Design of a Local, GPU-Accelerated Conversational Agent for Defensive Scambaiting Research

Martin Jajarmi, Sukhraj Singh, and Yahya Fazal

Department of Computer Science and Technology, Algoma University

Information Technology Security and Privacy - Group 14 - February 5, 2026
Course instructor: Dr. Yazan Otoum

Abstract- Telephone-based social-engineering scams remain a significant vector for cybercrime, relying on real-time persuasion to extract credentials, payments, or system access from technologically naive victims. Defensive scambaiting, the controlled engagement of scammers to delay operations and collect behavioral data, has emerged as a potential research and mitigation strategy. This project proposes the design and implementation of a fully local, GPU-accelerated conversational agent capable of engaging scam callers in real time while enforcing strict safety constraints. The system integrates automatic speech recognition (ASR), large language model (LLM) inference, dialogue policy enforcement, and text-to-speech (TTS) synthesis into a modular, cross-platform architecture. The proposed solution emphasizes low-latency inference, sandboxed deployment, and extensibility through a statically linked core engine with platform-specific wrappers and a standalone application frontend written in C++20.

Index Terms- conversational agents, defensive scambaiting, local inference, safety policy, social engineering, voice phishing.

I. INTRODUCTION

Telephone fraud and voice-based phishing (vishing) continue to scale globally, exploiting human trust rather than software vulnerabilities. Recent public reporting indicates that reported fraud losses increased substantially in the United States, with billions of dollars in reported losses in 2024 and a notable share attributed to contact methods such as calls and texts [1]. Telecom-side mitigations such as caller ID authentication under the STIR/SHAKEN framework aim to reduce spoofing and improve trust signals at the network layer [2], [3]. However, these mitigations do not fully address human-in-the-loop persuasion once a call is connected.

Scambaiting, the deliberate engagement of scammers to occupy their resources, has historically been performed manually. Human-operated scambaiting can be effective, but it is not easily reproducible or scalable for academic experimentation. This project proposes an automated, local-first conversational agent for controlled, inbound-only research scenarios. The system prioritizes low-latency interaction while enforcing hard safety constraints to prevent disclosure of sensitive information or execution of harmful actions.

II. PROBLEM STATEMENT

How can a real-time, fully local conversational agent be designed to safely engage scam callers for extended periods while preventing harmful actions and maintaining low-latency interaction? The proposed solution must integrate speech processing and LLM inference, enforce safety constraints in code, and remain portable across platforms.

III. CASE STUDY CONTEXT

This work was conducted as a course project for Information Technology Security and Privacy, with an emphasis on defensive techniques against social-engineering attacks. The agent is treated as a research tool rather than an operational, production-ready mitigation system. Evaluation is restricted to controlled inbound calls and synthetic test callers, with appropriate legal practices for any recording or data retention.

IV. REVIEW OF RELATED WORK

A. Vishing and Social Engineering

Vishing has been studied as a form of social engineering in which attackers apply persuasion and compliance techniques during live voice interactions. Jones et al. analyzed how persuasion principles are applied during vishing attacks, providing insight into the conversational tactics that enable escalation and victim compliance [4]. More recent work highlights vishing and voice-based social engineering as an increasing challenge and motivates automated methods for understanding and mitigating spoken deception [5]. Broader surveys of smishing and vishing detection discuss feature engineering and machine learning approaches for identifying suspicious communications [6].

B. Network-Layer and Call-Screening Mitigations

Telecom-focused interventions, including STIR/SHAKEN caller ID authentication, attempt to reduce caller ID spoofing and enable downstream blocking and labeling strategies [2], [3]. Scammers can still adapt through legitimate numbers, call forwarding, or social pretexting, leaving a gap for defenses at the conversational layer. Applied proposals have used conversational AI to analyze call content and interaction patterns [7], while recent research has examined LLM-based phone-scam detection through conversational dynamics [8].

C. LLM-Driven Agents and Dialogue Control

General-purpose LLMs can generate fluent dialogue, but robust interactive systems require an explicit control layer for objectives and safety. Agent paradigms such as ReAct interleave reasoning and action selection, improving interpretability by structuring how models select the next action [9]. A scambaiting or call-screening system can similarly separate policy decisions, such as stall, clarify, or refuse, from surface realization and enforce hard constraints outside the model.

D. Speech Processing Components for Local Agents

Modern local ASR has been accelerated by large-scale weakly supervised training. The Whisper model family demonstrates robust speech recognition across many conditions and languages [10], while whisper.cpp provides a C/C++ implementation for local inference [11]. Voice activity detection is used to segment speech and avoid unnecessary computation; practical WebRTC-based VAD implementations are common in voice applications [12].

E. Local Inference Runtimes and Cross-Platform I/O

Local runtimes such as llama.cpp target efficient LLM inference in C/C++ across varied hardware [13]. LlamaLib provides a higher-level wrapper for llama.cpp [19]. RtAudio offers a cross-platform C++ interface for real-time audio streaming [14]. Vulkan supplies a vendor-neutral API for accelerated computation [15], [16]. Platform differences in file I/O, multithreading, and input handling can be handled at compile time for POSIX and Windows APIs. GLFW and Dear ImGui provide cross-platform windowing and immediate-mode interface support with Vulkan backends [17], [18].

V. PROPOSED SYSTEM ARCHITECTURE

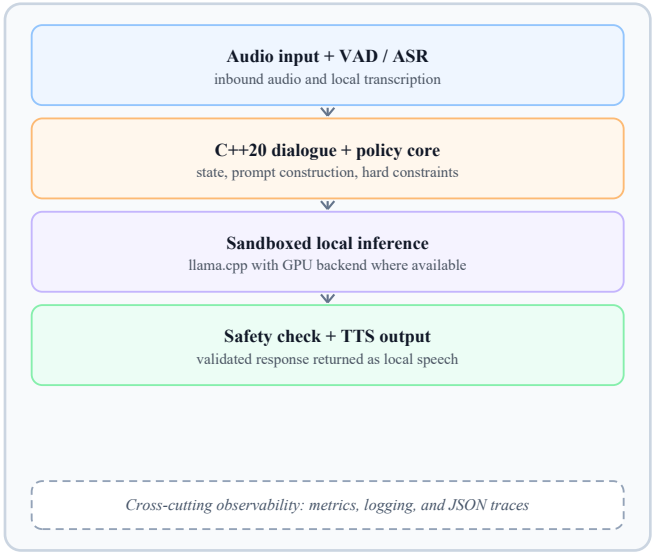


Fig. 1. Proposed local-first SCAMPA architecture with an explicit policy and sandbox boundary.

The system is organized around a platform-agnostic engine core implemented as a static library. The core contains dialogue state management, policy enforcement, prompt construction, inference orchestration, and metrics logging. Platform-specific dependencies, including audio I/O, GPU backends, and speech synthesis, are accessed through narrow adapter interfaces.

A standalone console application links against the core and exposes a frontend through the engine's layer system. This separation supports runtime model configuration, replacement of ASR or TTS components, and cross-platform portability without moving policy enforcement into the model. Future work may examine direct VoIP integration; the initial evaluation instead uses external calling software and virtual audio routing in a controlled environment [20], [21].

VI. PROPOSED EVALUATION METRICS

Future phases will measure end-to-end latency from utterance completion to TTS start, ASR accuracy, LLM response time, safety-policy trigger rates, and sustained engagement time under controlled scenarios. Custom instrumentation in the core library will emit JSON traces for review with Chrome's Trace Event Format tooling, supporting bottleneck analysis and validation of real-time performance [22].

VII. CONCLUSION AND PHASE 2 PLAN

This Phase 1 report defines the cybersecurity motivation for automated defensive scambaiting, reviews related work in vishing, call screening, LLM agents, and speech processing, and proposes a modular, cross-platform architecture for a local voice agent. Phase 2 will implement the engine core, integrate local ASR and LLM inference with Vulkan acceleration where available, and validate safety enforcement and latency targets in controlled experiments.

TABLE I
PHASE 1 SCOPE AND VALIDATION BOUNDARY

Dimension	Phase 1 position
Research use	Controlled, inbound-only tests with synthetic callers.
System status	Architecture and interfaces defined; engine implementation planned for Phase 2.
Safety	Policy decisions and hard constraints remain outside unconstrained generation.
Evaluation	Latency, ASR accuracy, model response time, policy triggers, and engagement duration.

REFERENCES

[1] Federal Trade Commission, "Consumer Sentinel Network Data Book 2024," 2025. [Online]. Available: https://www.ftc.gov/system/files/ftc_gov/pdf/csn-annual-data-book-2024.pdf

[2] Federal Communications Commission, "Combating Spoofed Robocalls with Caller ID Authentication (STIR/SHAKEN)," 2020-2025. [Online]. Available: https://www.fcc.gov/call-authentication

[3] Federal Communications Commission, "Call Authentication Trust Anchor," Federal Register, Aug. 19, 2025. [Online]. Available: https://www.federalregister.gov/documents/2025/08/19/2025-15809/call-authentication-trust-anchor

[4] K. S. Jones, M. E. Armstrong, M. K. Tornblad, and A. S. Namin, "How social engineers use persuasion principles during vishing attacks," 2020. [Online]. Available: https://par.nsf.gov/servlets/purl/10281805

[5] A. Triantafyllopoulos et al., "Vishing: Detecting social engineering in spoken communication," 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0885230825000270

[6] A. E. Chichwadia et al., "Detecting Smishing and Vishing Attacks using Machine Learning," 2024. [Online]. Available: https://infonomics-society.org/wp-content/uploads/Detecting-Smishing-and-Vishing-Attacks-using-Machine-Learning.pdf

[7] K. M. Rao, "Suspicious Call Detection and Mitigation Using Conversational AI," 2023. [Online]. Available: https://www.tdcommons.org/cgi/viewcontent.cgi?article=7596&context=dpubs_series

[8] "Combating Phone Scams with LLM-based Detection," arXiv, 2024. [Online]. Available: https://arxiv.org/html/2409.11643v2

[9] S. Yao et al., "ReAct: Synergizing Reasoning and Acting in Language Models," arXiv:2210.03629, 2022. [Online]. Available: https://arxiv.org/abs/2210.03629

[10] A. Radford et al., "Robust Speech Recognition via Large-Scale Weak Supervision," in Proc. ICML, 2023. [Online]. Available: https://proceedings.mlr.press/v202/radford23a.html

[11] ggml-org, "whisper.cpp," GitHub repository, accessed 2026. [Online]. Available: https://github.com/ggml-org/whisper.cpp

[12] serenadeai, "webrtcvad," GitHub repository documentation, accessed 2026. [Online]. Available: https://github.com/serenadeai/webrtcvad/blob/master/README.md

[13] ggml-org, "llama.cpp: LLM inference in C/C++," GitHub repository, accessed 2026. [Online]. Available: https://github.com/ggml-org/llama.cpp

[14] G. P. Scavone, "RtAudio: A Cross-Platform C++ Class for Realtime Audio Input/Output," in Proc. ICMC, 2002. [Online]. Available: https://caml.music.mcgill.ca/~gary/papers/rtaudio2002.pdf

[15] Khronos Group, "Khronos Vulkan Registry," accessed 2026. [Online]. Available: https://registry.khronos.org/vulkan/

[16] Khronos Group, "Vulkan API Specification," accessed 2026. [Online]. Available: https://registry.khronos.org/vulkan/specs/latest/html/vkspec.html

[17] M. Magnusson and C. Lovdahl, "GLFW - An OpenGL and Vulkan Windowing Library," accessed 2026. [Online]. Available: https://www.glfw.org

[18] O. Cornut, "Dear ImGui: Immediate Mode GUI," GitHub repository, accessed 2026. [Online]. Available: https://github.com/ocornut/imgui

[19] undreamai, "LlamaLib: High-Level LLM Runtime for C++ and C#," GitHub repository, accessed 2026. [Online]. Available: https://github.com/undreamai/LlamaLib

[20] E. Muzychenko, "Virtual Audio Cable," accessed 2026. [Online]. Available: https://vac.muzychenko.net

[21] Google LLC, "Google Voice," accessed 2026. [Online]. Available: https://voice.google.com

[22] Google, "Trace Event Format," Chromium Project Documentation, accessed 2026. [Online]. Available: https://chromium.googlesource.com/catapult/+/HEAD/tracing/README.md