DEMO

First chapter only

The Dual-Protocol Integration Guide

Google AP2 + Stripe ACP in a Single Codebase

pragma.vision

SOFT.HOUSE

The Dual-Protocol Integration Guide

Trademark Notice

Google, Google Pay, Google Cloud, and Android are trademarks of Google LLC. Stripe is a trademark of Stripe, Inc. Cloudflare and Cloudflare Workers are trademarks of Cloudflare, Inc. Supabase is a trademark of Supabase, Inc. OpenAI and ChatGPT are trademarks of OpenAI, Inc. Claude is a trademark of Anthropic, PBC. W3C is a trademark of the World Wide Web Consortium. Visa is a trademark of Visa International Service Association. OWASP is a trademark of the OWASP Foundation. Midjourney is a trademark of Midjourney, Inc. Canva is a trademark of Canva Pty Ltd. Etsy is a trademark of Etsy, Inc. Amazon is a trademark of Amazon.com, Inc. All other trademarks are the property of their respective owners.

No Affiliation

This book is an independent publication. It is not authorized, sponsored, or endorsed by any of the companies or organizations whose products or services are mentioned herein.

No Professional Advice

The information in this book is provided for educational purposes only. It does not constitute legal, financial, investment, tax, or other professional advice. Readers should consult qualified professionals for guidance specific to their situation.

Code Examples

Code examples in this book are provided for illustration only. They may not be suitable for production use without additional validation, error handling, and security review.

Published by Pragma Vision LLC

First edition, 2026.

Contents

1

Introduction: Why Two Protocols?

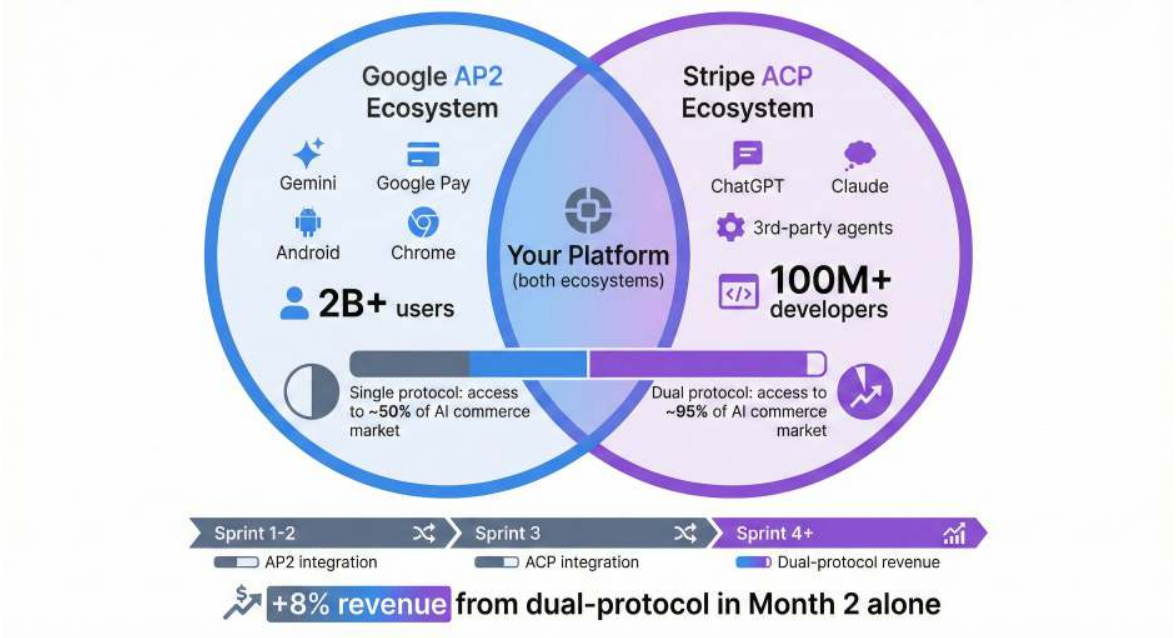

Figure 1. Supporting both Google AP2 and Stripe ACP lifts reach from about 50% to about 95% of the AI commerce market: AP2 spans Gemini, Google Pay, Android, and Chrome (2B+ users), ACP adds ChatGPT, Claude, and 100M+ developers, for +8% revenue by Month 2

AI-powered commerce has split the payments landscape into competing ecosystems. Google released the Agent Payments Protocol (AP2) under Apache 2.0, assembling more than sixty partners—Mastercard, Adyen, PayPal, Coinbase—around a cryptographic mandate model. Stripe and OpenAI countered with the Agentic Commerce Protocol (ACP), an open standard for programmatic commerce flows between AI agents and businesses. Each protocol addresses a different question in the transaction lifecycle.

Key Insight

AP2 and ACP are not competitors—they are complementary layers. AP2 answers "Did the user authorize this spend?" while ACP answers "Process the actual payment." A production system needs both.

1.1 The Three-Layer Model

Agentic commerce requires three distinct protocol layers. Collapsing any layer into another removes critical functionality:

Layer	Protocol	Question Answered
Identity	Visa TAP	Is this agent legitimate?
Authorization	Google AP2	Did the user approve this spend?
Execution	Stripe ACP	Process the transaction

Each layer solves a distinct problem in the agentic commerce stack.

1.2 The Cost of Single-Protocol Lock-In

Choosing only AP2 locks you into the Google ecosystem and excludes ChatGPT and Claude users. Choosing only ACP excludes Google Assistant users and sacrifices the autonomous mandate model that lets agents spend without per-transaction approval.

The AI assistant market is fragmenting, not consolidating—and each assistant brings its own payment protocol.

80%

of future commerce interactions projected to involve AI assistants by 2030[1]

1.3 The Dual-Protocol Thesis

This book demonstrates with production code that you can support *both* protocols in a single codebase with over 60% shared infrastructure. AP2 and ACP share the same core operations: nonce validation, cryptographic signing, mandate lifecycle management, rate limiting, and audit logging. The protocol-specific logic—ECDSA signatures for AP2, HMAC webhook verification for ACP—accounts for less than 40% of total code. The result is 1,180 fewer lines to write, test, and maintain.

1.4 About This Book

Every code example in this book comes from a measured dual-protocol architecture implemented for Cloudflare Workers. The architecture is designed to power a growing family of interconnected platforms—wish.now, phantoid.com, great.gift, and others—through a shared infrastructure layer. Where the text says "we achieved 69.5% code sharing," that number comes from a measured line count, not an estimate.

[1]Gartner, "Predicts 2025: AI Agents Will Transform Digital Commerce," 2024; Bain & Company estimates similar figures.

1.5 Who This Book Is For

- **Backend engineers** implementing AI-native payment flows
- **Architects** designing multi-protocol payment systems
- **CTOs** evaluating whether to commit to one protocol or both
- **Security engineers** reviewing cryptographic payment architectures
- **Developer relations teams** at payment companies seeking integration patterns

1.6 What You Will Build

By the end of this book, you will have a complete dual-protocol payment system with:

1. A unified mandate database storing AP2 and ACP data in shared tables
2. Protocol detection middleware that auto-routes requests
3. A shared cryptographic service supporting ECDSA, HMAC, and ML-DSA-65
4. Idempotency key infrastructure preventing duplicate charges
5. A comprehensive E2E test suite validating both protocols

DEMO

This is a free preview of the full edition.

Get the complete book at:

https://shop.pragma.vision