



DEMO

First chapter only

The Dual-Protocol Integration Guide

Google AP2 + Stripe ACP in a Single Codebase

The Dual-Protocol Integration Guide

© 2026 Pragma Vision LLC. All rights reserved.

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?	6
1.1 The Three-Layer Model	7
1.2 The Cost of Single-Protocol Lock-In	7
1.3 The Dual-Protocol Thesis	8
1.4 About This Book	8
1.5 Who This Book Is For	9
1.6 What You Will Build	9
2 Google AP2 Deep Dive	10
2.1 The Mandate Chain	11
2.1.1 Intent Mandate	11
2.1.2 Cart Mandate	12
2.1.3 Payment Mandate	13
2.2 ECDSA Signature Verification	13
2.3 The Autonomous Flow	15
2.4 AP2 Ecosystem	15
3 Stripe ACP Deep Dive	17
3.1 ACP Specification Versions	17
3.2 Checkout Session Lifecycle	18
3.3 The Four ACP Endpoints	19
3.4 SharedPaymentToken (Payment Delegation)	20
3.5 Bearer Token Authentication	20
3.6 Stripe Webhook HMAC Verification	21

3.7	ACP vs AP2: A Comparison	22
4	The Shared Infrastructure Layer	23
4.1	Architecture Overview	24
4.2	The Unified Mandate Interface	25
4.3	The Unified Mandate Service	26
4.4	Shared Services Inventory	28
4.5	Directory Structure	28
5	Protocol Detection Middleware	30
5.1	Detection Priority Chain	30
5.2	Context-Based Auto-Detection	31
5.3	The Protocol Router	33
5.4	AI Assistant Detection	35
6	Implementing the Unified Mandate Database	37
6.1	Design Principles	38
6.2	The Intent Mandates Table	39
6.3	JSONB Column Structures	41
6.4	Cart and Payment Mandate Tables	41
6.5	Row-Level Security	42
6.6	Cross-Protocol Queries	43
7	Cryptographic Services: Classical + Quantum-Safe	45
7.1	Why Hybrid Signatures?	46
7.2	The CryptoService	47
7.3	Hybrid Verification Flow	50
7.4	Constant-Time Comparison	51
7.5	Timestamp Validation	52
8	Idempotency and Financial Safety	53
8.1	The Idempotency Key Pattern	53
8.2	Integration with ACP Checkout Completion	55

8.3	AP2 Payment Mandate Idempotency	57
8.4	Financial Safety Checklist	59
9	Testing Both Protocols	60
9.1	Test Architecture	61
9.2	Testing the Shared Layer	61
9.3	AP2 E2E Tests	63
9.4	ACP E2E Tests	65
9.5	Cross-Protocol Test	67
10	The 69.5% Code Sharing Achievement	69
10.1	Code Sharing Metrics	70
10.2	What 69.5% Sharing Means	71
10.3	Maintenance Multiplier	71
10.4	Future-Proofing: Adding a Third Protocol	72
10.5	Architecture Principles Summary	73
10.6	From Dual to Ecosystem	73
	What's Next	75
	About Pragma.Vision	77

1

Introduction: Why Two Protocols?

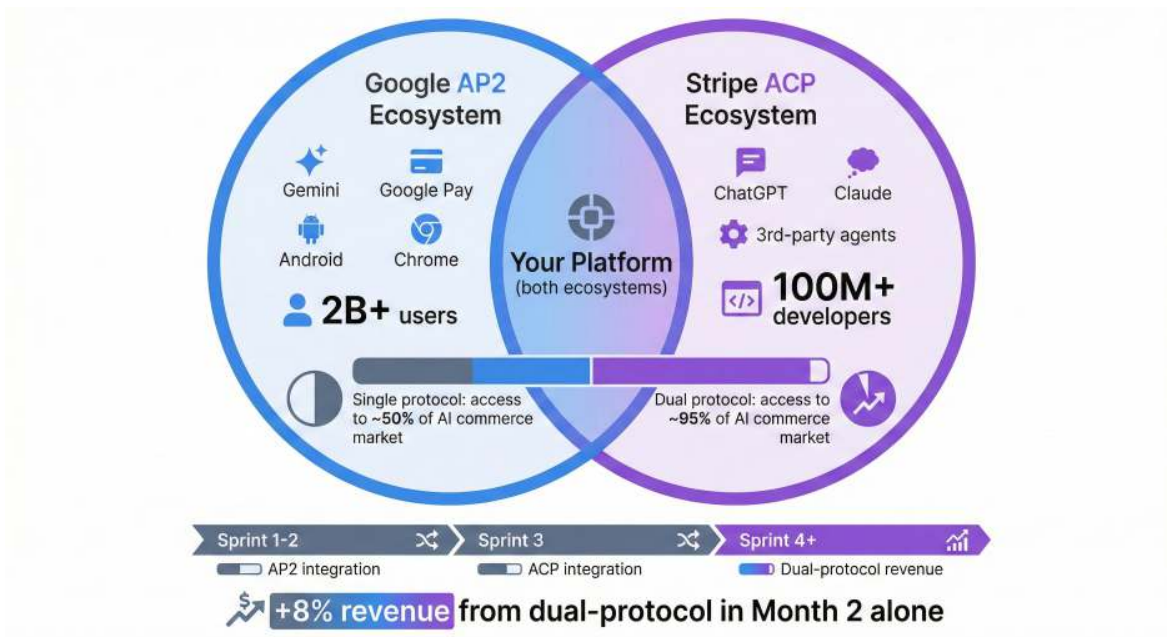


Figure 1. Supporting both Google AP2 and Stripe ACP lifts reach from about 50% to about 95% of the AI commerce market: AP2 spans Gemini, Google Pay, Android, and Chrome (2B+ users), ACP adds ChatGPT, Claude, and 100M+ developers, for +8% revenue by Month 2

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 2030¹

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.

¹Gartner, “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

Get the complete book — <https://shop.pragma.vision>

DEMO

This is a free preview of the full edition.

Get the complete book at:

<https://shop.pragma.vision>