2)}]_J & 0 & 0 & 0 \\
[A_{v,2}^{(2)}]_1 + c_v^{(2)}[J^{(2)}]_1 & [A_{v,2}^{(2)}]_S + c_v^{(2)}[J^{(2)}]_S & [A_{v,2}^{(2)}]_R + c_v^{(2)}[J^{(2)}]_R & [A_{v,2}^{(2)}]_J + c_v^{(2)}[J^{(2)}]_J - z_v^{(2)} & 0 & 0 & 0 \\
[A_w^{(2)}]_1 + c_w^{(2)}[R^{(2)}]_1 - z_w^{(2)} & [A_w^{(2)}]_S + c_w^{(2)}[R^{(2)}]_S & [A_w^{(2)}]_R + c_w^{(2)}[R^{(2)}]_R & [A_w^{(2)}]_J + c_w^{(2)}[R^{(2)}]_J & 0 & 0 & 0 \\
[R^{(2)}]_1 & [R^{(2)}]_S & [R^{(2)}]_R - [E_R^{(2)}]_1 & [R^{(2)}]_J & -[E_R^{(2)}]_S & -[E_R^{(2)}]_R & -[E_R^{(2)}]_J
\end{pmatrix}.$$

We refer to the six rows of each forgery's block, in order, as its t-, u-, v_1-, v_2-, w-, and receiver rows; the derivations below can be checked entry by entry against this expanded form.

To begin, we claim that a secret key can be *extracted* from any accepting hard-case transcript in which either (a) some Schnorr-like row $(-L, \chi_S, \chi_R, \chi_J, 0, 0, 0)$ has $(\chi_S, \chi_R, \chi_J) \neq (0, 0, 0)$, or (b) the polynomial in sk_R induced by some receiver row—a notion we make precise below—is not identically zero. Here extraction means: an algorithm given the transcript, the AGM representations, and two of the three secret keys can compute the third. (In the full argument at the end of the proof we define three reductions that realize these extractions for arbitrary ℓ, each embedding a discrete-log challenge in one of the three keys and generating the other two itself; see Figures 11–13. For now we need only the extraction arithmetic.) Below we refer to a row satisfying (a) or (b) as *live*. Consequently, if no key can be extracted from an accepting hard-case transcript, then every Schnorr-like row has zero key-coefficient vector and both receiver rows induce identically zero polynomials. To see why, consider first a Schnorr-like row $(-L, \chi_S, \chi_R, \chi_J, 0, 0, 0)$. The corresponding scalar equation is

$$-L + \chi_S\mathsf{sk}_S + \chi_R\mathsf{sk}_R + \chi_J\mathsf{sk}_J = 0.$$

If $\chi_S \neq 0$, then given sk_R and sk_J one computes

$$\mathsf{sk}_S = \frac{L - \chi_R\mathsf{sk}_R - \chi_J\mathsf{sk}_J}{\chi_S}.$$

If $\chi_S = 0$ but $\chi_R \neq 0$, then given sk_S and sk_J one computes

$$\mathsf{sk}_R = \frac{L - \chi_S\mathsf{sk}_S - \chi_J\mathsf{sk}_J}{\chi_R}.$$

If $\chi_S = \chi_R = 0$ and $\chi_J \neq 0$, then given sk_S and sk_R one computes

$$\mathsf{sk}_J = \frac{L - \chi_S \mathsf{sk}_S - \chi_R \mathsf{sk}_R}{\chi_J}.$$

Thus any nonzero Schnorr-like key-coefficient vector gives extraction, which proves case (a); we have to show the same fact about the receiver rows $\mathsf{rec}(R, E_R)$. First we make precise the polynomial a receiver row induces. The inner product of a receiver row with $\mathbf{m}$ is a polynomial in $\mathsf{sk}_S, \mathsf{sk}_R, \mathsf{sk}_J$; substituting in the values of sk_S and sk_J (the two keys available to an extractor of sk_R) and collecting by powers of sk_R leaves a univariate polynomial in sk_R of degree at most two—the polynomial *induced* by the row. (Note that whether it is zero depends on the values $\mathsf{sk}_S, \mathsf{sk}_J$, not on the row's coefficient vector alone.) Explicitly—reading the receiver row of the expanded matrix above against $\mathbf{m}$ and collecting by powers of sk_R—the polynomial induced by $\mathsf{rec}(R, E_R)$ is

$$\begin{aligned}\Pi_{R,E_R}(Y) := {} & [R]_1 + [R]_S \mathsf{sk}_S + [R]_J \mathsf{sk}_J \\ & + \big([R]_R - [E_R]_1 - [E_R]_S \mathsf{sk}_S - [E_R]_J \mathsf{sk}_J\big) Y - [E_R]_R Y^2.\end{aligned}$$

Acceptance implies $\Pi_{R,E_R}(\mathsf{sk}_R) = 0$; this is a rewriting of the corresponding row of the matrix. In particular, on an accepting transcript the true sk_R is itself a root of Π_{R,E_R}; so if Π_{R,E_R} is not the zero polynomial, it has at least one and at most two roots in $\mathbb{Z}_p$, and the extractor computes them all and identifies sk_R by testing each root y against $\mathsf{pk}_R = G^y$. Note that this extraction is sound even for a party that cannot itself evaluate the receiver check $R = E_R^{\mathsf{sk}_R}$—which will be the situation of the reduction extracting sk_R in the full argument: it simply attempts the extraction unconditionally, and the root test rules out false positives. If Π_{R,E_R} is nonzero but has no root in $\mathbb{Z}_p$—or has roots, none matching pk_R—then $\Pi_{R,E_R}(\mathsf{sk}_R) \neq 0$, so the transcript was not accepting, and such transcripts contribute nothing to the winning probability an extractor must cover. Therefore, if no key can be extracted from an accepting transcript, then no Schnorr-like row is live and no receiver row is live. The remainder of the argument is showing that the probability $\mathcal{A}$ wins in the "hard case" but outputs forgeries that give no live rows is negligible.

We have arrived at the technical core of the hard case: bounding the probability that all twelve rows are dead. Before diving in, we fix a convention separating the two kinds of quantities in play, according to *when they are determined* relative to forgery i's special query:
Committed quantities are determined before the special query returns: the representation γ_X, $\kappa_{s,X}$, $\kappa_{r,X}$, $\kappa_{j,X}$, τ_X of every element X of $L^{(i)}$ (in the query input), and the game's own randomness—the three keys and the commitment exponent t_1, all sampled before the session's queries begin.
Post-hoc quantities are chosen by the adversary only after it has seen the hash values: the shared session challenge c_1 (and with it the response $s_1 = t_1 + c_1 \mathsf{sk}_S$), the per-forgery splits $c_t^{(i)}, c_v^{(i)}$, and the responses $z_t^{(i)}, z_u^{(i)}, z_v^{(i)}, z_w^{(i)}$.

The hash value $c^{(i)}$ itself belongs to neither class: it is fresh randomness drawn at the query, and it is against this randomness that we will measure probabilities. The analysis below is organized entirely around this split. Every dead-row condition will be written as a polynomial equation whose *coefficients* are committed and whose *variables* are post-hoc quantities or the hash. The post-hoc quantities are then dealt with one at a time: the responses each absorb a single coordinate, the splits are eliminated (one by substitution, one by resultants), and the lone survivor is c_1—the adversary's real leverage, shared across both forgeries.

Concretely, the analysis proceeds in three steps. *Step 1:* for each forgery and each OR clause, compress deadness of the clause's rows into a family of *obstruction polynomials* with committed coefficients, in the variables λ (standing for c_1) and h (the forgery's hash). *Step 2:* show that unless a key can be extracted or a verification side condition fails, each forgery's obstruction system retains genuine h-dependence at $\lambda = c_1$; this is where the receiver clause and the checks $E_R \neq 1$, $E_J \neq 1$ do their work, and it is also where the two OR clauses interlock rather than being handled separately. *Step 3* (the completion, below): root-counting—the earlier forgery's obstructions pin c_1 to at most two candidate values, and the later forgery's fresh hash then kills each candidate with probability $1/p$.

Let λ denote a formal variable for the session challenge c_1. We un-freeze c_1 to λ and regard each coordinate $[X]_c$ as a polynomial in λ. Only the G- and pk_S-coordinates move:

$$[X]_S = \kappa_{s,X} - \lambda \tau_X, \qquad [X]_1 = (\gamma_X + t_1 \tau_X) + \lambda\, \mathsf{sk}_S\, \tau_X$$

(the latter via $s_1 = t_1 + \lambda \mathsf{sk}_S$), while $[X]_R = \kappa_{r,X}$ and $[X]_J = \kappa_{j,X}$ are committed constants. The realized transcript is $\lambda = c_1$, at which $[X]_c$ takes the value appearing in row X of E_{hc}. We write $[X]_\bullet := ([X]_S, [X]_R, [X]_J)$ for the triple of key coordinates of X. Fix forgery i, and write $h := c^{(i)}$, $d := c_t^{(i)}$. (The convention here is the same one that introduced λ: transcript scalars keep their game names, and we introduce a fresh letter exactly when a quantity is about to be treated as a variable of a polynomial—h for forgery i's hash value, and d and, below, e for the two splits, which we eliminate by substitution and by the pairwise minors respectively.) Mind the difference in scope: λ is *global*, since the two forgeries' obstruction systems constrain the one shared session challenge, whereas h, d, and e are *local* to the forgery under consideration—in particular, P_1^L and P_2^L do *not* share an h variable. We nevertheless leave h unsubscripted: Steps 1 and 2 always work with forgery i fixed, and in the one argument where both systems appear together (Step 3), each system's h-slot is evaluated at its own forgery's hash value—which receives its own name there—before the systems ever interact.

The left obstruction system P_i^L. We now read off what deadness of the two left-OR rows forces. The t-clause verification equation $G^{z_t} = A_t \cdot \mathsf{pk}_S^d$, after taking dlog_G and substituting the reduced representation of A_t, is the scalar identity

$$([A_t]_1 - z_t) + ([A_t]_S + d)\,\mathsf{sk}_S + [A_t]_R\,\mathsf{sk}_R + [A_t]_J\,\mathsf{sk}_J = 0,$$

whose coefficient vector against $(1, \mathsf{sk}_S, \mathsf{sk}_R, \mathsf{sk}_J)$ is forgery i's t-row of E_{hc} (compare its first four entries). If the row is dead its three key coordinates vanish, and the identity then forces the constant coordinate to vanish as well; so deadness of the t-row is

$$[A_t]_1 = z_t, \qquad [A_t]_S = -d, \qquad [A_t]_R = 0, \qquad [A_t]_J = 0.$$

Likewise the u-clause equation $G^{z_u} = A_u \cdot J^{\,h-d}$ (recall $c_u^{(i)} = c^{(i)} - c_t^{(i)} = h - d$) is forgery i's u-row; here the statement element J is adversarial, so its whole coordinate vector enters scaled by $h - d$, and deadness of the u-row is

$$\begin{aligned} [A_u]_1 + (h-d)[J]_1 &= z_u, & [A_u]_R + (h-d)[J]_R &= 0, \\ [A_u]_S + (h-d)[J]_S &= 0, & [A_u]_J + (h-d)[J]_J &= 0. \end{aligned}$$

The constant coordinates only pin the responses, $z_t = [A_t]_1$ and $z_u = [A_u]_1 + (h-d)[J]_1$, which we use only in the shared-query argument below; the t-row's key coordinates give the side conditions $[A_t]_R = [A_t]_J = 0$ and *spend* the split, $d = -[A_t]_S = -\kappa_{s,A_t} + \lambda\tau_{A_t}$ (a committed-coefficient function of λ).

Substituting this d into the three key coordinates of the u-row gives the *left obstruction system* P_i^L: a triple of polynomials in $\mathbb{Z}_p[\lambda, h]$, indexed by (S, R, J). Abbreviating the multiplier $\mu := h - d = h + [A_t]_S = h + \kappa_{s,A_t} - \lambda\tau_{A_t}$, its components are

$$P_i^L(\lambda, h) = \big(\ [A_u]_S + \mu[J]_S,\ \ [A_u]_R + \mu[J]_R,\ \ [A_u]_J + \mu[J]_J\ \big).$$

Deadness of the left OR is the simultaneous vanishing of all three components, which we abbreviate $P_i^L(\lambda, h) = 0$, together with the side conditions $[A_t]_R = [A_t]_J = 0$. Writing $[X]_S$ out as $\kappa_{s,X} - \lambda\tau_X$, the three components read

$$\begin{aligned} (\kappa_{s,A_u} - \lambda\tau_{A_u}) + \mu\,(\kappa_{s,J} - \lambda\tau_J) &= 0, \\ \kappa_{r,A_u} + \mu\,\kappa_{r,J} &= 0, \\ \kappa_{j,A_u} + \mu\,\kappa_{j,J} &= 0. \end{aligned}$$

Each component has degree at most two in λ and is affine in h; the h-coefficient of the system is the vector $([J]_S, [J]_R, [J]_J)$.

The right obstruction system P_i^R. The right clause is handled by the same elimination principle as the left, but the machinery must get heavier, and it is worth seeing why before wading in. First write $e := c_v^{(i)}$. In the left clause the split d enters verification through the statement element pk_S: a game-generated element whose reduced coordinates are $[\mathsf{pk}_S]_c = \mathbf{1}[c = S]$ (the S-unit vector). Consequently d appears in exactly one scalar equation—the S-coordinate of the t-row—with coefficient identically 1. That equation always determines d, substitution eliminates it, and "every coefficient of d vanishes" cannot occur. In the right clause the split e instead enters through the statement elements E_J, J, and R, which are adversarial elements with

arbitrary committed representations. Deadness of the three right-OR rows therefore yields a list of scalar equations $u_a + e\,v_a = 0$ in which *every* coefficient v_a is an adversarially chosen committed scalar, any of which may be zero—and *which* of them are zero is under the adversary's control, so no single equation can be designated in advance to solve for e. The symmetric way to demand a common solution without privileging any equation is to require all pairwise minors (equivalently, resultants) to vanish, and the case in which every v_a vanishes—so that e constrains nothing—must then be split off and handled separately. (In this language the left clause is simply the degenerate-easy case: writing its dead-row equations as $u_a + d\,v_a = 0$, the t-row's S-coordinate has $v \equiv 1$, the minors of the u-row equations against that row are exactly the components of P_i^L, and structural degeneracy is impossible.) Note also that a single row contributes several constraints on e—one per coordinate—since the bad event we care about is that the χs of *all three* keys vanish.

Each row equation below is derived exactly as for the left OR: take dlog_G of the verification equation, substitute the reduced representations, and observe that deadness kills the key coordinates while the verification identity then kills the remaining one; the coordinates of each can be checked directly against the v_1-, v_2-, and w-rows of the expanded matrix. The only wrinkle is the second Chaum–Pedersen row, whose verification equation has base pk_J rather than G: there the response z_v lands in the J coordinate instead of the constant coordinate—see the $-z_v^{(i)}$ in the fourth column of the expanded v_2-row—and since z_v was already spent absorbing the first row's constant coordinate, it absorbs nothing here.

The three right-OR rows are the v_1-, v_2-, and w-rows of E_{hc}; we read their constraints off as before, now eliminating the two responses z_v, z_w. Each surviving constraint has the form $u_a(\lambda, h) + e\,v_a(\lambda) = 0$, and we collect the ten pairs (u_a, v_a) as follows.

First Chaum–Pedersen row ($G^{z_v} = A_{v,1}E_J^e$). Its constant coordinate pins $z_v = [A_{v,1}]_1 + e[E_J]_1$, and its three key coordinates give $[A_{v,1}]_k + e[E_J]_k = 0$ for $k \in \{S, R, J\}$, contributing the pairs $\big([A_{v,1}]_k, [E_J]_k\big)$ $(a = 1, 2, 3)$.

Second Chaum–Pedersen row ($\mathsf{pk}_J^{z_v} = A_{v,2}J^e$). Because the base is pk_J, the response z_v sits in the J coordinate; the constant, S, and R coordinates carry no z_v and must vanish outright,

$$[A_{v,2}]_1 + e[J]_1 = 0, \qquad [A_{v,2}]_S + e[J]_S = 0, \qquad [A_{v,2}]_R + e[J]_R = 0,$$

contributing $\big([A_{v,2}]_1, [J]_1\big)$ $(a = 4)$ and $\big([A_{v,2}]_k, [J]_k\big)$ for $k \in \{S, R\}$ $(a = 5, 6)$. For the J coordinate we substitute the z_v pinned by the first row: $[A_{v,2}]_J + e[J]_J = z_v$ becomes

$$\big([A_{v,2}]_J - [A_{v,1}]_1\big) + e\big([J]_J - [E_J]_1\big) = 0,$$

the mixed pair $\big([A_{v,2}]_J - [A_{v,1}]_1,\ [J]_J - [E_J]_1\big)$ $(a = 7)$.

The w row ($G^{z_w} = A_w R^{h-e}$). Eliminating z_w from the constant coordinate, the three key coordinates give $[A_w]_k + (h - e)[R]_k = 0$, i.e.

$$\big([A_w]_k + h[R]_k\big) + e\big(-[R]_k\big) = 0 \qquad (k \in \{S, R, J\}),$$

the pairs $\big([A_w]_k + h[R]_k,\ -[R]_k\big)$ for $k \in \{S, R, J\}$ $(a = 8, 9, 10)$.

Collect the pairs above into a list $\mathcal{P}_R = \big((u_a(\lambda, h),\ v_a(\lambda))\big)_{a=1}^{10}$, where the subscript a indexes position in the list, in order of derivation: $a = 1, 2, 3$ are the first C-P row's pairs $(k = S, R, J)$, $a = 4, 5, 6$ are the second C-P row's constant, S, and R pairs, $a = 7$ is the mixed J-coordinate pair, and $a = 8, 9, 10$ are the w-row pairs $(k = S, R, J)$. Note that the hash h occurs only in u_8, u_9, u_{10}. Each pair encodes the requirement $u_a + e\,v_a = 0$. The list is nothing but forgery i's three right-OR rows of E_{hc} read as functions of the split: regarding the realized split $c_v^{(i)}$ as the formal variable e (so $c_w^{(i)} = h - e$), the v_1-, v_2-, and w-rows are each *affine in* e, of the form $U + eV$ with U, V free of e, and (u_a, v_a) is the (U, V) split of one scalar coordinate of one of these rows. Nine of the ten pairs come entrywise, from the coordinates free of the responses—the S, R, J coordinates of the v_1- and w-rows and the constant, S, R coordinates of the v_2-row—while the two response-bearing constant coordinates instead pin z_v and z_w; the tenth pair $(a = 7)$ is the v_2-row's J coordinate once z_v is substituted from the v_1-row. Stacking the ten pairs as the rows of a 10×2 matrix M over $\mathbb{Z}_p$, the requirements together assert $M \cdot (1, e)^\top = 0$. Since $(1, e)$ is nonzero, a common split e can satisfy all right-OR key-coefficient equations only if M has rank at most one—equivalently, only if all 2×2 minors of M vanish:

$$P_{a,b}^R(\lambda, h) := u_a(\lambda, h)v_b(\lambda) - u_b(\lambda, h)v_a(\lambda) = 0$$

for all pairs a, b. (Equivalently, $P^R_{a,b}$ is, up to sign, the resultant $\mathrm{Res}_e(u_a + e\, v_a,\ u_b + e\, v_b)$ of two polynomials affine in e: the minors eliminate the adversarially chosen split e, leaving conditions only on quantities committed before the fresh hash is drawn.) If some $v_a(\lambda) \neq 0$, these minors are also sufficient, since they force the common value

$$e = -\frac{u_a(\lambda, h)}{v_a(\lambda)}.$$

If all $v_a(\lambda)$ vanish, the split e has no effect; this is the structural-degeneracy case, and consistency requires $u_a(\lambda, h) = 0$ for every a.

For symmetry with the left clause we give the result a name: forgery i's *right obstruction system* is the family

$$P^R_i := \left(P^R_{a,b}\right)_{1 \le a < b \le 10}$$

of the $\binom{10}{2} = 45$ minors of $\mathcal{P}_R$ (we suppress the forgery index on the individual minors); in the structural-degeneracy case the operative constraints are instead the u_a themselves, as just noted. Like the components of P^L_i, each member of P^R_i has degree at most two in λ and is affine in h.

Step 2: obstruction polynomials keep their h-dependence. This completes Step 1: every deadness constraint is now an obstruction polynomial in (λ, h) with committed coefficients—for the left clause P^L_i (plus the side conditions), for the right clause the minors $P^R_{a,b}$ (or, in the structural-degeneracy case, the u_a themselves). Before starting Step 2, recall why it is needed. Step 3 will kill a candidate value λ_0 of c_1 by evaluating the later forgery's obstructions at λ_0 and using the freshness of that forgery's hash: an obstruction affine in h with nonzero h-coefficient vanishes at a fresh h with probability $1/p$. That argument is vacuous for an obstruction whose h-coefficient is *zero* at λ_0; if every obstruction of the forgery lost its h-dependence at λ_0, deadness there would impose no condition on the hash whatsoever, and no probability could be given. (Step 3 has a second potential failure mode—the earlier forgery's system could be identically zero as a polynomial in λ, yielding no candidates—but that one is handled probabilistically, in the proof of Claim 2, and needs no algebra here.) Step 2 therefore isolates exactly when a forgery's obstruction system can lose its h-dependence at the realized challenge, and shows each such degeneracy either yields a live receiver row or violates a verification side condition.

For the left OR, recall the coefficient of h in $P^L_i(\lambda, h)$ is $[J]_\bullet$—thus, P^L_i is independent of h exactly when $[J]_\bullet = 0$. Call this the left-degenerate case. For the right OR, h appears only in the pairs $a = 8, 9, 10$, where (with $k = S, R, J$ respectively)

$$u_a = [A_w]_k + h[R]_k, \qquad v_a = -[R]_k.$$

For any $a' \le 7$, the minor $P^R_{a',a}$ against $a \in \{8, 9, 10\}$ has h-coefficient $-v_{a'}[R]_k$; a minor between two of $a = 8, 9, 10$ has no h-dependence, its h-terms cancelling. Hence all right-OR minors lose h-dependence only if either $[R]_\bullet = 0$, or the coefficients $v_1, \ldots, v_7$ all vanish.

First suppose $[R]_\bullet = 0$. If the receiver row is not live, its induced polynomial in $Y = \mathsf{sk}_R$ is identically zero, i.e. all three of its coefficients vanish. Substituting $[R]_S = [R]_R = [R]_J = 0$ into those coefficients, this forces

$$[R]_1 = 0, \qquad [E_R]_1 + [E_R]_S \mathsf{sk}_S + [E_R]_J \mathsf{sk}_J = 0, \qquad [E_R]_R = 0.$$

We claim these equations force $E_R = 1$. Recall that E_R's reduced representation gives

$$\mathsf{dlog}_G E_R = [E_R]_1 + [E_R]_S\, \mathsf{sk}_S + [E_R]_R\, \mathsf{sk}_R + [E_R]_J\, \mathsf{sk}_J.$$

The second displayed equation says that all of the right-hand side except the sk_R-term sums to zero, and the third kills the sk_R-term itself; hence $\mathsf{dlog}_G E_R = 0$ and $E_R = G^0 = 1$, contradicting the verification side condition $E_R \neq 1$. (The first equation plays the same role for R: together with $[R]_\bullet = 0$ it gives $\mathsf{dlog}_G R = 0$, i.e. $R = 1$—consistent, since verification requires $R = E_R^{\mathsf{sk}_R}$.) Therefore the case $[R]_\bullet = 0$ either gives receiver-key extraction or violates verification.

It remains to consider the case $[R]_\bullet \neq 0$ and $v_1 = \cdots = v_7 = 0$. Writing these seven coefficients out gives $[E_J]_\bullet = 0$ $(a = 1, 2, 3)$, $[J]_1 = 0$ $(a = 4)$, $[J]_S = [J]_R = 0$ $(a = 5, 6)$, and $[E_J]_1 = ([J]_\bullet)_J$ $(a = 7)$.

Since $[E_J]_\bullet = 0$, the actual discrete logarithm of E_J is $[E_J]_1$. The verification side condition $E_J \neq 1$ therefore implies $[E_J]_1 \neq 0$. Consequently $([J]_\bullet)_J = [E_J]_1 \neq 0$, $[J]_\bullet \neq 0$.

Thus the left obstruction system P_i^L has a component with nonzero h-coefficient (equivalently, its h-coefficient vector $[J]_\bullet$ is nonzero). In other words, right-OR structural degeneracy forces the left OR to be nondegenerate, allowing extraction, unless $E_J = 1$. This completes Step 2—note how the two clauses interlock in this final case: the degeneracy of the right clause is discharged by showing it makes the *left* clause h-live. Claim 1 below records the outcome of Steps 1 and 2 in the exact form Step 3 consumes.

Completing the hard case. We now carry out Step 3, assembling the case analysis into a probability bound. The situation is this: for each forgery, killing every row forces a system of polynomial relations between the shared session challenge c_1—which $\mathcal{A}$ chooses *after* seeing both hash values—and that forgery's fresh hash value, and the degeneracy analysis guarantees that each forgery's system depends non-trivially on its own hash. The first-queried forgery's system therefore pins c_1 to at most two candidate values that are already determined when the second special query is made; the second forgery's system then demands that its fresh hash hit one predetermined value per candidate. We first make a bookkeeping observation, then give the claims.

Coordinates are fixed at the special query. The argument just sketched treats each dead-row condition as a polynomial identity in λ, and it needs the coefficients determined by the time the special query is made—before the fresh hash is returned—and, for root-counting to be meaningful, it needs the polynomial evaluated at a hypothetical λ_0 to describe the transcript in which $\mathcal{A}$ actually closes the session with challenge λ_0. Both hold for the $[X]_c$: as defined above they are polynomials in λ whose coefficients come from X's representation together with t_1 and sk_S (the latter enters $[X]_1$ via $s_1 = t_1 + \lambda \mathsf{sk}_S$), all fixed before either special query is made in the hard case; $[X]_1$ and $[X]_S$ are affine in λ while $[X]_R, [X]_J$ are constants. Hence the degree counts above hold (each component of P_i^L and each minor $P_{a,b}^R$ has degree at most two in λ and is affine in h), and both probabilistic steps are licensed: counting the roots in λ of a polynomial pinned down by the first special query, and treating the second hash value as fresh randomness against coefficients pinned down by the second. (One might object that $[X]_1$ involves t_1 and sk_S—quantities no extractor is given, and that for the OMDL reduction of the full argument are exactly the discrete logs it must compute. This is harmless: bounding the bad event is a statistical statement about the game itself, so the analysis may use t_1 and sk_S freely. An extractor never evaluates $[X]_1$ at a hypothetical λ; it uses only the realized value $[X]_1|_{\lambda=c_1} = \gamma_X + s_1\tau_X$, computable from the observed response s_1.)

For forgery i define its *witness list* $\mathcal{W}_i$: the three components of $P_i^L(\lambda, h)$, followed by all pairwise minors $P_{a,b}^R(\lambda, h)$, in some fixed order. Each element of $\mathcal{W}_i$ is a polynomial of degree at most two in λ and affine in h, with coefficients fixed at forgery i's special query; write a generic element as $Q(\lambda, h) = U(\lambda) + h\,V(\lambda)$.

Claim 1. *Fix an accepting hard-case transcript in which no Schnorr-like row and no receiver row is live. Then for each forgery $i \in \{1, 2\}$: (i) every element of $\mathcal{W}_i$ vanishes at $(\lambda, h) = (c_1, c^{(i)})$; and (ii) some element of $\mathcal{W}_i$ has nonzero h-coefficient at $\lambda = c_1$, i.e. $V(c_1) \neq 0$.*

Proof. (i) Deadness of the two left-OR rows gives, after eliminating d as above, $P_i^L(c_1, c^{(i)}) = 0$. Deadness of the three right-OR rows says exactly that $u_a(c_1, c^{(i)}) + e\,v_a(c_1) = 0$ for every pair (u_a, v_a) in the list, with the common split $e = c_v^{(i)}$; hence every minor vanishes there, since $u_a v_b - u_b v_a = (-ev_a)v_b - (-ev_b)v_a = 0$.

(ii) All coordinates here are evaluated at $\lambda = c_1$. If $[J]_\bullet \neq 0$, some component of P_i^L has nonzero h-coefficient $[J]_k$ and we are done. So suppose $[J]_\bullet = 0$. Then $[R]_\bullet = 0$ is impossible: as shown above, it makes the receiver row live or forces $E_R = 1$. Likewise "$v_a = 0$ for every $a \leq 7$" is impossible: as shown above, it forces $[J]_J = [E_J]_1 \neq 0$, contradicting $[J]_\bullet = 0$. Hence there is an index $a' \leq 7$ with $v_{a'} \neq 0$ and a coordinate k with $[R]_k \neq 0$, and the minor $P_{a',a}^R$ against the pair $a \in \{8, 9, 10\}$ corresponding to k has h-coefficient $-v_{a'}\,[R]_k \neq 0$. □

Claim 2. *For any hard-case $\mathcal{A}$, the event* BAD *that* $\mathbf{G}_0$ *accepts and no row is live satisfies* $\Pr[\mathsf{BAD}] \leq 3/p$.

Proof. The adversary's two random-oracle queries are its two special queries; let α denote the earlier and β the later. Let forgery a be the one whose special query is α and b the other, and write $h_\alpha := c^{(a)}$, $h_\beta := c^{(b)}$. All coefficients of the elements of $\mathcal{W}_a$ and $\mathcal{W}_b$ are fixed before h_β is drawn, and those of $\mathcal{W}_a$ before h_α is drawn.

Let Z be the event that every element of $\mathcal{W}_a$, with h_α substituted for h, is the zero polynomial of $\mathbb{Z}_p[\lambda]$. We bound $\mathsf{BAD} \wedge Z$ and $\mathsf{BAD} \wedge \neg Z$ separately.

$\underline{\mathcal{B}^{\mathtt{chal},\mathtt{dlog}}(G)}$:

$\mathsf{pk}_R, \mathsf{sk}_R, \mathsf{pk}_J, \mathsf{sk}_J \leftarrow\!\!\$\ \mathsf{KGen}$; $\zeta \leftarrow 0$; $b \leftarrow 1$; $X \leftarrow \mathtt{chal}$
$t_{\ell+1} \leftarrow\!\!\$\ \mathbb{Z}_p$; $T_{\ell+1} \leftarrow G^{t_{\ell+1}}$
$(m_1, \sigma_1), \ldots, (m_{\ell+1}, \sigma_{\ell+1}) \leftarrow\!\!\$\ \mathcal{A}^{\tilde{\mathsf{S}}_1, \tilde{\mathsf{S}}_2}(X, \mathsf{pk}_R, \mathsf{pk}_J)$
$\forall i : (L^{(i)}, (c_t^{(i)}, z_t^{(i)}, z_u^{(i)}), (c_v^{(i)}, z_v^{(i)}, z_w^{(i)})) \leftarrow \sigma_i$
$(J^{(i)}, R^{(i)}, E_J^{(i)}, E_R^{(i)}, A_t^{(i)}, A_u^{(i)}, A_{v,1}^{(i)}, A_{v,2}^{(i)}, A_w^{(i)}) \leftarrow L^{(i)}$
If $\exists i \neq j$ such that $(m_i, \sigma_i) = (m_j, \sigma_j)$: Return 0
For $i = 1$ to $\ell + 1$:
  $c^{(i)} \leftarrow H(X, \mathsf{pk}_R, \mathsf{pk}_J, L^{(i)}, m_i)$
  $c_u^{(i)} \leftarrow c^{(i)} - c_t^{(i)}$
  $c_w^{(i)} \leftarrow c^{(i)} - c_v^{(i)}$
  $b \xleftarrow{\wedge} R^{(i)} = \left(E_R^{(i)}\right)^{\mathsf{sk}_R} \wedge E_R^{(i)} \neq 1$
  $b \xleftarrow{\wedge} G^{z_t^{(i)}} = A_t^{(i)} \cdot X^{c_t^{(i)}}$
  $b \xleftarrow{\wedge} G^{z_u^{(i)}} = A_u^{(i)} \cdot \left(J^{(i)}\right)^{c_u^{(i)}}$
  $b \xleftarrow{\wedge} G^{z_v^{(i)}} = A_{v,1}^{(i)} \cdot \left(E_J^{(i)}\right)^{c_v^{(i)}} \wedge E_J^{(i)} \neq 1$
  $b \xleftarrow{\wedge} \mathsf{pk}_J^{z_v^{(i)}} = A_{v,2}^{(i)} \cdot \left(J^{(i)}\right)^{c_v^{(i)}}$
  $b \xleftarrow{\wedge} G^{z_w^{(i)}} = A_w^{(i)} \cdot \left(R^{(i)}\right)^{c_w^{(i)}}$
If $b = 0$: Return $\perp$
Find a forgery i and a Schnorr-like check whose row
  $(-L, \chi_s, \chi_r, \chi_j)$ has $\chi_s \neq 0$; if none: Return $\perp$
$\mathsf{sk}_S \leftarrow (L - \chi_r \mathsf{sk}_R - \chi_j \mathsf{sk}_J)/\chi_s$
For $j = 1$ to ℓ: $t_j \leftarrow s_j - c_j \mathsf{sk}_S$
Return $(X, \mathsf{sk}_S), (T_1, t_1), \ldots, (T_\ell, t_\ell)$

$\underline{\tilde{\mathsf{S}}_1}$:

If $\mathsf{sess} = \mathsf{open}$: Return $\perp$
$\mathsf{sess} \leftarrow \mathsf{open}$; $\zeta \leftarrow \zeta + 1$
If $\zeta < \ell + 1$:
$T_\zeta \leftarrow \mathtt{chal}$
Else:
  $T_\zeta \leftarrow T_{\ell+1}$
Return T_ζ

$\underline{\tilde{\mathsf{S}}_2(c)}$:

If $\mathsf{sess} \neq \mathsf{open}$: Return $\perp$
$c_\zeta \leftarrow c$; $s_\zeta \leftarrow \mathtt{dlog}(T_\zeta X^c)$
$\mathsf{sess} \leftarrow \mathsf{closed}$
Return s_ζ

$\underline{H(A, B, C, \vec{L}, m)}$:

if $\mathcal{T}[A, B, C, \vec{L}, m] = \perp$:
  $x \leftarrow\!\!\$\ \mathbb{Z}_p$
  $\mathcal{T}[A, B, C, \vec{L}, m] \leftarrow x$
Return $\mathcal{T}[A, B, C, \vec{L}, m]$

Figure 11: Reduction $\mathcal{B}$ from OMUNF of BAMF to the one-more discrete logarithm problem in $\mathbb{G}$. The row coefficients $(-L, \chi_s, \chi_r, \chi_j)$ of a Schnorr-like check are computed from $\mathcal{A}$'s AGM representations together with the recorded (c_j, s_j) and $t_{\ell+1}$, as defined in the proof of Theorem 2. $\mathcal{B}$'s OMDL challenges are $X, T_1, \ldots, T_\ell$, and it answers all of them.

In BAD, Claim 1(ii) applied to forgery a gives an element of $\mathcal{W}_a$ with $V(c_1) \neq 0$, so in particular $V \neq 0$ as a polynomial; hence the *first* element $Q^\dagger = U^\dagger + hV^\dagger$ of $\mathcal{W}_a$ with $V^\dagger \neq 0$ exists, and its position depends only on data fixed before h_α is drawn. Under Z we have $U^\dagger + h_\alpha V^\dagger \equiv 0$; since $V^\dagger \neq 0$, at most one value of h_α satisfies this, and that value is determined before h_α is drawn. Hence $\Pr[\mathsf{BAD} \wedge Z] \leq 1/p$.

Now condition on the state of the game just before h_β is drawn, and suppose $\neg Z$. Let Q^* be the first element of $\mathcal{W}_a(\cdot, h_\alpha)$ that is nonzero in $\mathbb{Z}_p[\lambda]$; it has degree at most two, hence at most two roots, and both Q^* and its roots are determined by the conditioning. By Claim 1(i) applied to forgery a, in BAD the challenge c_1 is a root of Q^*. Fix a candidate root λ_0. If $c_1 = \lambda_0$ then, by Claim 1 applied to forgery b, the first element $U' + hV'$ of $\mathcal{W}_b$ with $V'(\lambda_0) \neq 0$ exists and vanishes at (λ_0, h_β), so $h_\beta = -U'(\lambda_0)/V'(\lambda_0)$, a single value determined by the conditioning. Since h_β is uniform and independent of the conditioning, each of the at most two candidates contributes at most $1/p$, so $\Pr[\mathsf{BAD} \wedge \neg Z] \leq 2/p$. □

Combining Claim 2 with the extraction trichotomy summarizes the hard case: on all but a $3/p$-fraction of winning hard-case executions, some row is live, and a party holding the appropriate two secret keys extracts the third. What remains is to exhibit reductions that actually hold two of the keys, can nevertheless simulate the dvOMUNF game, and profit from the extracted key; we do this for arbitrary ℓ in the full argument below.

Distinctness of the special queries. The final hard-case restriction excludes nothing: suppose two distinct forgeries were to share their special query, i.e. $L^{(1)} = L^{(2)}$, $m_1 = m_2$, and hence $c^{(1)} = c^{(2)} =: h$. We claim that on an accepting transcript with no live row the two signatures then coincide, so $\mathbf{G}_0$'s distinctness

$\underline{\mathcal{C}(X)}$:

$\mathsf{pk}_S, \mathsf{sk}_S, \mathsf{pk}_R, \mathsf{sk}_R \leftarrow\!\!\$\ \mathsf{KGen}\ ;\ \zeta \leftarrow 0\ ;\ b \leftarrow 1$
$(m_1, \sigma_1), \ldots, (m_{\ell+1}, \sigma_{\ell+1}) \leftarrow\!\!\$\ \mathcal{A}^{\mathsf{S}_1, \mathsf{S}_2}(\mathsf{pk}_S, \mathsf{pk}_R, X)$
$\forall i : (L^{(i)}, (c_t^{(i)}, z_t^{(i)}, z_u^{(i)}), (c_v^{(i)}, z_v^{(i)}, z_w^{(i)})) \leftarrow \sigma_i$
$(J^{(i)}, R^{(i)}, E_J^{(i)}, E_R^{(i)}, A_t^{(i)}, A_u^{(i)}, A_{v,1}^{(i)}, A_{v,2}^{(i)}, A_w^{(i)}) \leftarrow L^{(i)}$
If $\exists i \neq j$ such that $(m_i, \sigma_i) = (m_j, \sigma_j)$: Return 0
For $i = 1$ to $\ell + 1$:
  $c^{(i)} \leftarrow H(\mathsf{pk}_S, \mathsf{pk}_R, X, L^{(i)}, m_i)$
  $c_u^{(i)} \leftarrow c^{(i)} - c_t^{(i)}$
  $c_w^{(i)} \leftarrow c^{(i)} - c_v^{(i)}$
  $b \xleftarrow{\wedge} R^{(i)} = \left(E_R^{(i)}\right)^{\mathsf{sk}_R} \wedge E_R^{(i)} \neq 1$
  $b \xleftarrow{\wedge} G^{z_t^{(i)}} = A_t^{(i)} \cdot \mathsf{pk}_S^{c_t^{(i)}}$
  $b \xleftarrow{\wedge} G^{z_u^{(i)}} = A_u^{(i)} \cdot \left(J^{(i)}\right)^{c_u^{(i)}}$
  $b \xleftarrow{\wedge} G^{z_v^{(i)}} = A_{v,1}^{(i)} \cdot \left(E_J^{(i)}\right)^{c_v^{(i)}} \wedge E_J^{(i)} \neq 1$
  $b \xleftarrow{\wedge} X^{z_v^{(i)}} = A_{v,2}^{(i)} \cdot \left(J^{(i)}\right)^{c_v^{(i)}}$
  $b \xleftarrow{\wedge} G^{z_w^{(i)}} = A_w^{(i)} \cdot \left(R^{(i)}\right)^{c_w^{(i)}}$
If $b = 0$: Return $\perp$
Find a forgery i and a Schnorr-like check whose row
  $(-L, \chi_s, \chi_r, \chi_j)$ has $\chi_j \neq 0$; if none: Return $\perp$
Return $(L - \chi_s \mathsf{sk}_S - \chi_r \mathsf{sk}_R)/\chi_j$

$\underline{H(A, B, C, \vec{L}, m)}$:

if $\mathcal{T}[A, B, C, \vec{L}, m] = \perp$:
  $x \leftarrow\!\!\$\ \mathbb{Z}_p$
  $\mathcal{T}[A, B, C, \vec{L}, m] \leftarrow x$
Return $\mathcal{T}[A, B, C, \vec{L}, m]$

Figure 12: Reduction $\mathcal{C}$ from OMUNF of BAMF to the discrete logarithm problem in $\mathbb{G}$. Note that since $\mathcal{C}$ knows sk_S, it uses the real code of the OMUNF game oracles; we omit these for brevity. Row coefficients are computed as in Figure 11; because $\mathcal{C}$ knows both sk_S and sk_R, any row with $\chi_j \neq 0$ yields $\mathsf{sk}_J = \mathsf{dlog}_G X$.

check fails and $\mathcal{A}$ does not win. Indeed (all coordinates at $\lambda = c_1$), deadness of the t-rows pins $c_t^{(i)} = -[A_t]_S$ for both i, and the first coordinates then pin $z_t^{(i)} = [A_t]_1$ and $z_u^{(i)} = [A_u]_1 + (h - c_t^{(i)})[J]_1$. On the right, $[R]_\bullet \neq 0$ (else the receiver row is live or $E_R = 1$); choosing k with $[R]_k \neq 0$, deadness of the kth coordinate of the w-row pins $h - c_v^{(i)} = -[A_w]_k/[R]_k$, hence $c_v^{(i)}$, and the first coordinates pin $z_v^{(i)}$ and $z_w^{(i)}$. All six scalars agree across the two forgeries, so $(m_1, \sigma_1) = (m_2, \sigma_2)$. Note that this argument uses only the realized values of the reduced coefficients—the polynomial structure in λ plays no role—so it applies wherever the shared query is made; we reuse it below.

Finishing the proof. We now lift the hard-case analysis to an arbitrary algebraic $\mathcal{A}$ that makes ℓ signing queries and q_h random-oracle queries. Recall the normalizations from the beginning of the proof: $\mathcal{A}$ opens $\ell + 1$ sessions and closes the first ℓ—write c_j, s_j for the challenge and response of session j—and, by the theorem's hypothesis, it queries H on every forgery it outputs. Representations now range over the basis $(G, \mathsf{pk}_S, \mathsf{pk}_R, \mathsf{pk}_J, T_1, \ldots, T_{\ell+1})$, say $X = G^{\gamma_X} \mathsf{pk}_S^{\kappa_{s,X}} \mathsf{pk}_R^{\kappa_{r,X}} \mathsf{pk}_J^{\kappa_{j,X}} \prod_j T_j^{\rho_{j,X}}$, and the reduced coordinates $[X]_c$ generalize to

$$[X]_1 := \gamma_X + \sum_{j=1}^{\ell} \rho_{j,X}\, s_j + \rho_{\ell+1,X}\, t_{\ell+1}, \qquad [X]_S := \kappa_{s,X} - \sum_{j=1}^{\ell} \rho_{j,X}\, c_j\,,$$

with $[X]_R = \kappa_{r,X}$, $[X]_J = \kappa_{j,X}$, folding in the ℓ session identities $T_j = G^{s_j} \mathsf{pk}_S^{-c_j}$ and the game-known $t_{\ell+1}$; as before, $\mathsf{dlog}_G X = [X]_1 + [X]_S \mathsf{sk}_S + [X]_R \mathsf{sk}_R + [X]_J \mathsf{sk}_J$. The verification rows—now $6(\ell+1)$ of them—and the live/dead trichotomy carry over unchanged, and the extractions of the hard case are realized by three concrete reductions $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$, defined in Figures 11–13. $\mathcal{B}$ embeds its OMDL challenges as pk_S and the commitments $T_1, \ldots, T_\ell$, simulates the jth S_2 query by spending its jth $\mathtt{dlog}$ query, extracts sk_S from any live Schnorr-like row, and then solves *all* $\ell + 1$ of its OMDL challenges via $t_j = s_j - c_j \mathsf{sk}_S$. $\mathcal{C}$ (resp. $\mathcal{D}$)

$\underline{\mathcal{D}(X)}$:

$\mathsf{pk}_S, \mathsf{sk}_S, \mathsf{pk}_J, \mathsf{sk}_J \leftarrow\!\$\ \mathsf{KGen}\ ;\ \zeta \leftarrow 0\ ;\ b \leftarrow 1$
$(m_1, \sigma_1), \ldots, (m_{\ell+1}, \sigma_{\ell+1}) \leftarrow\!\$\ \mathcal{A}^{\mathsf{S}_1, \mathsf{S}_2}(\mathsf{pk}_S, X, \mathsf{pk}_J)$
$\forall i : (L^{(i)}, (c_t^{(i)}, z_t^{(i)}, z_u^{(i)}), (c_v^{(i)}, z_v^{(i)}, z_w^{(i)})) \leftarrow \sigma_i$
$(J^{(i)}, R^{(i)}, E_J^{(i)}, E_R^{(i)}, A_t^{(i)}, A_u^{(i)}, A_{v,1}^{(i)}, A_{v,2}^{(i)}, A_w^{(i)}) \leftarrow L^{(i)}$
If $\exists i \neq j$ such that $(m_i, \sigma_i) = (m_j, \sigma_j)$: Return 0
For $i = 1$ to $\ell + 1$:
  $c^{(i)} \leftarrow H(\mathsf{pk}_S, X, \mathsf{pk}_J, L^{(i)}, m_i)$
  $c_u^{(i)} \leftarrow c^{(i)} - c_t^{(i)}$
  $c_w^{(i)} \leftarrow c^{(i)} - c_v^{(i)}$
  $b \xleftarrow{\wedge} E_R^{(i)} \neq 1$
  $b \xleftarrow{\wedge} G^{z_t^{(i)}} = A_t^{(i)} \cdot \mathsf{pk}_S^{c_t^{(i)}}$
  $b \xleftarrow{\wedge} G^{z_u^{(i)}} = A_u^{(i)} \cdot \left(J^{(i)}\right)^{c_u^{(i)}}$
  $b \xleftarrow{\wedge} G^{z_v^{(i)}} = A_{v,1}^{(i)} \cdot \left(E_J^{(i)}\right)^{c_v^{(i)}} \wedge E_J^{(i)} \neq 1$
  $b \xleftarrow{\wedge} \mathsf{pk}_J^{z_v^{(i)}} = A_{v,2}^{(i)} \cdot \left(J^{(i)}\right)^{c_v^{(i)}}$
  $b \xleftarrow{\wedge} G^{z_w^{(i)}} = A_w^{(i)} \cdot \left(R^{(i)}\right)^{c_w^{(i)}}$
If $b = 0$: Return $\perp$
For each forgery i and Schnorr-like check whose row
  $(-L, \chi_s, \chi_r, \chi_j)$ has $\chi_r \neq 0$:
  $y \leftarrow (L - \chi_s \mathsf{sk}_S - \chi_j \mathsf{sk}_J)/\chi_r$; If $G^y = X$: Return y
For $i = 1$ to $\ell + 1$ with $\Pi_{R^{(i)}, E_R^{(i)}} \not\equiv 0$:
  For each root $y \in \mathbb{Z}_p$ of $\Pi_{R^{(i)}, E_R^{(i)}}$:
    If $G^y = X$: Return y
Return $\perp$

$\underline{H(A, B, C, \vec{L}, m)}$:

if $\mathcal{T}[A, B, C, \vec{L}, m] = \perp$:
  $x \leftarrow\!\$\ \mathbb{Z}_p$
  $\mathcal{T}[A, B, C, \vec{L}, m] \leftarrow x$
Return $\mathcal{T}[A, B, C, \vec{L}, m]$

Figure 13: Reduction $\mathcal{D}$ from OMUNF of BAMF to the discrete logarithm problem in $\mathbb{G}$. Since $\mathcal{D}$ knows sk_S it runs the real OMUNF signing oracles, which we omit as in Figure 12. $\mathcal{D}$ cannot evaluate the receiver check $R = E_R^{\mathsf{sk}_R}$, lacking $\mathsf{sk}_R = \mathsf{dlog}_G X$; it attempts extraction unconditionally, from any row with $\chi_r \neq 0$ or from the roots of an induced receiver polynomial $\Pi_{R^{(i)}, E_R^{(i)}}$ (degree at most two, coefficients computable from $\mathsf{sk}_S, \mathsf{sk}_J$), and the test $G^y = X$ rules out false candidates.

embeds a discrete-log challenge as pk_J (resp. pk_R), generates the other two keys itself and runs the real signing oracles, and extracts its challenge from a live row as computed in the trichotomy—in $\mathcal{D}$'s case via the induced receiver polynomial when the live row is a receiver row. Each of the three simulates $\mathbf{G}_0$ perfectly. We may also assume the $\ell + 1$ special queries are pairwise distinct: the distinctness argument above used only the realized reduced coefficients, so two distinct forgeries that share a special query and produce no live row coincide, regardless of where the shared query falls.

It remains to bound the probability that $\mathcal{A}$ wins while no row is live. Following [54], classify each special query as *inside* if it is made while some eventually-closed session is open, and *outside* otherwise—before the first session opens, between sessions, or during the last, never-closed session.

Suppose some special query is outside. At the moment it is made, every coefficient of the corresponding forgery's witness list is a fixed scalar: sessions that have not yet opened contribute zero coefficients, closed sessions contribute only through the already-fixed (c_j, s_j), and $T_{\ell+1}$ only through the game-known $t_{\ell+1}$. Claim 1 holds verbatim with every polynomial so evaluated: on an accepting no-live-row transcript the witness list—now a list of affine functions of that forgery's hash alone—vanishes at the fresh hash value, while some entry has a nonzero h-coefficient, and the first such entry is determined before the hash is drawn. Each random-oracle query therefore contributes at most $1/p$, for at most q_h/p in total. This is the analogue of case C_1 of [54]; the only new ingredient is that the nonvanishing of the h-coefficient is supplied by Claim 1(ii)

rather than being automatic.

Otherwise, all $\ell+1$ special queries are inside. Since only ℓ sessions are eventually closed and the special queries are distinct, by pigeonhole two of them—say α and later β—are made during the same session j_0, exactly as in case C_2 of [54]. Every ingredient of the hard case reappears with $(T_{j_0}, t_{j_0}, c_{j_0}, s_{j_0})$ in the role of (T_1, t_1, c_1, s_1): when α and β are made, sessions after j_0 do not yet exist, so their coefficients vanish (just as the hard case forbade T_2), and sessions before j_0 are closed, so they enter only as fixed constants inside $[X]_1$ and $[X]_S$. Each dead-row condition for the two forgeries is therefore again a polynomial of degree at most two in $\lambda = c_{j_0}$, affine in the forgery's hash, with coefficients fixed at its special query. For each fixed pair (α, β) of random-oracle queries, the proof of Claim 2 applies unchanged to the event that $\mathcal{A}$ wins with no live row while α, β are special queries falling inside one session, bounding its probability by $3/p$. Which pair plays this role is not known in advance, so we finish with a union bound over the at most $\binom{q_h}{2}$ pairs.

Collecting the two cases,

$$\Pr\big[\mathbf{G}_0 \text{ accepts and no row is live}\big] \;\le\; \frac{3\binom{q_h}{2} + q_h}{p}\,,$$

and on the complementary event one of the three perfectly simulating reductions extracts. Bounding $3\binom{q_h}{2} \le 3q_h^2$ for a cleaner statement, we conclude

$$\mathsf{Adv}^{\mathsf{dvomunf}}_{\mathsf{BAMF},\mathcal{A}} \;\le\; \mathsf{Adv}^{\mathsf{omdl}}_{\mathcal{B},\mathbb{G}} + \mathsf{Adv}^{\mathsf{dl}}_{\mathcal{C},\mathbb{G}} + \mathsf{Adv}^{\mathsf{dl}}_{\mathcal{D},\mathbb{G}} + \frac{3q_h^2 + q_h}{p}\,.$$

□

Discussion: a more modular organization of the bad-event analysis. We record an alternative organization of the no-live-row analysis that we did not pursue, but that may be preferable for future proofs of this kind (e.g. for concurrent variants, or for other multi-key designated-verifier primitives). The bad event is naturally a conjunction $E_S \wedge E_J \wedge E_R$, where E_S is the event that every χ_s extraction coefficient in the explainability matrix vanishes, E_J is defined analogously for χ_j, and E_R is the event that every χ_r vanishes *and* both receiver rows induce the zero polynomial (so that reduction $\mathcal{D}$, whose live rows may be either Schnorr-like or receiver rows, fails). As noted in the proof overview, the *naive* modularization—a union bound over the three per-key events—is doomed, because any single E_k is cheap for the adversary to force; the modularity described here is of the algebra, with probability extracted only from the conjunction. Two observations make the structure exploitable. First, on the complement side the conjunction splits exactly into the extraction trichotomy:

$$\Pr[\text{win}] = \Pr[\text{win} \wedge \neg E_S] + \Pr[\text{win} \wedge E_S \wedge \neg E_R] + \Pr[\text{win} \wedge E_S \wedge E_R \wedge \neg E_J] + \Pr[\text{win} \wedge E_S \wedge E_J \wedge E_R].$$

To see the equality, write win $=$ (win $\wedge \neg E_S$) $\vee$ (win $\wedge E_S$) (an event and its negation), then split the second disjunct the same way on E_R, and that result again on E_J; expanding, the four events above are mutually exclusive (any two differ on one of E_S, E_R, E_J) and their union is win, so their probabilities add. The first three terms are precisely the success events of the three reductions—the trichotomy resolves a live transcript to the first extractable key in the order sender, receiver, judge, matching the order E_S, E_R, E_J in which we split. (This is finite additivity over the partition induced by E_S, E_R, E_J.) Second—and this is the part our clause-based presentation obscures—the three sub-events are algebraically almost independent: the χ_s coefficient of every row involves only sender coefficients $[\cdot]_S$ together with the shared post-hoc scalars (the splits, the challenge, and the hash); the χ_r coefficients involve only receiver coefficients; and the χ_j coefficients only judge coefficients plus the response z_v of the pk_J-base row. Thus E_S, E_R, E_J are three small polynomial systems coupled *only* through the shared post-hoc unknowns, and one could eliminate the post-hoc variables from each subsystem separately (yielding three independently verifiable families of per-key obstruction polynomials) before a short combining step over the shared variables—rather than deriving obstructions clause by clause as we did. Two cautions for anyone attempting this. There is no identity expressing $\Pr[E_S \wedge E_J \wedge E_R]$ as a sum of single-event probabilities conditioned on the others; sums arise only from partitions, so the probabilistic step should still proceed by partitioning on a transcript-computable scenario (as our Step 2 does), with the sibling events entering each case only as *deterministic algebraic*

hypotheses. In particular, one should not condition probabilistically on sibling bad events: they constrain the very hash values whose freshness the argument consumes, and conditioning on them can destroy the uniformity that the $1/p$ bounds require. Conditioning on the past (the state of the game before a hash value is drawn), as in the proof above, remains the safe discipline.

A corollary: unforgeability of the original AMF. Because our dvOMUNF game uses the strong (distinct message–signature pairs) win condition and BAMF's user side is randomized, sequential one-more unforgeability implies unforgeability of the induced non-interactive scheme in which the signer runs both sides of the protocol and outputs the result: each signing query is simulated by one complete session, and a fresh-message forgery supplies the $(\ell+1)$-st distinct pair. (The reduction loses at most $\binom{q_s}{2}/p$, since two honest signatures on the same message collide only with probability $1/p$; here q_s is the number of signing queries.) This is the honest-user-only case of honest-user unforgeability [72]. For BAMF, the induced scheme is the original AMF franking scheme of [85], since an honest execution outputs an identically distributed signature. Because our dvOMUNF game verifies forgeries with the *receiver* secret key, the notion obtained is a weakening of the *unforgeability* (UNF) notion of [85]. It is weaker because it does not give the adversary partial control over keys in the oracle calls. Theorem 2 therefore also yields an AGM+ROM analysis of this weakened UNF for the original AMF, under OMDL and DL—to our knowledge the first correct one: the verification of [85] accepts trivial forgeries (e.g. $E_J = 1$ with $c_v = z_v = 0$), which are excluded only by the checks $E_R \neq 1$ and $E_J \neq 1$ that our verification adds (Figure 4). It's very likely the original proofs from [85] of their stronger UNF, as well as r-bind and s-bind, can be repaired, but we leave this to future work.

Finally, we observe that much of the difficulty above is inherited from the CDS-style OR composition [30] used by AMF, whose adversarially chosen challenge splits are what give rise to the obstruction machinery and the shared-special-query subtlety. An AMF variant built instead on *sequential*-OR composition [15, 39], where each branch's challenge is a random-oracle output bound to the sibling branch's commitment, would eliminate the splits as free adversarial choices and should substantially simplify the unforgeability analysis—at the cost of departing from the deployed AMF signature format and re-deriving its deniability properties. We leave exploring this variant to future work.

Concurrent security. Our new definition, and this proof for BAMF, only capture the sequential setting. It is known that achieving dvOMUNF in the concurrent setting—where the adversary can open many signing sessions in parallel—is much more difficult in general. Indeed, our protocol is likely not concurrently secure as written because of the ROS attack [21] on the blind Schnorr protocol we use as a building block. We believe our sequential security result is useful both in its own right—in some settings it is possible to prevent concurrent attacks at the application level—and as a stepping stone to concurrent security. We suspect that applying known techniques [77] can render our protocol, or a small variant of it, concurrently secure, but we leave the details to future work.

10 Related Work

Telephone & call authentication. In [66], Prasad et al. conducted a large scale study of robocalls using honeypots and consumer complaints over 5 years. Their study discusses measurements of STIR/SHAKEN's effectiveness and adoption based on the calls they received. In [68], Reaves et al. propose AuthLoop, an alternative caller ID authentication system that runs a version of TLS over the voice channel resulting in a $\approx$ 9-second call-setup delay. Du et al. propose UCBlocker in [32], a privacy preserving caller ID authentication and spam prevention scheme based on issuing anonymous credentials to callers and allowing callees to define custom policies. In [17], Adei et al. propose Jäger, a scheme that automates the traceback process but does not authenticate Caller ID or block robocalls in real time. In their criticism of STIR/SHAKEN, they also identify an issue with "timing-out" `x5u` URLs preventing real-time verification and obstructing call traceback, however they do not explore attacks relating to expiring URLs or replaced certificates nor do they conduct measurements related to these issues. Additionally, we propose a solution to this problem compatible with STIR/SHAKEN as opposed to an entirely novel protocol. Concurrent with our work, Adei et al. [18] present an OOBSS solution that introduces additional infrastructure requirements. Our approach prioritizes broader compatibility with existing infrastructure and the currently deployed telephone ecosystem.

Blind signatures & AMFs. Abe [14] and Abe and Okamoto [16] propose concurrently secure partially blind signature schemes that involve OR proofs, but their work is not related to designated verifier signatures. Tessaro and Zhu [77], Fuchsbauer and Wolf [43], and Fuchsbauer et al. [42] advance the understanding of blind Schnorr and other discrete log based signing schemes, especially in the concurrent setting. Sahu et al. introduce post-quantum designated verifier blind signature schemes based on supersingular isogenies in [70]. In [24], Cao et al. propose a designated verifier proxy blind signature scheme from bilinear group-pairs. [55] by Klooß and Reichle introduce an efficient pairing-free blind signature scheme secure in the random oracle model under the standard DDH assumption. Hecate proposed by Issa et al. in [52] builds a new AMF scheme that allows anonymous message forwarding but requires out-of-band interaction with the moderator. We do not believe these features are necessary in our setting and the preprocessing step presents a challenge for use in the telephone ecosystem.

11 Conclusion and Takeaways

In this work we have shown that STIR/SHAKEN falls short of its security goals and creates new privacy harms. Because our findings implicate a uniquely broad set of stakeholders, we close with concrete recommendations for each.

Standards bodies. Privacy must be treated as a first-class design concern for standards. Deniability deserves the same treatment even when not stated as an explicit goal: STIR/SHAKEN is now the second deployed protocol, after DKIM, in which signatures originally intended as ephemeral abuse-reporting attestations have become long-lived publicly-verifiable evidence reusable in unrelated proceedings (§5).

The design processes around TLS 1.3, MLS, and Privacy Pass show that engaging cryptographers in standards work early leads to good outcomes.

Two specific items deserve early attention from the relevant standards bodies. First, the handset-side rendering of STIR/SHAKEN attestations is part of the threat model: when a "verified" indicator can be displayed next to a number that was not the one validated (§4), the cryptographic guarantee has been undone by the user-interface layer, and the standards should constrain which fields handsets may display alongside `verstat`. Second, proposals to extend STIR/SHAKEN to SMS deserve particular scrutiny: signing message content as well as metadata would make the deniability picture strictly worse (§5).

The STIR/SHAKEN governance regime. The STIR/SHAKEN PKI is operating well below current best practice for public-key infrastructures. We recommend that the STI-GA and STI-PA adopt substantive policies analogous to those developed by the CA/Browser Forum for the web PKI, including enforcement against CAs and telecoms that do not comply: our measurements (§7) show that noncompliance is widespread among both. Certificate transparency [56] should be deployed without delay (§8). The revocation story—particularly the ad hoc process by which the PA learns which certificates a misbehaving telecom holds—must be firmed up (§4). In the short term, the governance regime should also take action to prevent the "I deleted my certificate" attacks (§4) that frustrate post-call traceback. Our survey (§6) additionally suggests that STIR/SHAKEN weighs most heavily on small telecoms, with consequences for the privacy of the people they serve that we take up below.

STIR/SHAKEN certificate authorities and third-party AS/VS providers. Our certificate analytics in §7 are particularly unflattering for CAs in the STIR/SHAKEN ecosystem: nearly half of trusted CA certificates and roughly the same fraction of telecom certificates violate the relevant specifications. We urge CAs to bring their issuance practices into line with current standards and to commit to publishing into a certificate transparency log. Third-party AS/VS providers should likewise recognize that their position concentrates the call metadata of a large fraction of US calls into a small number of corporate hands (§5, §6); the appropriate response is restrictive data-handling policies and transparent commitments to subscribers, not opportunistic reuse of that metadata in adjacent product lines.

Telecom operators. A STIR/SHAKEN-signed PASSporT is substantially more sensitive than ordinary call metadata: it carries publicly-verifiable cryptographic evidence that a particular call took place and, if RCD is enabled, evidence of what the call was about. Operators should treat such records accordingly, minimizing retention and limiting access. Where feasible, signing and verification should be performed on-

premises. Where a third-party AS or VS is used, operators should obtain written assurances about how the resulting metadata is handled; absent explicit contractual restrictions on data flow, subscribers' call metadata should be assumed reachable by the third party. As we detail in §5, several of the dominant third-party providers operate in adjacent business lines whose interests are not obviously aligned with subscriber privacy, and the contractual and reputational exposure that follows from a future incident falls on the originating telecom, not on the third party. Finally, operators should support the SIP 603-based key discovery mechanism we describe in §8, which is a prerequisite for the cryptographic improvements we propose.

Cryptographers. STIR/SHAKEN is the latest instance of a recurring design pattern: *cryptographic attestations issued by service providers to support abuse prevention and attribution.* DKIM is an earlier instance, and Facebook's message-franking system [49] a third. The instances differ instructively. Facebook's franking tags can be verified only by the platform, which sits on the message path; DKIM and STIR/SHAKEN instead use publicly verifiable signatures, and so produce long-lived, transferable evidence. Asymmetric message franking [85] was designed for exactly the setting STIR/SHAKEN presents: abuse is adjudicated by a party *off* the message path, yet the evidence should convince only that party. Each instance has surfaced novel definitional questions, yet the pattern itself has not been studied as a class; we believe it deserves systematic treatment. Within STIR/SHAKEN specifically, the AMF blind-signing protocol BAMF and the verifier-hiding blindness notion we introduce (§8.2) may be of independent interest beyond telephony, as would further study of post-quantum deniable signatures. More broadly, cryptographers should offer help with deployment efforts where possible. We recognize that academic incentives are not always aligned with this kind of work, but as the STIR/SHAKEN experience shows, the cost of being absent when these systems are designed is borne by hundreds of millions of users.

A disparate privacy impact. The privacy costs described above are not borne evenly across the telephone network. We believe the costs fall hardest on the subscribers of small providers, namely those in rural and high-cost service areas. Though we lack the data to determine this definitively, we believe it follows from two of our findings. First, our respondents describe the compliance burden of STIR/SHAKEN as falling hardest on small telecoms (§6). A majority called it a burden on their operations, and their open responses locate that burden in the cost of compliance; one small carrier called third-party certificate and signing services "ridiculously priced," and another said big companies use STIR/SHAKEN to "squash small companies." Second, the usual way to discharge that burden is to outsource: 10 of the 17 small providers that provided an answer use an off-premise third-party AS or VS. This is the remedy the industry itself prescribes: "Hosted SHAKEN" and "Carrier SHAKEN" (two outsourcing arrangements described in §3) exist precisely as a way for small providers to reduce the cost of authenticating their traffic [10, 48]. But outsourcing is precisely the deployment in which PASSporT metadata leaves the provider for a party that would otherwise never have seen it (§5). Whether a call's metadata is exposed to an off-path third party therefore depends substantially on whether the subscriber's carrier was large enough to keep signing and verification in house. That is not a fact the subscriber chooses or can observe. Nor is the exposure diversified. Rather, the providers that outsource concentrate into the same handful of vendors (§6), so their subscribers' metadata accumulates in aggregated form, increasing its value. To whatever extent small carriers serve populations that larger ones do not—rural and high-cost service areas being the obvious case—the privacy loss tracks those populations too. The cost of STIR/SHAKEN compliance has so far been argued over as a question of which carriers can afford to stay in business. It is also a question of whose call metadata ends up in a third party's hands, and that one is settled for subscribers rather than by them.

Policy makers and end users. STIR/SHAKEN should not be confused with robocall prevention: it is an *attribution* mechanism that operates after a call has been routed, and our survey (§6) shows that telecoms themselves are divided on whether it meaningfully reduces unwanted calls. The TRACED Act mandate created an apparatus for after-the-fact tracing; whether that apparatus has reduced fraud at the receiver is a separate empirical question that deserves continued attention. End users, meanwhile, should be aware that STIR/SHAKEN-attested calls now carry substantially more persistent identifying metadata than unattested ones (§5), that the "verified" indicators surfaced by handsets can disagree with the field actually validated (§4), and that the federally-mandated *67 privacy mechanism is broken at at least one large provider (§7). More broadly, subscribers currently have no way to learn how far their call metadata travels: which third parties sit on the signing and verification path for their calls, and what those parties may do with what

they see. Policy makers should close this gap with mandatory disclosure. The FCC's broadband consumer labels [5] offer a template: a standardized per-carrier disclosure naming the AS/VS providers that handle a subscriber's PASSporTs, along with their retention and reuse terms, would let subscribers, and the regulators acting for them, see a cost that is currently invisible.

12 Acknowledgements

We would like to thank the anonymous reviewers of IEEE S&P 2026 for helpful comments. We thank Ryan Hurst of Martini Security for his assistance in understanding the telecom ecosystem. We thank Richard Barnes and other attendees of RWC '24 for helpful discussions following our presentation there. We would also like to thank Yoon Sung Ji of the University of Michigan for his help in conducting the survey. This work was supported by NSF grant #2236784 and a generous gift from Meta. Finally, the last author would like to thank the bluegrass band The Infamous Stringdusters, whose cover of The Cure's "Just Like Heaven" he listened to eighty-one times in a row during this work's initial submission deadline.

References

[1] Caller identification information in successor or replacement technologies. Technical report, Federal Communications Commission, June 2011. Submitted pursuant to Public Law No. 111-331. URL: https://docs.fcc.gov/public/attachments/DA-11-1089A1_Rcd.pdf.

[2] Rules and policies regarding calling number identification service – caller id; waiver of federal communications commission regulations at 47 c.f.r. § 64.1601(b) on behalf of jewish community centers. Notice of Proposed Rulemaking FCC 17-76, Federal Communications Commission, June 2017. CC Docket No. 91-281; Adopted and released June 22, 2017. URL: https://docs.fcc.gov/public/attachments/FCC-17-76A1.pdf.

[3] Atis technical report on a framework for display of verified caller id. Technical Report ATIS-1000081, Alliance for Telecommunications Industry Solutions, May 2018. Approved May 2018. URL: https://access.atis.org/higherlogic/ws/public/download/40779.

[4] Call authentication trust anchor; implementation of traced act section 6(a) — knowledge of customers by entities with access to numbering resources. Technical Report FCC 20-42, Federal Communications Commission, March 2020. Adopted and released March 31, 2020; WC Docket Nos. 17-97, 20-67. URL: https://docs.fcc.gov/public/attachments/FCC-20-42A1.pdf.

[5] Empowering broadband consumers through transparency, report and order and further notice of proposed rulemaking. Report and Order FCC 22-86, CG Docket No. 22-2, Federal Communications Commission, November 2022. URL: https://docs.fcc.gov/public/attachments/FCC-22-86A1.pdf.

[6] Extending stir/shaken over tdm. Technical Report ATIS-1000095.v002, Alliance for Telecommunications Industry Solutions, August 2022. Approved August 26, 2022. URL: https://access.atis.org/apps/group_public/download.php/67542/ATIS-1000095.v002.pdf.

[7] Robocall call block notification. Technical Report ATIS-1000099, Alliance for Telecommunications Industry Solutions, August 2022. Approved August 16, 2022. URL: https://access.atis.org/higherlogic/ws/public/download/67424.

[8] Signature-based handling of asserted information using tokens (shaken): Governance model and certificate management. Technical Report ATIS-1000080.v005, Alliance for Telecommunications Industry Solutions, December 2022. Approved December 12, 2022. URL: https://access.atis.org/higherlogic/ws/public/download/69428.

[9] Voip interconnection over the public internet. Technical Report ATIS-1000100, Alliance for Telecommunications Industry Solutions, December 2022. Approved December 12, 2022. URL: https://access.atis.org/higherlogic/ws/public/download/69420/ATIS-1000100.pdf.

[10] Eighth report and order. Report and Order FCC 24-120, Federal Communications Commission, November 2024. WC Docket 17-97; Adopted November 21, 2024 and released November 22, 2024. URL: https://docs.fcc.gov/public/attachments/FCC-24-120A1.pdf.

[11] Call authentication trust anchor. Notice of Proposed Rulemaking FCC 25-25, Federal Communications Commission, April 2025. WC Docket No. 17-97; Adopted April 28, 2025; Released April 29, 2025. URL: https://docs.fcc.gov/public/attachments/FCC-25-25A1.pdf.

[12] Call authentication trust anchor. Notice of Proposed Rulemaking FCC-CIRC2504-03, Federal Communications Commission, April 2025. WC Docket No. 17-97; Circulated April 7, 2025. URL: https://docs.fcc.gov/public/attachments/DOC-410645A1.pdf.

[13] Eighth report and order. Report and Order FCC 25-15, Federal Communications Commission, February 2025. CG Docket 17-59; Adopted February 27, 2025 and released February 28, 2025. URL: https://docs.fcc.gov/public/attachments/FCC-25-15A1.pdf.

[14] Masayuki Abe. A secure three-move blind signature scheme for polynomially many signatures. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 136–151. Springer, Berlin, Heidelberg, May 2001. doi:10.1007/3-540-44987-6_9.

[15] Masayuki Abe, Miyako Ohkubo, and Koutarou Suzuki. 1-out-of-n signatures from a variety of keys. In Yuliang Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 415–432. Springer, Berlin, Heidelberg, December 2002. doi:10.1007/3-540-36178-2_26.

[16] Masayuki Abe and Tatsuaki Okamoto. Provably secure partially blind signatures. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 271–286. Springer, Berlin, Heidelberg, August 2000. doi:10.1007/3-540-44598-6_17.

[17] David Adei, Varun Madathil, Sathvik Prasad, Bradley Reaves, and Alessandra Scafuro. Jäger: Automated telephone call traceback. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *ACM CCS 2024*, pages 2042–2056. ACM Press, October 2024. doi:10.1145/3658644.3690290.

[18] David Adei, Varun Madathil, Nithin Shyam S., and Bradley Reaves. Secure and efficient out-of-band call metadata transmission, 2025. URL: https://arxiv.org/abs/2509.12582, arXiv:2509.12582.

[19] Alliance for Telecommunications Industry Solutions. Signature-based handling of asserted information using tokens (shaken): Out-of-band passport transmission involving tdm networks. Technical Report ATIS-1000096, Alliance for Telecommunications Industry Solutions, July 2021. Approved July 15, 2021. URL: https://access.atis.org/higherlogic/ws/public/download/60535.

[20] Alliance for Telecommunications Industry Solutions. Signature-based handling of asserted information using tokens (shaken): Calling name and rich call data handling procedures. Technical Report ATIS-1000094.v002, Alliance for Telecommunications Industry Solutions, June 2025. Approved April 30, 2025. URL: https://access.atis.org/higherlogic/ws/public/download/81545/ATIS-1000094.v002.pdf.

[21] Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova. On the (in)security of ROS. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 33–53. Springer, Cham, October 2021. doi:10.1007/978-3-030-77870-5_2.

[22] Joshua M. Bercu and Jessica Thompson. Re: Enforcement bureau requests information on the status of private-led traceback efforts of suspected unlawful robocalls, eb docket no. 20-195, da 25-261, May 2025. SHAKEN claims, page 2. URL: https://www.fcc.gov/ecfs/document/1050176246723/1.

[23] Babette Boliek and Eric Burger. Beating back unwanted robocalls. https://www.fcc.gov/news-events/blog/2019/06/05/beating-back-unwanted-robocalls, June 2019. Federal Communications Commission Blog.

[24] Tianjie Cao, Dongdai Lin, and Rui Xue. Universal designated-verifier proxy blind signatures for e-commerce. In Yue Hao, Jiming Liu, Yu-Ping Wang, Yiu-ming Cheung, Hujun Yin, Licheng Jiao, Jianfeng Ma, and Yong-Chang Jiao, editors, *Computational Intelligence and Security*, pages 1036–1041, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[25] Cisco Webex. Caller identification and call redirection. https://help.webex.com/en-us/article/nl5cvg6/Caller-Identification-and-Call-Redirection, February 2025. Accessed: 2025-06-04.

[26] Federal Communications Commission. Targeting and eliminating unlawful text messages. Notice of Proposed Rulemaking FCC 22-72, Federal Communications Commission, September 2022. CG Docket No. 21-402; Adopted September 23, 2022; Released September 27, 2022. URL: https://docs.fcc.gov/public/attachments/FCC-22-72A1_Rcd.pdf.

[27] STI-GA Technical Committee. Technical committee best practices, November 2020. URL: https://sti-ga.atis.org/wp-content/uploads/2020/12/Technical-Committee-Best-Practices.pdf.

[28] US Congress. 18 usc ch. 87a. - national do not call registry. Accessed June 6, 2025. URL: https://uscode.house.gov/view.xhtml?req=granuleid%3AUSC-prelim-title15-chapter87A&edition=prelim.

[29] US Congress. Truth in caller id act of 2009.

[30] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 174–187. Springer, Berlin, Heidelberg, August 1994. doi:10.1007/3-540-48658-5_19.

[31] Anthony Cresti. International regulators eye call authentication. *The Fast Mode*, January 2023. Accessed: 2025-06-04. URL: https://www.thefastmode.com/expert-opinion/29826-international-regulators-eye-call-authentication.

[32] Changlai Du, Hexuan Yu, Yang Xiao, Y. Thomas Hou, Angelos D. Keromytis, and Wenjing Lou. UCBlocker: Unwanted call blocking using anonymous authentication. In Joseph A. Calandrino and Carmela Troncoso, editors, *USENIX Security 2023*, pages 445–462. USENIX Association, August 2023. URL: https://www.usenix.org/conference/usenixsecurity23/presentation/du.

[33] Facebook Inc. Messenger secret conversations technical whitepaper. Technical report, Facebook Inc., May 2017. Originally published July 8, 2016. URL: https://about.fb.com/wp-content/uploads/2016/07/messenger-secret-conversations-technical-whitepaper.pdf.

[34] FDIC. Fdic consumer compliance examination manual. Accessed June 6, 2025. URL: https://www.fdic.gov/resources/supervision-and-examinations/consumer-compliance-examination-manual/documents/8/viii-5-1.pdf.

[35] Federal Communications Commission. 47 cfr § 64.1601 – delivery requirements and privacy restrictions. https://www.ecfr.gov/current/title-47/chapter-I/subchapter-B/part-64/subpart-P/section-64.1601. Accessed: 2025-06-04.

[36] Federal Communications Commission. Robocall mitigation database. https://www.fcc.gov/robocall-mitigation-database. Accessed May 30, 2025.

[37] Federal Communications Commission. Telephone number portability, first report and order and further notice of proposed rulemaking. Technical Report FCC 96-286, Federal Communications Commission, July 1996. CC Docket No. 95-116, RM 8535; adopted June 27, 1996. URL: https://transition.fcc.gov/Bureaus/Common_Carrier/Orders/1996/fcc96286.txt.

[38] Federal Communications Commission. Eb issues final determination order against belthrough llc. https://www.fcc.gov/document/eb-issues-final-determination-order-against-belthrough-llc, March 2026. FCC Report and Order, FCC DA-26-237, Belthrough LLC.

[39] Marc Fischlin, Patrick Harasser, and Christian Janson. Signatures from sequential-OR proofs. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 212–244. Springer, Cham, May 2020. doi:10.1007/978-3-030-45727-3_8.

[40] Alliance for Telecommunications Industry Solutions. Signature-based handling of asserted information using tokens (shaken). Technical Report ATIS-1000074.v003, Alliance for Telecommunications Industry Solutions, August 2022. Approved August 16, 2022. URL: https://access.atis.org/higherlogic/ws/public/download/67436.

[41] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Cham, August 2018. doi:10.1007/978-3-319-96881-0_2.

[42] Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. Blind Schnorr signatures and signed ElGamal encryption in the algebraic group model. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 63–95. Springer, Cham, May 2020. doi:10.1007/978-3-030-45724-2_3.

[43] Georg Fuchsbauer and Mathias Wolf. Concurrently secure blind schnorr signatures. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part II*, volume 14652 of *LNCS*, pages 124–160. Springer, Cham, May 2024. doi:10.1007/978-3-031-58723-8_5.

[44] Grandstream Networks. *GRP261x / GRP262x / GRP263x / GRP2670 / GRP2650 Series – Administration Guide*. Accessed: 2025-06-04.

[45] Matthew Green. Ok google: please publish your dkim secret keys. Accessed June 5, 2025. URL: https://blog.cryptographyengineering.com/2020/11/16/ok-google-please-publish-your-dkim-secret-keys/.

[46] Matthew Green. Ok Google: please publish your DKIM secret keys. A Few Thoughts on Cryptographic Engineering (blog), November 2020. URL: https://blog.cryptographyengineering.com/2020/11/16/ok-google-please-publish-your-dkim-secret-keys/.

[47] Industry Traceback Group. Industry traceback group policies and procedures, April 2022. URL: https://tracebacks.org/wp-content/uploads/2022/04/ITG-Policies-and-Procedures-Updated-Apr-2022.pdf.

[48] NANC Call Authentican Trust Anchor Working Group. Deployment of stir/shaken by small voice service providers, October 2021. URL: https://docs.fcc.gov/public/attachments/DOC-377426A1.pdf.

[49] Paul Grubbs, Jiahui Lu, and Thomas Ristenpart. Message franking via committing authenticated encryption. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 66–97. Springer, Cham, August 2017. doi:10.1007/978-3-319-63697-9_3.

[50] iconectiv. About npac. Technical report, iconectiv, 2026. Number Portability Administration Center (NPAC); iconectiv is the U.S. Local Number Portability Administrator (LNPA). Accessed 2026-07-20. URL: https://numberportability.com/about/npac.

[51] Secure telephone identity revisited (stir) working group charter. https://datatracker.ietf.org/wg/stir/charter/, December 2024. Working Group in the Applications and Real-Time Area (ART), Chairs: Ben Campbell, Russ Housley.

[52] Rawane Issa, Nicolas Alhaddad, and Mayank Varia. Hecate: Abuse reporting in secure messengers with sealed sender. In Kevin R. B. Butler and Kurt Thomas, editors, *USENIX Security 2022*, pages 2335–2352. USENIX Association, August 2022. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/issa.

[53] Simon Josefsson and Ilari Liusvaara. Edwards-Curve Digital Signature Algorithm (EdDSA). RFC 8032, January 2017. URL: https://www.rfc-editor.org/info/rfc8032, doi:10.17487/RFC8032.

[54] Julia Kastner, Julian Loss, and Jiayu Xu. On pairing-free blind signature schemes in the algebraic group model. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part II*, volume 13178 of *LNCS*, pages 468–497. Springer, Cham, March 2022. doi:10.1007/978-3-030-97131-1_16.

[55] Michael Klooß and Michael Reichle. Blind signatures from proofs of inequality. Cryptology ePrint Archive, Paper 2024/2076, 2024. URL: https://eprint.iacr.org/2024/2076.

[56] Ben Laurie, Adam Langley, and Emilia Kasper. Certificate Transparency. RFC 6962, June 2013. URL: https://www.rfc-editor.org/info/rfc6962, doi:10.17487/RFC6962.

[57] TransUnion LLC. Trucontact phone behavior intelligence: Increase right-party contact rates through authoritative contact identity and behavioral insights. Online, 2025. Product Web Page. URL: https://www.transunion.com/solution/trucontact/customer-contact-intelligence/phone-behavior-intelligence.

[58] Anna Lysyanskaya. Security analysis of RSA-BSSA. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 251–280. Springer, Cham, May 2023. doi:10.1007/978-3-031-31368-4_10.

[59] Jonathan Mayer, Patrick Mutchler, and John C. Mitchell. Evaluating the privacy properties of telephone metadata. *Proceedings of the National Academy of Sciences*, 113(20):5536–5541, 2016. URL: https://www.pnas.org/doi/10.1073/pnas.1508081113, doi:10.1073/pnas.1508081113.

[60] netnumber. The netnumber services registry(nnsr®). Accessed June 5, 2025. URL: https://netnumber.com/the-netnumber-services-registry/.

[61] a TransUnion Company Neustar. Restore trust in outbound communications. Online, 2022. White paper. URL: https://www.cdn.neustar/resources/whitepapers/risk/neustar-restore-trust-in-outbound-communications.pdf.

[62] First Orion and Neustar Inc. First orion and neustar partner to speed adoption of branded calling and stir/shaken call authentication. *Yahoo Finance*, 2022. January 25, 2022. URL: https://finance.yahoo.com/news/first-orion-neustar-partner-speed-140000018.html.

[63] J. Peterson, C. Jennings, E. Rescorla, and C. Wendt. Authenticated identity management in the session initiation protocol (sip). RFC 8224, February 2018. URL: https://www.rfc-editor.org/info/rfc8224, doi:10.17487/RFC8224.

[64] J. Peterson and S. Turner. Secure telephone identity credentials: Certificiates. RFC 8226, February 2018. URL: https://www.rfc-editor.org/info/rfc8226, doi:10.17487/RFC8226.

[65] Jon Peterson and Chris Wendt. Messaging use cases and extensions for secure telephone identity revisited (stir). RFC 9475, Internet Engineering Task Force, December 2023. URL: https://www.rfc-editor.org/rfc/rfc9475.html, doi:10.17487/RFC9475.

[66] Sathvik Prasad, Aleksandr Nahapetyan, and Bradley Reaves. Characterizing robocalls with multiple vantage points. In *2025 IEEE Symposium on Security and Privacy (SP)*, pages 96–96. IEEE Computer Society, 2025.

[67] Xianrui Qin, Cailing Cai, and Tsz Hon Yuen. One-more unforgeability of blind ECDSA. In *Computer Security – ESORICS 2021*, volume 12973 of *Lecture Notes in Computer Science*, pages 313–331. Springer, 2021. doi:10.1007/978-3-030-88428-4_16.

[68] Bradley Reaves, Logan Blue, and Patrick Traynor. AuthLoop: End-to-end cryptographic authentication for telephony over voice channels. In Thorsten Holz and Stefan Savage, editors, *USENIX Security 2016*, pages 963–978. USENIX Association, August 2016. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/reaves.

[69] Jonathan Rosenberg, Henning Schulzrinne, Gonzalo Camarillo, Alan Johnston, Jon Peterson, Robert Sparks, Mark Handley, and Eve Schooler. Rfc3261: Sip: session initiation protocol, 2002.

[70] Rajeev Anand Sahu, Agnese Gini, and Ankan Pal. Supersingular isogeny-based designated verifier blind signature. Cryptology ePrint Archive, Paper 2019/1498, 2019. URL: https://eprint.iacr.org/2019/1498.

[71] Margot Saunders and Chris Frascella. Scam robocalls: Telecom providers profit. Technical report, National Consumer Law Center and Electronic Privacy Information Center, June 2022. URL: https://www.nclc.org/wp-content/uploads/2022/09/Rpt_Scam_Robocalls.pdf.

[72] Dominique Schröder and Dominique Unruh. Security of blind signatures revisited. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 662–679. Springer, Berlin, Heidelberg, May 2012. doi:10.1007/978-3-642-30057-8_39.

[73] Christian Shepherd, Cate Cadell, Ellen Nakashima, Joseph Menn, and Aaron Schaffer. Leaked files from chinese firm show vast international hacking effort. *The Washington Post*, February 2024. Accessed: 2025-06-04. URL: https://www.washingtonpost.com/world/2024/02/21/china-hacking-leak-documents-isoon/.

[74] STI-GA. Improper attestation. https://sti-ga.atis.org/wp-content/uploads/2023/05/Improper-Attestation-Final.pdf, May 2024. Accessed: 2025-06-05. URL: https://sti-ga.atis.org/wp-content/uploads/2023/05/Improper-Attestation-Final.pdf.

[75] STI-GA. Policy decision 002: Certificate policy, August 2024. URL: https://authenticate.iconectiv.com/sites/authenticate/files/2025-02/CP_v1.4.2.pdf.

[76] STI-GA. Policy decision 003: Certificate revocation policy, October 2024. URL: https://cdn.atis.org/sti-ga.atis.org/2024/10/07193732/241004-Policy-003-SPC-token-Revocation-Policy-Final.pdf.

[77] Stefano Tessaro and Chenzhi Zhu. Short pairing-free blind signatures with exponential security. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 782–811. Springer, Cham, May / June 2022. doi:10.1007/978-3-031-07085-3_27.

[78] TransNexus. Branded calling. https://transnexus.com/branded-calling/. Accessed: 2025-06-04.

[79] TransNexus. Secure telephone identity call placement service (sti-cps). https://cps.transnexus.com/. Accessed: 2025-06-04.

[80] TransNexus. Stir/shaken statistics from april 2025. https://transnexus.com/blog/2025/shaken-statistics-april/, May 2025.

[81] TransNexus. Stir/shaken statistics from april 2025. *TransNexus Blog*, May 2025. Accessed: 2025-06-04. URL: https://transnexus.com/blog/2025/shaken-statistics-april/.

[82] TransUnion. What is branded calling? https://www.transunion.com/faq/what-is-branded-calling. Accessed: 2025-06-04.

[83] TransUnion. Transunion accelerates growth of identity-based solutions with agreement to acquire neustar for $3.1 billion. Online, September 2021. Press release, TransUnion Newsroom. URL: https://newsroom.transunion.com/transunion-accelerates-growth-of-identity-based-solutions--with-agreement-to-acquire-neustar-for-31-billion/.

[84] Twilio. What is branded calling? https://www.twilio.com/en-us/blog/what-is-branded-calling, March 2025. Accessed: 2025-06-04.

[85] Nirvan Tyagi, Paul Grubbs, Julia Len, Ian Miers, and Thomas Ristenpart. Asymmetric message franking: Content moderation for metadata-private end-to-end encryption. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 222–250. Springer, Cham, August 2019. doi:10.1007/978-3-030-26954-8_8.

[86] A.J. Vicens. Us adds 9th telcom to list of companies hacked by chinese-backed salt typhoon cyberespionage. *Reuters*, December 2024. Accessed: 2025-06-04. URL: https://www.reuters.com/technology/cybersecurity/us-adds-9th-telcom-list-companies-hacked-by-chinese-backed-salt-typhoon-2024-12-27/.

[87] Enze Wang, Jianjun Chen, Wei Xie, Chuhan Wang, Yifei Gao, Zhenhua Wang, Haixin Duan, Yang Liu, and Baosheng Wang. Where URLs become weapons: Automated discovery of SSRF vulnerabilities in web applications. In *2024 IEEE Symposium on Security and Privacy*, pages 239–257. IEEE Computer Society Press, May 2024. doi:10.1109/SP54263.2024.00198.

[88] Malte Wessels, Simon Koch, Giancarlo Pellegrino, and Martin Johns. SSRF vs. developers: A study of SSRF-defenses in PHP applications. In Davide Balzarotti and Wenyuan Xu, editors, *USENIX Security 2024*. USENIX Association, August 2024. URL: https://www.usenix.org/conference/usenixsecurity24/presentation/wessels.

[89] Wikileaks. Dkim verification. Accessed June 5, 2025. URL: https://wikileaks.org/DKIM-Verification.html.

[90] Yealink. Line identification presentation. https://support.yealink.com/en/portal/knowledge/show?id=64995b696a27da76bd0718f0, November 2023. Accessed: 2025-06-04.

A Further Survey Methodology

Methodology. Between February and March 2024, we conducted a survey of telecoms. Our list of telecoms and their contact information came from the FCC's Robocall Mitigation Database. The database contains a list of all telecoms operating in the United States, their contact information, and their STIR/SHAKEN implementation status. The FCC maintains this database to track compliance with the TRACED Act. We only contacted telecoms that were listed as having a "Complete STIR/SHAKEN Implementation."

To better understand what types of questions to ask, we spoke with industry experts and crafted a short list of multiple choice and open-ended questions. We then spoke with five telecoms to test our survey and refine our questions. During that process, we learned of a novel approach fraudsters are attempting to use to circumvent STIR/SHAKEN (detailed below). We made sure to include questions about this attack in our survey.

After finalizing our survey, we created a Google Form. We entered the responses from these five initial telecoms where the questions matched the final survey. We then sent a link to the Google form to the emails listed for 28 random telecoms in the FCC's database. After receiving no responses, we began calling telecoms and conducting the survey over the phone. For free response questions, we included direct quotes and paraphrased responses. Overall, we contacted 331 telecoms and received a total of 29 responses. Our survey population consisted of 2,796 U.S. telecoms listed as having a complete STIR/SHAKEN implementation. The database as a whole is considerably larger as it also lists telecoms with partial or no STIR/SHAKEN implementation, as well as many foreign providers.

Before conducting the survey, respondents were informed that their participation was voluntary, every question is optional, and were assured their responses would be kept anonymous. If respondents did not understand a question, we attempted to clarify it or rephrase it. If a respondent refused to answer a question or was unsure of the answer, the question was skipped. Not every respondent answered every question. Percentages are computed over those who provided an answer to the relevant question.

Respondents were first asked to self-report the size of their telecom: small, medium, or large. We did not provide definitions for our size categories and asked respondents to judge relative to the industry.

We then asked to confirm their STIR/SHAKEN implementation status. About 7% of respondents (2) reported that they did not use STIR/SHAKEN. For respondents that indicated they did not use STIR/SHAKEN, the survey was terminated. Since we sometimes spoke to the owner of the telecom and other times to technical personnel, we believe this inconsistency with the FCC's database is more likely attributable to ignorance of what STIR/SHAKEN is rather than inaccuracies in the FCC's database. For the remaining respondents, we asked them a series of multiple choice and open-ended questions.

We acknowledge that our survey is likely biased towards smaller telecoms as they are more likely to answer the phone and respond to our survey. Indeed, of the 26 respondents who reported a size, 80.8% self-identified as small telecoms, 15.4% as medium, and 3.8% as large. The remaining 3 respondents did not report a size.

After discussing our results with industry experts, they indicated that they believed our survey results were relatively representative of the industry.

A.1 Other Survey Results

Disproportionate burden on smaller telecoms. We asked telecoms about their experiences with STIR/SHAKEN. 56% of respondents agreed or strongly agreed that STIR/SHAKEN is a burden on their telecom with only 16% disagreeing or strongly disagreeing. Broken down by size, 55% of small telecoms (11 of the 20 that answered the question) indicated that STIR/SHAKEN is a burden on their telecom, compared to 60% of medium and large telecoms (3 of 5). Our sample contains too few medium and large respondents to support a statistical comparison. At the end of the survey, respondents were given two opportunities to provide any additional comments or feedback. Numerous providers expressed frustration at the perceived burden STIR/SHAKEN has placed on their telecoms and at the cost of third party solutions, one calling them "ridiculously priced." A variety of telecoms noted that STIR/SHAKEN is particularly burdensome for small telecoms, with one going as far to say "big companies" use STIR/SHAKEN to "squash small companies." Similarly, a medium-sized company complained that STIR/SHAKEN costs 50% of their [revenue] and that STIR/SHAKEN is going to "kill all the small companies...the competitors."

STIR/SHAKEN's effectiveness. We wanted to understand the perceived effectiveness of STIR/SHAKEN, whether STIR/SHAKEN accomplishes its design goals. Responses were mixed, varying from extremely negative to lukewarm positive. Even telecoms that considered STIR/SHAKEN effective often recognized that there was room for significant improvement, but were optimistic that STIR/SHAKEN would improve over time. In total, 37.5% of telecoms agreed or strongly agreed that STIR/SHAKEN is effective at reducing robocalls (33.3% agreeing and 4.2% strongly agreeing). A larger share, 45.8%, disagreed or strongly disagreed (25% and 20.8%, respectively). The remaining 16.7% were neutral.

In the free response section, telecoms expressed a variety of opinions on STIR/SHAKEN's effectiveness. Many companies complained of improper STIR/SHAKEN implementations resulting in their customer's calls falsely being marked as spam with one stating "I hate STIR/SHAKEN." Again, smaller telecoms expressed frustration with larger telecoms for labeling calls that didn't originate from other large telecoms as spam. One company stated explicitly that STIR/SHAKEN "doesn't help at all" and another labeled it "a waste." According to one company's internal statistics, "it has stopped zero calls" with robocallers simply switching to "more creative" methods of placing fraudulent calls.

Other companies expressed the opposite sentiment with a large degree of cautious optimism. One telecom called STIR/SHAKEN a "very effective tool if utilized properly" and another said it "seems to be working." A few providers suggested that STIR/SHAKEN would "eventually be effective" but that the FCC needed to improve enforcement. Others noted that STIR/SHAKEN deals well with the "80% of lowest level spam", whereas the remaining 20% use "more sophisticated" techniques.

Inquiries by fraudsters. During our preliminary conversations, one of the telecoms described an emerging means by which fraudsters are attempting to circumvent STIR/SHAKEN. The telecom described receiving numerous inquiries from bad actors asking to route their traffic through the telecom's network thereby receiving attestations for their calls. We added a question to our survey asking telecoms if they had received any inquiries from fraudsters and if so to describe the inquiries. While only 17.4% of telecoms reported receiving inquiries, these telecoms reported receiving a considerable number of inquiries. One telecom reported having received "hundreds" in 2023 and "dozens in the last 90 days." Other telecoms reported receiving many emails and calls, especially from foreign countries. In describing their procedures for detecting fraudulent customers, one telecom indicated that during the onboarding process they can sometimes hear fraudsters working in the background. After onboarding, the telecom monitors call recordings for fraudulent activity. Telecoms described annoyance with the inquiries.