Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: 2 December 2025

F. Hao
University of Warwick
B. Thomas
Squire Technologies Ltd
S. Smith
trueCall Ltd
MA. Azad
Birmingham City University
31 May 2025

Caller ID Verification In Heterogeneous Telecommunication Networks
draft-hao-civ-01

Abstract

This document defines an extension to the INVITE header of the Session Initiation Protocol (SIP) to support a Caller ID Verification (CIV) method. CIV authenticates the caller ID of an incoming call through a challenge-and-response process across both IP and non-IP networks without requiring a trusted third party or a public key infrastructure. When receiving a call with a claimed phone number, the called party holds the call and sends a quickly terminated INVITE request (like a flash call) to that number, carrying a short 4-digit challenge embedded as part of the caller ID. A genuine caller would receive the challenge and respond by echoing the same digits, e.g., through DTMF (Dual-Tone Multi-Frequency). The proposed extension involves two changes to the INVITE header. First, it adds a new option tag, "civ", in the Supported header field of the INVITE request. This tag allows the calling party to indicate support for CIV in the initial call. Second, it adds a special value "civ-veri-call" for the Purpose parameter of the Call-Info header field. This value allows the called party to make a verification call, indicating the purpose of the call is to transmit a challenge rather than establish a phone conversation. CIV uses the standard Session-ID header in the INVITE request to allow the calling party to explicitly match the verification call with the initial call. Whilst this document focuses on IP networks, the same CIV protocol also works with non-IP networks (e.g., SS7) by including the "civ" tag, the "civ-veri-call" value and the session ID in the User-to-User Information (UUI) parameter.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Copyright Notice

Table of Contents

1. Introduction

In a SIP network, it is easy to modify the caller ID using number spoofing. Sometimes, there are legitimate reasons to alter the caller ID, e.g., showing a single outgoing number for an organisation or a toll-free number for the called party to dial back. However, fraudsters and scammers frequently abuse number spoofing to hide their identity or to pretend to call from trusted sources. This has become a global problem in the Telecom industry.

This document describes a Caller ID Verification (CIV) method based on a peer-reviewed paper published in ACM Transactions on Privacy and Security (ACM TOPS) by Wang et al. in 2023 [5]. CIV authenticates the caller ID using a challenge-response protocol, which works with both IP and non-IP networks without requiring any trusted third party or a public key infrastructure. While it is possible to implement CIV on the users' phones, this document focuses on implementing CIV on telecommunication switches in the Telecom cloud for best performance.

In a telecommunication system, there are three distinct authentication problems: 1) the authentication of the carrier (typically done by verifying the possession of a private key for digital signature), 2) the authentication of the caller (through verifying a password, the possession of a token, or biometrics) and 3) the authentication of the caller ID. CIV specifically addresses the third problem through verifying whether the calling party possesses (or owns) the phone number that is sent as the caller ID, whereas the identity of the caller can be traced later if needed via the range holder of that number (assuming appropriate KYC checks are in place, or the service to number can be turned off). Crucially, CIV is a decentralized method that does not require any trusted third party (or trusted certification authorities in a public key infrastructure). Such a decentralized method is useful in many practical applications where proving the possession of the caller ID is needed to prevent spoofing attacks without involving any trusted

third party. In practice, CIV can be combined with other authentication techniques (e.g., carrier authentication and caller authentication) to provide enhanced security. As an example, some banks use a system called Voice ID (or "my voice is my password") to authenticate the caller based on analyzing their voice biometrics in telephone banking, but AI-generated voices pose a real threat to defeat this caller authentication method completely [9]. Using CIV would force the attacker to call from the victim's phone (or to prove the ownership of the phone number that is registered with the bank), hence making the AI spoofing attack more difficult. Some mobile payment systems, such as Unified Payment Interface (UPI), use the mobile phone number as the unique identity to facilitate payments between users. Verifying the ownership of the phone number is an important step to prevent fraud.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [2] [3] when, and only when, they appear in all capitals, as shown here.

2. Overview of CIV

At a high level, CIV is inspired by a call-back verification method used in practice. To verify if the caller ID of an incoming call is genuine, the receiver may just hang up and call back the displayed number [8]. This call-back method ensures that the receiver talks to the genuine caller who possesses (or owns) that number. However, the manual call-back verification is slow and tedious. It may also incur a charge to the verifying party.

CIV follows a similar approach, but it automates the verification process across heterogeneous networks in a secure and efficient manner without incurring a charge to the verifying party. In CIV, the authentication of the caller ID is defined based on checking the possession of that number. Accordingly, it distinguishes legitimate and illegitimate modifications of the caller ID based on whether the caller possesses the number. Suppose the caller uses an unmodified number or a legitimately modified number they own. In both cases, they should receive a challenge from a call-back and hence should be able to respond accordingly.

Supporting CIV in SIP [1] involves making two changes to the INVITE header. First, we propose a new option tag "civ" in the Supported header of INVITE. This tag indicates that the caller supports CIV. Second, we propose a new value "civ-veri-call" for the Purpose

parameter in the Call-Info header of INVITE. The "civ-veri-call" value, paired with a unique resource indicator (URI) that can be the same as the URI in the TO header, indicates that the purpose of this INVITE request is to verify the caller ID in the initial call based on CIV.

Based on the two proposed changes, the CIV protocol in SIP works as follows.

1. The calling party initiates an INVITE request, containing an option tag "civ" in the Supported header to indicate support for CIV.

2. When receiving this call with the "civ" tag, the called party holds the call and sends an INVITE request to the displayed caller ID, containing the value "civ-veri-call" for the Purpose parameter in the Call-Info header. This INVITE request uses a caller ID that contains a short challenge (4 random digits) in the last four digits of the caller ID. The INVITE request is quickly terminated before it is answered (like a flash call). Subsequently, the called party unholds the call and waits for the response. The purpose of the "civ-veri-call" value is to make this INVITE request lightweight: Telecom networks that understand this value in the call path do not need to reserve resources to prepare for voice conversation, hence making call routing faster. For networks that do not understand this value, they will route this INVITE request as a normal call. CIV still works, although at a suboptimal speed.

3. If the original calling party is genuine, it should be able to receive the challenge and send a response (by echoing the same 4 digits) through Dual-Tone Multi-Frequency (DTMF) in the initial call to complete the authentication process. DTMF is universally supported in both IP and non-IP Telecom networks.

In the second step of CIV, the called party holds the call before starting the challenge-response verification process. The time it takes for the verification process to finish will need to be set as a grace period, as the user has not actually answered the call while the verification of the caller ID is in process. With the current design of CIV, we expect the verification time to be mostly determined by the transmission of 4 DTMF digits (typically 0.5 seconds). If required, this grace period can be excluded from the calling party’s call time in billing.

The verification call needs to be matched with the initial call. This can be done explicitly based on the Session-ID header [4] in the INVITE request. The session ID consists of 32 hexadecimal values.

It uniquely identifies a communication session end-to-end, from the originating device to the terminating device. Unlike the Call-ID header field (which may be modified by intermediate networks), the session ID remains the same throughout the session. Therefore, we can utilize the session ID in CIV as follows. In the initial call, the Session-ID header field contains a unique session ID while the Remote parameter is null (32 zeros). In the verification call, the Session-ID header field contains a different session ID while the Remote parameter contains the same session ID as in the initial call. The receiver of the verification call can therefore explicitly match the verification call with the initial call based on the session ID. The following shows examples of the relevant headers in the INVITE request.

Example of the selected headers in the initial SIP call:

```
|  To: "Bob" <sip:+1234567222@example.com>
|  From: "Alice" <sip:+1415555111@example.com>;tag=887s
|  Supported: civ
|  Session-ID: ab30317f1a784dc48ff824d0d3715d86;
|  remote=00000000000000000000000000000000
```

Example of the selected headers in the CIV verification call (note that the last four digits of Bob’s caller ID in the From header field have been modified to contain a random challenge):

```
|  To: "Alice" <sip:+1415555111@example.com>
|  From: "Bob" <sip:+1234569685@example.com>; tag=a48s
|  Call-Info: <sip:+1415555111@example.com>; purpose=civ-veri-call
|  Session-ID: 47755a9de7794ba387653f2099600ef2;
|  remote=ab30317f1a784dc48ff824d0d3715d86
```

3. Implementing CIV in the Telecom Cloud

This document proposes to implement CIV in the Telecom cloud by performing the challenge-response authentication process between switches of the originating and terminating carriers (instead of between the users’ phones). Assume Alice is the caller, and Bob is the callee. We first consider the cases where both the originating and the terminating carriers support CIV. Later, we will discuss the compatibility modes where only one or no carrier supports CIV in Section 4.

3.1. Threat Model

In our threat model, we assume that the attacker can freely modify the caller ID when initiating a call. The modified caller ID is permitted by the originating carrier (who may add a digital signature) and can pass through subsequent networks. We assume the attacker is not able to intercept calls. We note that a powerful adversary can intercept calls through the Law Enforcement Monitoring Facility (LEMF), SIM swap, and SS7 hacking [6], but this is beyond the capability of ordinary telephone scammers behind the number spoofing attacks.

3.2. CIV call state

When the INVITE request contains the "civ" option tag in the Supported header field, this indicates that the calling party is willing to support the CIV check of the presented caller ID. For such a call, the originating carrier shall keep a CIV call state, containing at least the following information: {caller, callee, call time, session ID}, where "caller" represents the caller's number (which may be the caller ID or an internal number in a PBX system), "callee" represents the callee's number, "call time" indicates the time when the initial call is made, and "session ID" is the value contained in the Session-ID header field. The following is an example.

Caller	Callee	Call time	Session ID
1234567222	1415555111	2024-02-27 15:35:20.311	ab30317f1a784dc48ff824d0d3715d86

Table 1: Example of a CIV call state

In this example, the call time is represented in the "yyyy-MM-dd HH:mm:ss.SSS" format. Alternatively, it may be represented using a 13-digit epoch time, i.e., the number of milliseconds since January 1, 1970.

The CIV call state is transient data, kept during the call setup phase only. The originating carrier writes a record of the state when making the initial call containing the "civ" tag, and deletes the record immediately after the call is connected or has failed (e.g., after a timeout).

3.3. Case 1: a legitimate caller uses the unmodified number

In this case (most common), a legitimate caller (Alice) uses the unmodified number (Alice's number) as the caller ID. Figure 1 describes the steps of operations.

1. Alice initiates a call to Bob and uses her unmodified number as the caller ID.

2. Alice's carrier sends an INVITE request to Bob with Alice's unmodified caller ID, including the "civ" tag in the Supported header.

3. Detecting the presence of the "civ" tag, Bob's carrier holds the incoming call and sends an INVITE request to Alice's number as a verification call. The INVITE request includes the "civ-veri-call" value and four random digits replacing the last four digits of the caller ID as a challenge (the rest digits can follow the same as Bob's number). The verification call is quickly terminated before it is answered, so it does not incur a charge. Bob's carrier connects Alice's initial call and waits for a response through DTMF.

4. Alice's carrier retrieves the 4-digit challenge from the caller ID of the quickly terminated verification call, and sends the same digits as a response through DTMF in the initial call. Otherwise, it quietly discards the challenge.

5. Bob's carrier checks the received DTMF digits against the challenge. If they match, the caller ID is "verified". Bob's phone starts ringing, showing the caller ID and the "verified" status of this number. If Bob's carrier does not receive the response within a time limit or the received response does not match the challenge, the verification has failed. The terminating carrier may reject the call, connect the call to the user's phone with an explicit visual/audio warning about the invalidated caller ID, send the call directly to voicemail, or follow any other call handling process made available by the network and selected by the user.

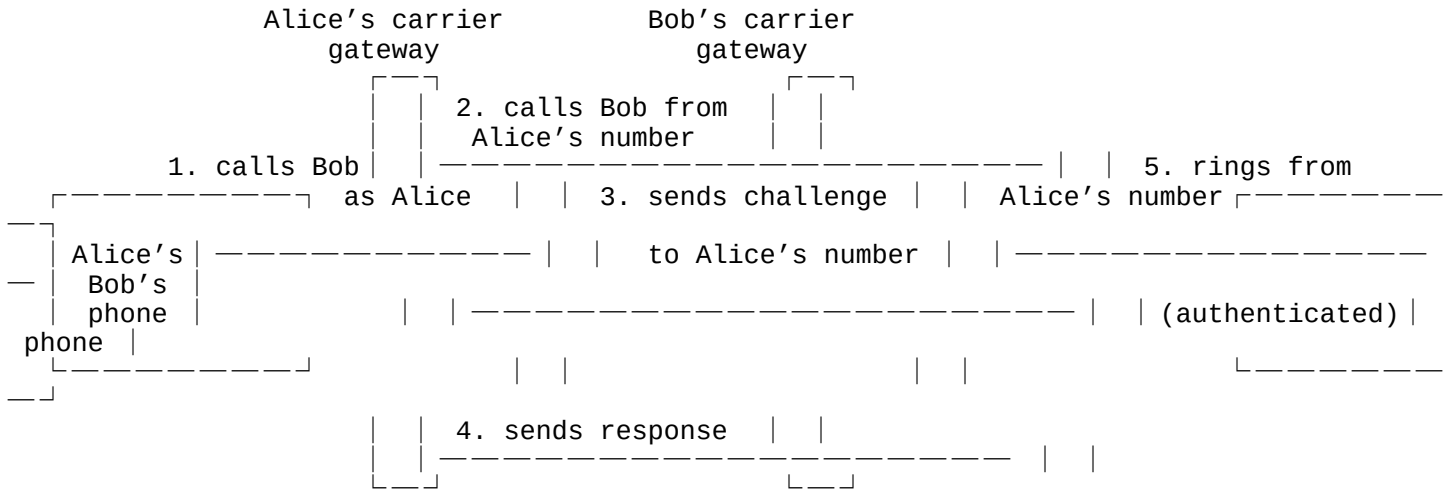

Figure 1: authenticated caller with an unmodified number

3.3.1. Call forwarding

The called party, Bob, may have a call-forwarding setting. When the call is forwarded to a voicemail, this is treated the same as being answered by a user; the CIV operation is not affected. When the call is forwarded to another number, CIV works as follows.

* Unconditional forwarding - the terminating carrier may just forward the call without doing CIV, leaving the next carrier to do the CIV check.

* Conditional forwarding - the terminating carrier will forward the call only when certain conditions are satisfied (e.g., busy, unanswered and unreachable). To test these conditions, the terminating carrier would need to ring the user's phone first. Without doing the CIV check, the ringing would display an unverified caller ID. To guarantee that only an authentic caller ID is displayed, the terminating carrier can do the CIV check before ringing the phone. When the call-forward conditions are satisfied, the carrier will forward the call. If the forwarded number belongs to the same carrier, there would be no need to perform the CIV check again, because the status of the verified caller ID is transferable within the domain of the same carrier. However, if the forwarded number belongs to a different carrier, the next carrier will most likely do the CIV check again. This implies that the originating carrier would need to perform the challenge-response process more than once. This is feasible as explained below.

In general, there are two ways to forward a call.

1. Redirection (client-based forwarding): the terminating carrier responds with a 3xx class response (commonly 302), indicating a new contact number. The originating carrier will send a new INVITE to the new number. Figure 2 shows an example. In this example, Bob’s phone is unreachable, so Bob’s carrier gateway responds with a 302 Moved Temporarily message with a new phone number in the Contact header field. Alice’s carrier gateway sends a new INVITE with a "civ" tag to the new number, and CIV works normally in the new session. RFC 3261 allows the 302 response to contain multiple new numbers. In that case, the calling party will send new INVITE requests to these numbers, either in sequence or in parallel. CIV supports both. Each INVITE request contains a unique session ID. Based on the session ID, the originating carrier can match the verification call to each outgoing call. CIV works normally in each call session.

2. Proxy (server-based forwarding): the terminating carrier directly forwards the INVITE request. If the forwarded number belongs to the same carrier, there would be no need for the carrier to do the CIV check again. But if it belongs to a different carrier, the next carrier can do the CIV check of the caller ID again as shown in Figure 3.

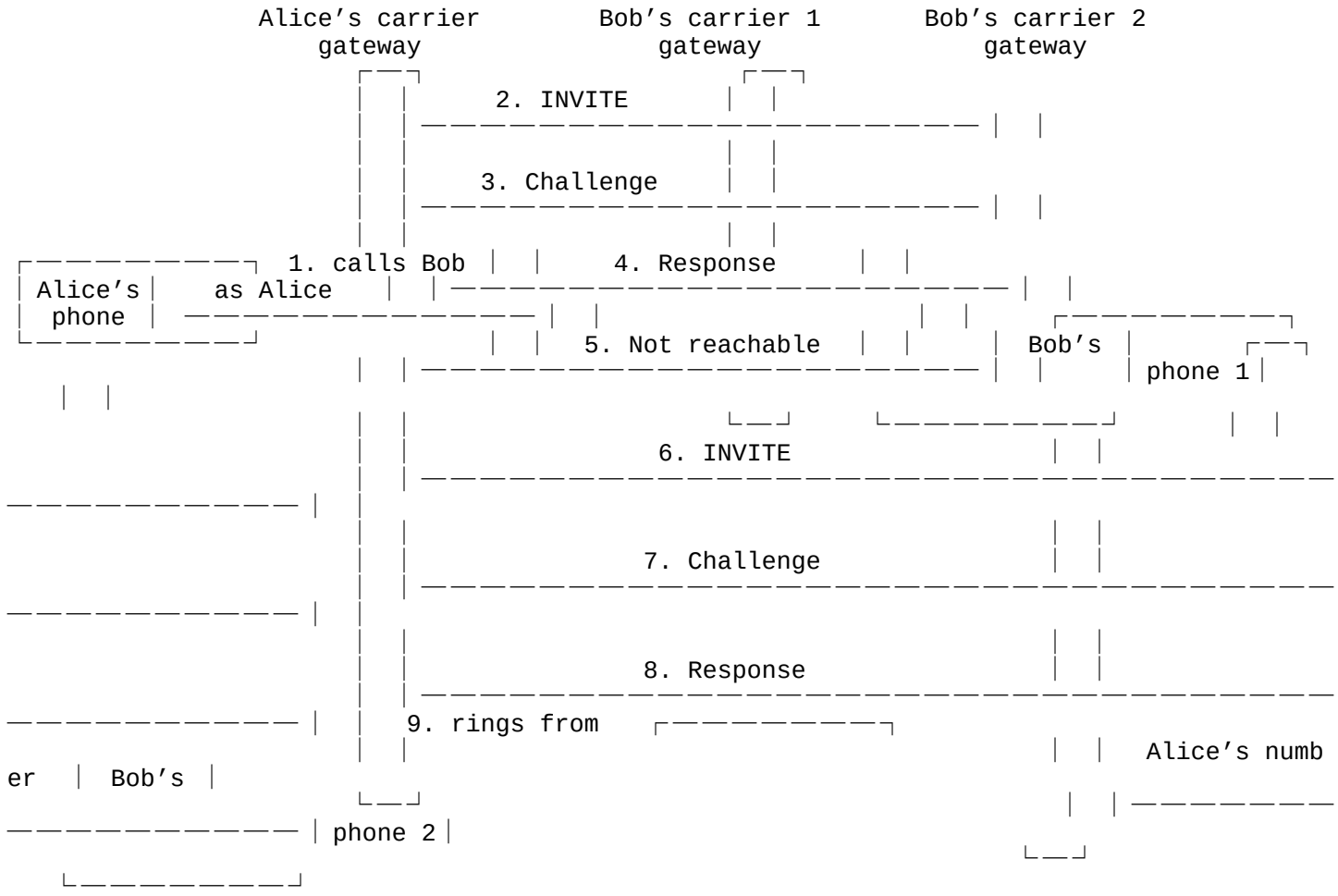

Figure 2: Call forwarding based on redirection

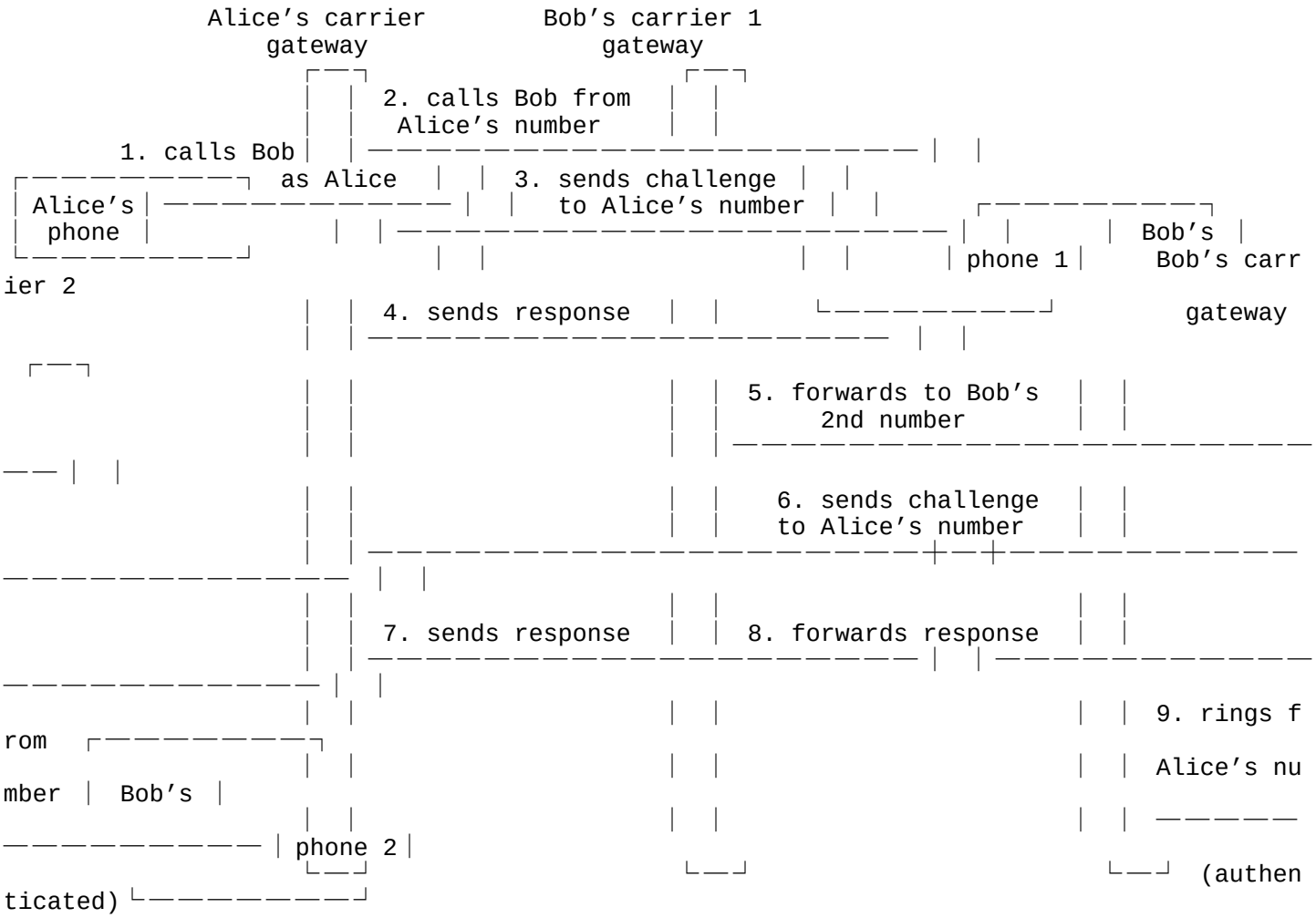

Figure 3: Call forwarding based on proxy forwarding

3.3.2. SIP forking

In proxy-based call forwarding, a single incoming SIP call may be simultaneously forwarded to multiple devices or extensions. For example, a call to a user's number can make the user's desk phone and the SIP softphone on his mobile ring at the same time. That is called SIP forking. We call a phone number "local" if it belongs to the same terminating carrier, and "remote" if it belongs to a different carrier. In general, within the same carrier, if the caller ID of an incoming call has been verified, the verified status is transferable within the trusted domain of the same carrier. In that case, there is no need for the carrier to perform the CIV check for this call again. However, when the call is forwarded to a remote number, the next carrier will want to check the caller ID (assuming that the previously verified status of the caller ID is not transferable to the next carrier). We consider how CIV works in the following SIP forking scenarios (see Figure 4).

1. The call is forked to multiple local numbers: after the terminating carrier performs the CIV check of the caller ID, all the numbers ring at the same time.

2. The call is forked to multiple local numbers and one remote number: after the terminating carrier performs the CIV check, all local numbers ring at the same time. Meanwhile, the call to the remote number is forwarded to the next carrier, which performs the CIV check and rings the phone accordingly. Assuming that the CIV check incurs a negligible delay, it would appear that all phones are ringing simultaneously.

3. The call is forked to multiple local numbers and more than one remote number: after the terminating carrier performs the CIV check, all local numbers ring at the same time. Meanwhile, the call is forwarded to each of the remote numbers sequentially, so the CIV check is done sequentially too. With the current design of CIV, the originating carrier can process multiple challenges received from an outgoing call, but it can only process them sequentially rather than in parallel. (From a security perspective, doing this sequentially helps prevent an attacker from using SIP forking to flood the originating carrier with simultaneous verification calls.) SIP forking to multiple remote numbers in parallel remains possible by removing the "civ" option tag before forking the INVITE request. In this case, CIV still adds value in security by allowing the carrier to validate the caller ID before forking the call in parallel to other carriers. This helps stop spoofing attacks in this specific forking scenario, although the caller ID of the forwarded call would appear unverified to the next carriers (due to the absence of the "civ" tag, the next carriers cannot do the CIV check from their end).

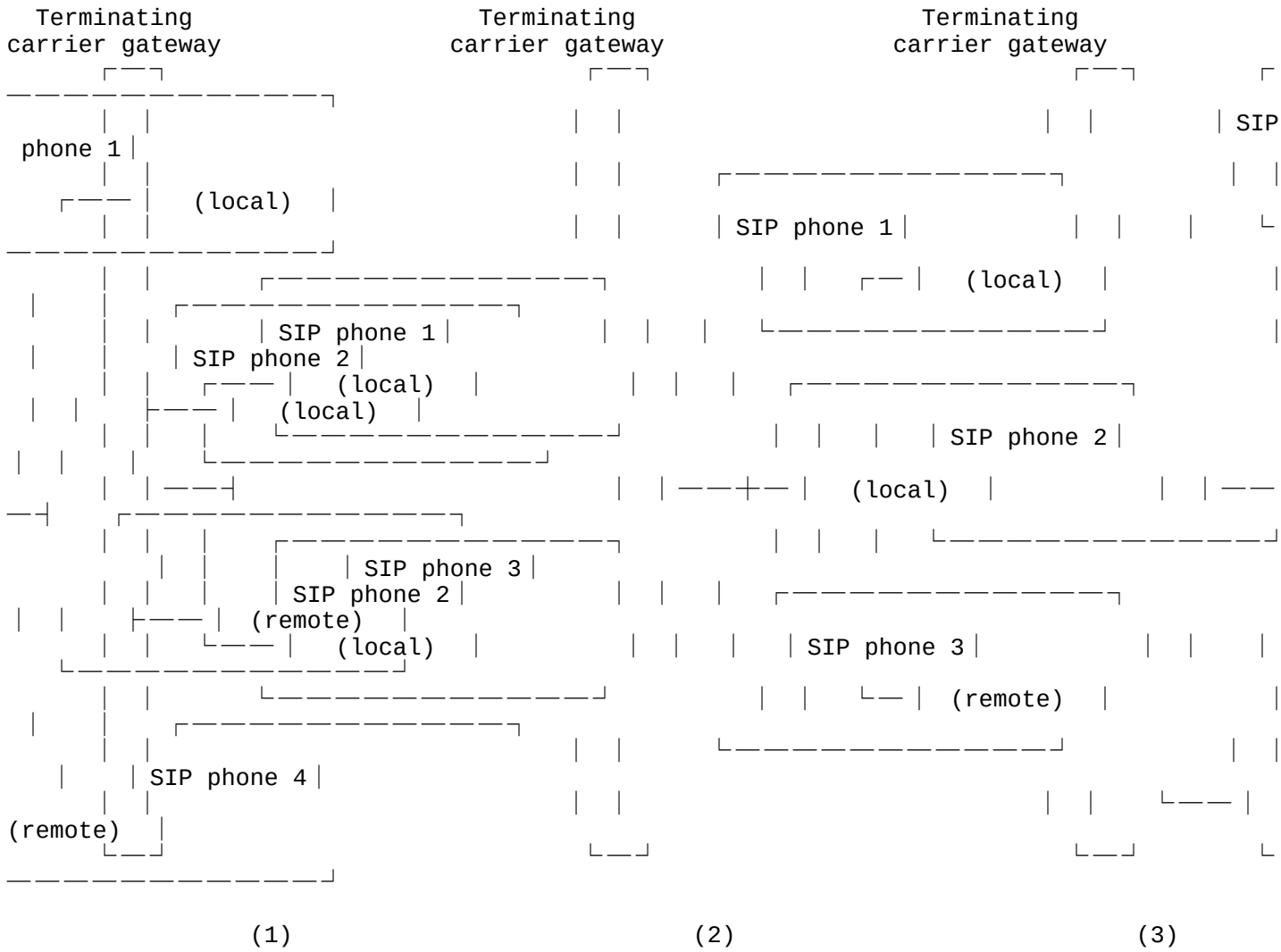

Figure 4: SIP forking: 1) multiple local numbers; 2) multiple local numbers and one remote number; 3) multiple local numbers and multiple remote numbers

3.4. Case 2: a legitimate caller uses a modified number they own

In this case, the caller (Alice) modifies the caller ID to another number (Alice2) she owns. CIV distinguishes legitimate and illegitimate spoofing of a phone number based on whether the caller owns that number. Therefore, if Alice owns Alice2's number, she can configure the call forwarding function so that the verification call sent to Alice2 is forwarded to Alice. The rest of the process is the same as in Case 1. Figure 5 summarizes the steps of operations.

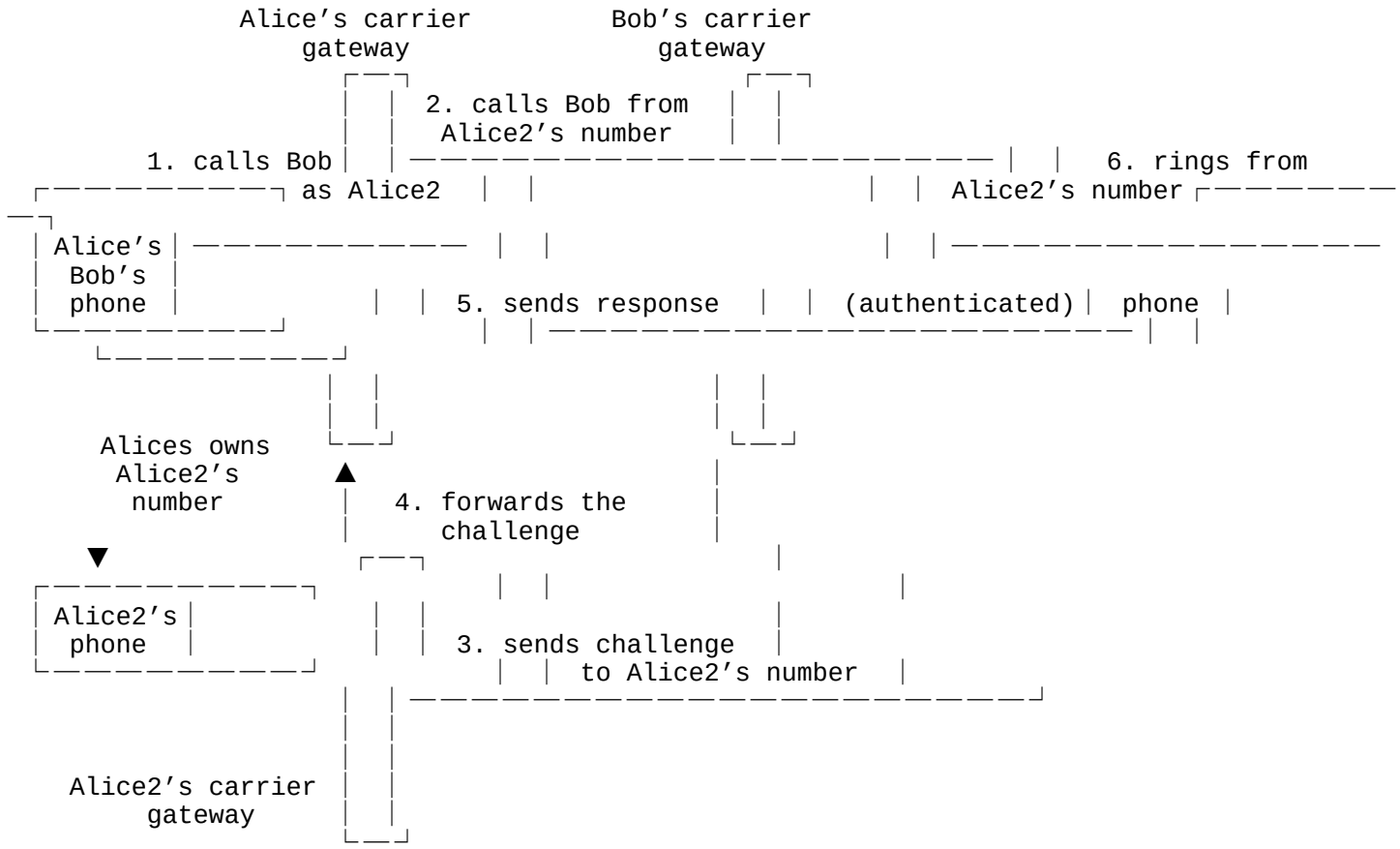

Figure 5: authenticated caller with a legitimately modified number

3.4.1. Private branch exchange

Alice's and Alice2's numbers may belong to the same carrier, e.g., when a private branch exchange (PBX) is used to show a single caller ID for all outgoing calls. In this case, a solution is needed to maintain the state of outgoing calls and forward the received challenge to the correct caller. This can be done by updating the PBX software or connecting PBX to a switch that maintains the states of outgoing calls and matches them with the verification calls.

3.4.2. Distributed call centre

Alice's and Alice2's numbers may belong to two different carriers. In this case, Alice can choose to forward only the CIV verification call (identified by the "civ-veri-call" value in the Call-Info header) sent to Alice2's number. However, in a distributed telephone network, where to forward the call may not always be obvious. The following is an example.

A large enterprise typically has multiple call centres that are distributed over distant geographic locations with limited or no communication between them. Calls from the enterprise commonly present a caller ID that reaches a main receptionist or could be answered by any member of a call centre. A call-back to that number may not go to the same call centre that originated the call. For CIV to work, the verification call will need to be forwarded to the system that keeps the state of outbound call setup attempts. There are two solutions to support legitimate spoofing of the caller ID in this distributed network setting.

1. SIP proxy: it is a common architecture choice to connect outbound SIP servers to a SIP proxy for easier management of calls (e.g., routing, load balancing, logging, monitoring etc). In this case, the SIP proxy is the central system that can maintain the states for all outgoing calls. When an inbound SIP server receives CIV verification calls, it forwards these calls to the SIP proxy, which will match them with the outgoing calls based on the session ID.

2. Encoding method: in case there is no SIP proxy linking up the outbound SIP servers, each outbound SIP server needs to maintain the state of its own outgoing calls. The verification call will need to be routed back to the same server that originated the call. In this case, we can use the first few bits of a session ID (say 10 bits) to encode the index of the outbound SIP server. As an example, using only 10 bits, we can encode the index for over 1,000 outbound SIP servers. Given that a session ID has 128 bits, there is still plenty of entropy remaining to guarantee the uniqueness of the session ID. Once an inbound SIP server receives a CIV verification call, it can determine the index of the outbound SIP server from the first few bits of the session ID and forward the call to that server. The encoding details (including how many bits are used) are part of the internal design of the enterprise call-center system.

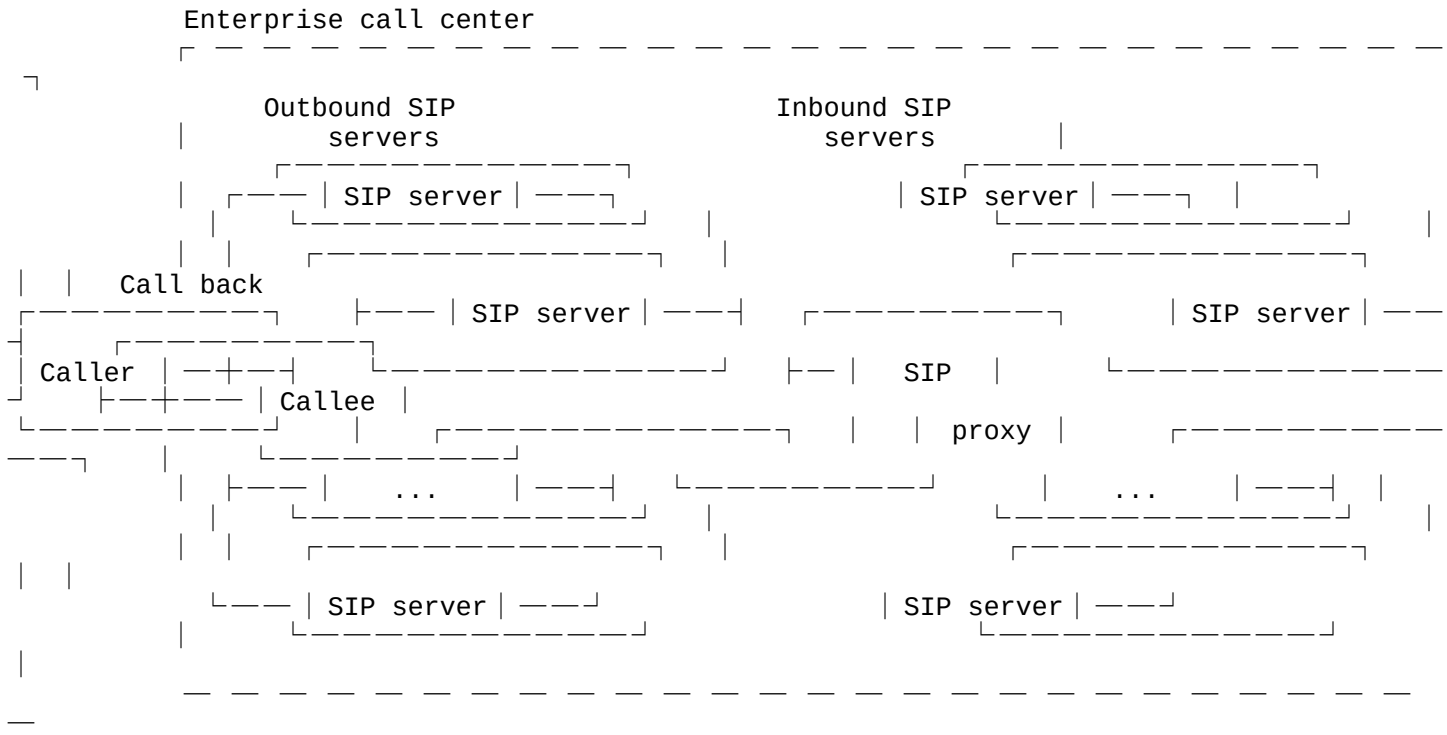

Figure 6: Using a SIP proxy to maintain the CIV call states

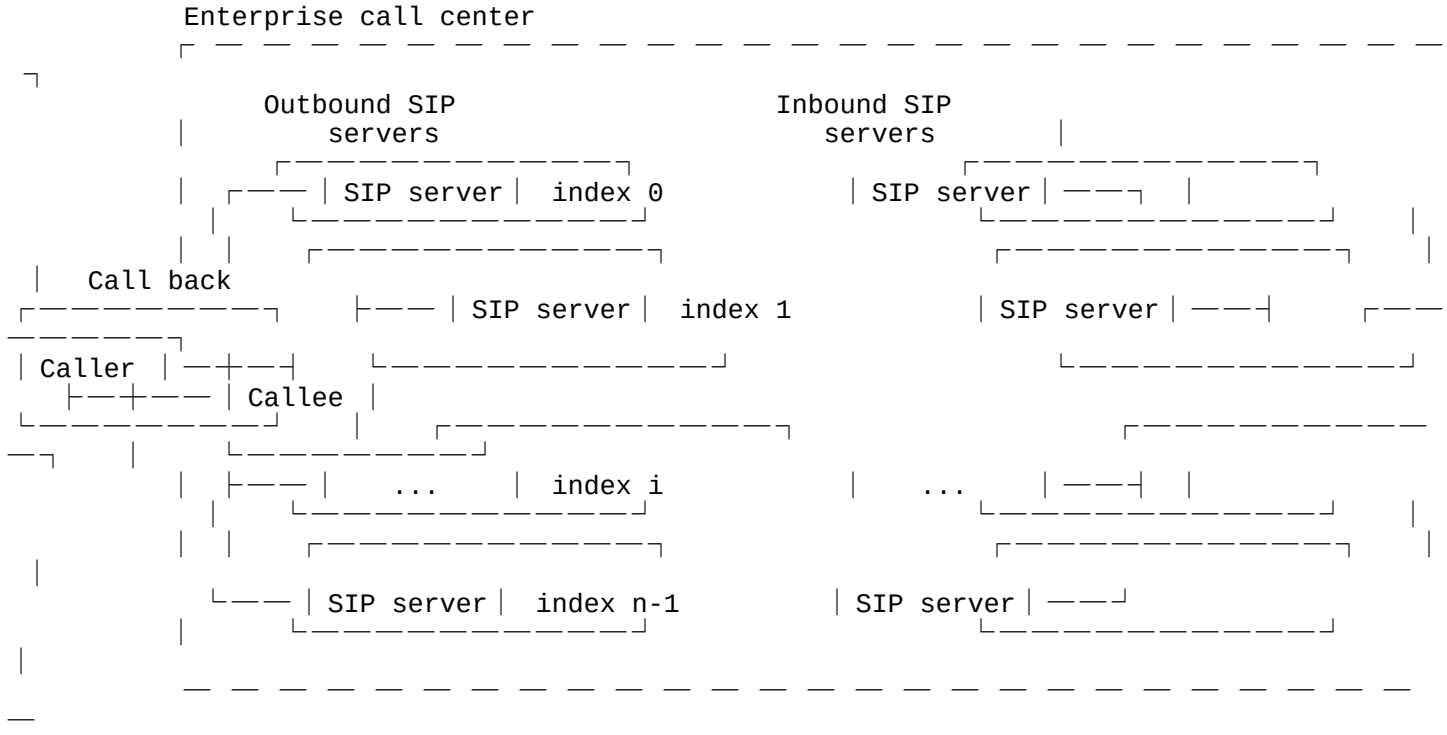

Figure 7: Using encoding to identify the outbound SIP server

3.5. Case 3: an illegitimate caller uses a modified number they don't own

In this case, a spoofing attacker (Eve) modifies the number to one (Alice) that he does not own. Figure 8 summarizes the steps of operations. The first three steps are the same as in Case 1 and 2. In Step 4, Alice's carrier receives the challenge from a quickly terminated verification call. However, it finds no matching record for Alice's outgoing call and no call-forwarding configuration for forwarding the verification call. It quietly discards the challenge. In Step 5, since Bob's carrier doesn't receive a response after a time-out, it concludes that the verification has failed. Depending on the carrier and the user-defined configurations, the terminating carrier may reject the call, connect it to the user's phone with an explicit visual/audio warning about the invalidated caller ID, send the call directly to voicemail, or take other actions.

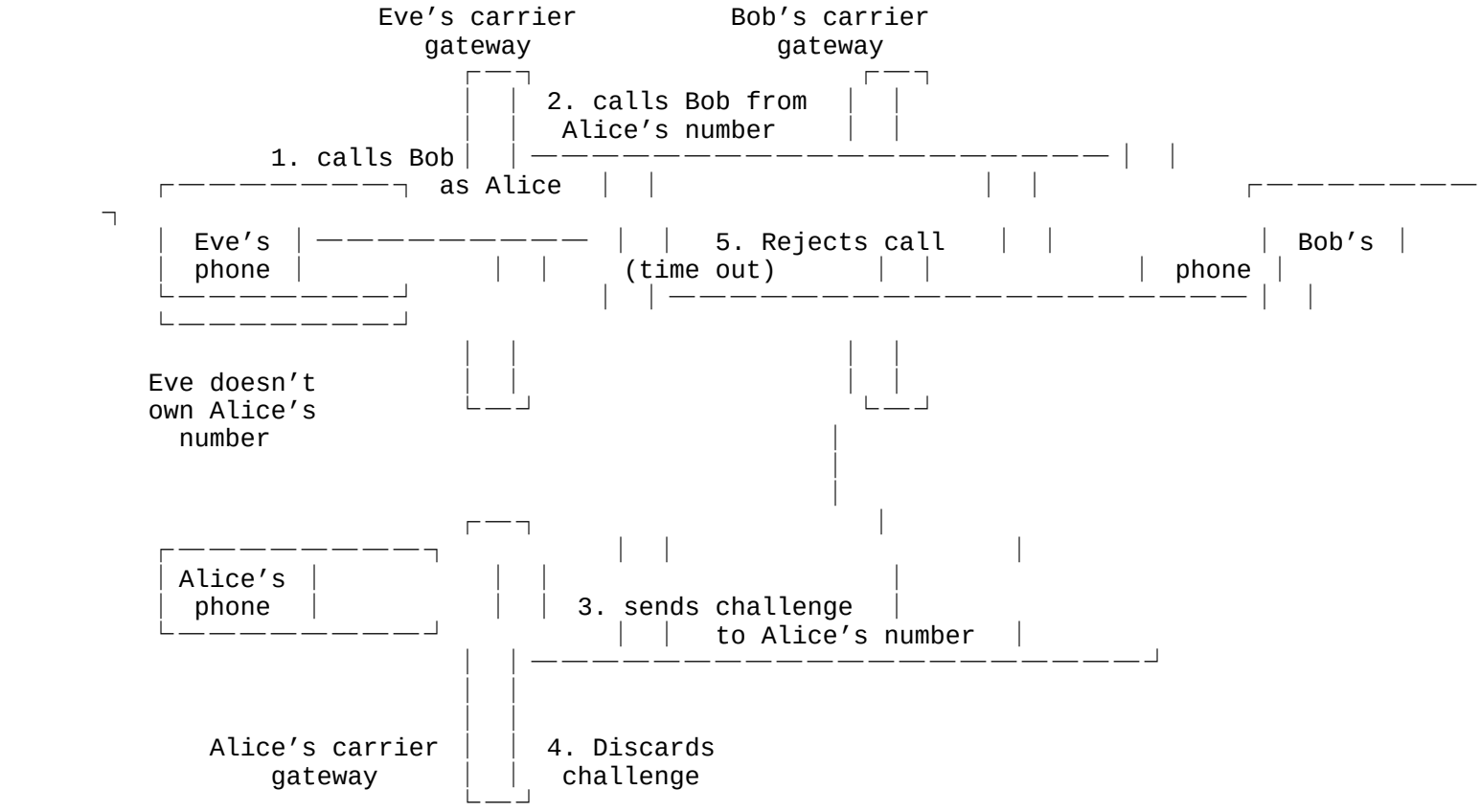

Figure 8: Unauthenticated caller with an illegitimately modified number

4. Compatibility modes

Table 2 summarizes compatibility modes depending on whether the originating or terminating carrier supports CIV.

1. Both carriers support CIV: This is covered in Section 3. After successful authentication, the caller ID with an explicit "verified" status shall be displayed on the user's phone. Otherwise, the call may be rejected at the gateway or connected to the user's phone with an explicit visual/audio warning.

2. Only the terminating carrier supports CIV: The (unverified) caller ID is displayed on the user's phone. The "unverified" status is explicit. The terminating carrier should explicitly communicate this status to the user, e.g., through visual or audio alerts.

3. Only the originating carrier supports CIV: The (unverified) caller ID is displayed on the user's phone. The "unverified" status is implicit.

4. Neither party supports CIV: The (unverified) caller ID is displayed on the user's phone. The "unverified" status is implicit.

	Originating carrier	Terminating carrier	Presentation to the called user
1	Supports CIV	Supports CIV	Caller ID with explicit "verified" status
2	Doesn't support CIV	Supports CIV	Caller ID with explicit "unverified" status
3	Supports CIV	Doesn't support CIV	Caller ID with implicit "unverified" status
4	Doesn't support CIV	Doesn't support CIV	Caller ID with implicit "unverified" status

Table 2: Summary of modes of operations for CIV

5. CIV exemption

In a normal CIV operation, the terminating carrier performs the challenge-response authentication process when the initial call contains a "civ" tag that indicates support for CIV. However, the terminating carrier may define a security policy to exempt the CIV check and immediately connect the call to the user. The following are possible scenarios.

* The called party is an emergency service (e.g., 999). We recommend that all calls to an emergency service should be exempt from the CIV checks, even if the calling party supports CIV. This is to minimize the call setup latency. Furthermore, making spoofed calls to emergency services is not the modus operandi for most scammers, and these calls can be traced by law enforcement if required.

* Owners of some telephone numbers (e.g., a restaurant reservation line) may choose to accept the risk of receiving calls with spoofed numbers and use other measures to manage that risk (e.g., training staff not to trust the caller ID display at all). In that case, they can define a personalized security policy with their carrier to allow all calls to go through without performing any CIV check of the caller ID.

6. Working with non-IP networks

In this document, we assume that both the originating and the terminating carriers support SIP and use standard SIP trunking for connecting with other phone networks. Typically, a SIP call traverses IP networks. However, sometimes the call path may contain a non-IP segment (e.g., PSTN, SS7), which allows limited data transmission. CIV is designed to carry only the minimum amount of data required for caller ID authentication. The data involved in each of the three CIV steps is explained below.

Step 1: The initial call contains the following data: 1) a binary flag that indicates the support for "civ" (1 bit); 2) a 128-bit session ID. The total size is 129 bits (less than 17 bytes).

Step 2: The verification call contains the following data: 1) a binary flag that indicates it is a CIV verification call (1 bit); 2) two 128-bit session IDs. The total size is 257 bits (less than 33 bytes). The challenge is embedded as part of the caller ID, which is universally supported by both IP and non-IP networks. The conversion of the caller ID between IP and non-IP networks (i.e., between INVITE in SIP and IAM in SS7) is done automatically.

Step 3: The originating carrier sends a 4-digit response through DTMF, which is universally supported by both IP and non-IP networks. The conversion of DTMF between IP and non-IP networks is done automatically.

To make CIV work reliably across heterogeneous networks, we should ensure the CIV data is not lost along the call path. When traversing a non-IP segment, we recommend using the User-to-User Information (UUI) parameter in SS7 to carry the CIV data during the IP to non-IP network conversion, and restore the same CIV data during the non-IP to IP network conversion. Normally, the data size limit of UUI is 128 bytes, as specified in ITU-T Q.763 (or 131 bytes in ANSI) in the payload. This is more than sufficient to store the small amount of CIV data in the initial call (17 bytes) and in the verification call (33 bytes). Details of the encoding of the CIV data in UUI and the conversion between IP and non-IP networks are outside the scope of this document.

In the rare case when the originating or terminating carrier does not use SIP trunking and only uses legacy SS7 for connecting with other networks, they can still support CIV at the switch level by carrying the CIV data in the UUI parameter. The CIV protocol works across heterogeneous networks by converting data in the UUI parameter to the SIP INVITE header, and vice versa.

7. Security Considerations

7.1. CIV impacts on tracing

The telephone system only works because each phone call resolves to a single phone number hosted and allocated by a network operator - the range holder for that number. The range holder can identify to whom the number is allocated (which may need to be done via number resellers). It also has the power to disable the number if necessary. Scammers often use spoofed phone numbers to hide their identity and to avoid being tracked down. Widespread adoption of CIV would prevent calls using illegitimately spoofed numbers from getting through. This would force telephone scammers to use phone numbers that are allocated to them as their caller ID. Therefore, they would

be traceable (via the range holder) and their service could be terminated quickly.

7.2. Length of the challenge

A spoofing attacker will not receive the challenge but may try to guess its value. For a challenge of n random digits, the probability of guessing successfully is 1 in 10^n. A larger value of n decreases the probability exponentially, but at the cost of transmitting more digits. We recommend n = 4 as a suitable trade-off, which corresponds to a success rate of 0.01% by random guessing. Consecutively failed guesses can be easily detected by the terminating carrier, and could be dealt with accordingly, e.g., by increasing the delay between calls.

7.3. Downgrade attack

To bypass CIV, a spoofing attacker may use a carrier that does not support CIV or set up their own SIP server that does not include the "civ" tag in the Supported header of the INVITE request. In this downgrade attack, the call will be connected to the user, but the caller ID remains unverified. The terminating carrier can explicitly communicate the "unverified" status to the user, e.g., through visual/audio alerts on the phone, or a pre-recorded warning when the user answers the phone. In some applications (e.g., telephone banking), the system may restrict the caller from performing certain security operations during a call if the caller ID is not verified.

7.4. Reflected DoS

CIV uses a quickly terminated flash call to transmit a challenge. In a normal CIV operation, the verification call is transmitted between the two carriers, and is never shown to users. However, a malicious carrier may include the "civ" tag in the INVITE request and use a spoofed caller ID. One might be concerned that the verification call would reach a target user’s phone as a "missed call" (with a caller ID that may or may not be dialable). The presence of such a "missed call", especially when it makes the phone ring momentarily, can be a nuisance to the user. This could be regarded as a Denial of Service (DoS) attack. Note that the attacker can always make "missed calls" to the user directly, e.g., in Wangiri scams. Unlike a Wangiri call, in CIV, the attacker cannot control the caller ID of the reflected verification call. Hence, the impact of the attack is more limited. Here, instead of attacking the user directly, the attacker uses a carrier to launch a reflected DoS attack.

The reflected DoS attack can be easily prevented as follows. First, if the carrier that receives the verification call supports CIV, it will discard the call by design based on checking it against the state of outgoing calls. Second, if the carrier does not support CIV, it can stop this attack by simply filtering incoming calls containing the "civ-veri-call" value. Third, in 4G and 5G networks, voice services run over IMS (IP Multimedia Subsystems) and that stack uses SIP end-to-end, including on the user equipment. Therefore, the user's device can detect the "civ-veri-call" value and block the verification call (which should not reach the user's phone in the first place), hence providing a further layer of defence on the phone. This solution works with modern mobile phones (4G and onwards) and SIP-based desk phones, although it does not work with legacy GSM/3G and landline phones.

8. IANA Considerations

8.1. "civ" option tag for Supported header

This document defines a new option tag "civ" for the Supported header of an INVITE request in the "Option Tags" registry as below.

Name: civ
Description: This option tag is used by the calling party to indicate support for the Caller ID Verification (CIV) protocol in the Supported header of an INVITE request.

8.2. "civ-veri-call" value for Call-Info header

This document also defines a "civ-veri-call" value for the Purpose parameter in the Call-Info header of an INVITE request as below. This value, paired with the same URI as in the TO header, indicates that the purpose of the current INVITE is to make a verification call.

Header Field: Call-Info
Paramter Name: purpose
Predefined Values: yes

9. References

9.1. Normative References

[1] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <https://www.rfc-editor.org/info/rfc3261>.

[2] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[3] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[4] Jones, P., Salgueiro, G., Pearce, C., and P. Giralt, "End-to-End Session Identification in IP-Based Multimedia Communication Networks", RFC 7989, DOI 10.17487/RFC7989, October 2016, <https://www.rfc-editor.org/info/rfc7989>.

[5] Wang, W., Delavar, M., Azad, M., Nabizadeh, F., Smith, S., and F. Hao, "Spoofing Against Spoofing: Towards Caller ID Verification In Heterogeneous Telecommunication Systems", ACM Transactions on Privacy and Security, DOI 10.1145/3625546, December 2023, <https://doi.org/10.1145/3625546>.

9.2. Informative References

[6] Anderson, R., "Security engineering: a guide to building dependable distributed systems", John Wiley & Sons, 2020.

[7] FCC, "Combating spoofed robocalls with caller ID authentication", <https://www.fcc.gov/call-authentication>.

[8] Ofcom, "Number spoofing scams", January 2023, <https://www.ofcom.org.uk/phones-and-broadband/scam-calls-and-messages/phone-spoof-scam>.

[9] BBC, "Cloned customer voice beats bank security checks", November 2024, <https://www.bbc.co.uk/news/articles/c1lg3ded6j9o>.

[10] Ofcom, "Consultation: Calling Line Identification (CLI) authentication - a potential approach to detecting and blocking spoofed numbers", April 2023, <https://www.ofcom.org.uk/phones-and-broadband/phone-numbers/cli-authentication>.

Acknowledgments

We thank ...

Authors' Addresses

Feng Hao
University of Warwick
Department of Computer Science
Coventry
CV4 7AL
United Kingdom
Email: feng.hao@warwick.ac.uk

Basil Thomas
Squire Technologies Ltd
Prospect House, Sandford Lane
Wareham
BH20 4DY
United Kingdom
Email: BThomas@squire-technologies.com

Steve Smith
trueCall Ltd
2 Old Palace Lane
Richmond
TW9 1PG
United Kingdom
Email: stevesmith@truecall.co.uk

Muhammad Ajmal Azad
Birmingham City University
Department of Computer Science
Birmingham
B5 5JU
United Kingdom
Email: Muhammadajmal.azad@bcu.ac.uk