cTrader Sidecar Integration Guide

(Non-Normative Reference Document)

VeritasChain Protocol (VCP) v1.1

Document Version: 1.0 | January 2026

1. Purpose and Scope

This document provides a non-normative reference guide for integrating VeritasChain Protocol (VCP) cryptographic audit trails with cTrader-based trading workflows using a sidecar architecture. It is intended for technical audiences seeking to understand the integration approach demonstrated in the VCP cTrader RTA reference implementation.

1.1 What This Document Is

- A technical reference describing one possible integration approach
- Documentation of the architecture used in the reference implementation
- A guide for developers exploring VCP integration with cTrader
- Non-normative: does not define requirements or specifications

1.2 What This Document Is NOT

- NOT a product manual or official integration guide
- NOT a certification, endorsement, or approval of any kind
- NOT affiliated with or endorsed by Spotware Systems (cTrader vendor)
- NOT a guarantee of regulatory compliance

Important: This reference implementation demonstrates technical feasibility only. Production deployments require independent security review, legal assessment, and compliance verification appropriate to the deployment context.

1.3 Target Audience

This guide is intended for software developers, system architects, and technical evaluators with familiarity in trading platform APIs, cryptographic concepts (hash functions, Merkle trees, digital signatures), and distributed systems architecture.

2. Architecture Overview

The VCP integration with cTrader employs a sidecar architecture pattern, where cryptographic audit trail generation runs as an independent process alongside the trading platform without modifying the platform itself.

2.1 Key Design Principles

Principle	Description
Non-Invasive	No modifications to cTrader platform or broker infrastructure required
Asynchronous	Cryptographic processing does not block or delay trading operations
API-Based	Uses only official, documented cTrader APIs (Open API, FIX)
Platform-Agnostic	VCP processing is decoupled from platform-specific logic
Verifiable	Generated audit trails can be independently verified without system access

2.2 Component Architecture

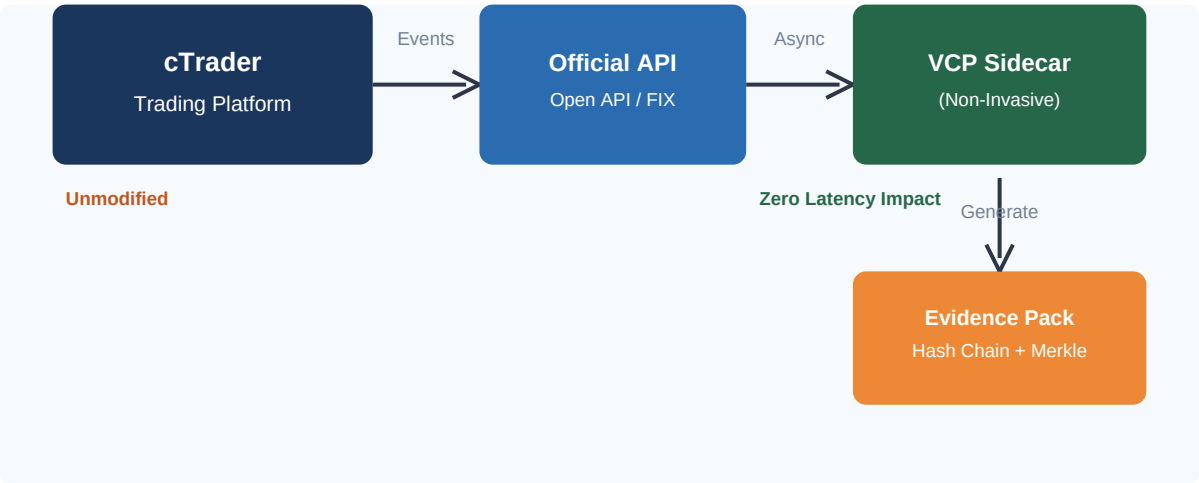

Figure 1: VCP Sidecar Architecture for cTrader Integration

2.3 Component Responsibilities

Component	Responsibility	Modification Required
cTrader Platform	Order execution, market data, position management	None
Official API	Event streaming, order status, execution reports	None
VCP Sidecar	Event capture, hash chain, Merkle tree, anchoring	N/A (new component)
Evidence Pack	Persistent storage of verifiable audit trail	N/A (output)

The sidecar pattern ensures that trading latency is unaffected by cryptographic processing. Events are captured via API callbacks and queued for asynchronous processing, allowing trading operations to proceed at full platform speed.

3. Event Capture Points

This section describes which API layers are used for event capture and which trading events are included in the audit trail.

3.1 API Layer Selection

API	Use Case	Event Types	Latency
cTrader Open API	Primary: Real-time events	Orders, Executions, Positions	~10-50ms
cTrader FIX API	Alternative: FIX integration	ExecutionReport, OrderStatus	~5-20ms
Reporting API	Supplementary: Historical	Trade history, Account data	Batch

The reference implementation primarily uses the **cTrader Open API** for real-time event capture due to its comprehensive event coverage and Protocol Buffers serialization. FIX API may be used as an alternative where FIX infrastructure already exists.

3.2 Captured Event Types

VCP Event	cTrader Source	Description
INIT	OnStart callback	System/algorithm initialization
SIG	Custom signal logic	Trading signal generation
ORD	ExecuteMarketOrder	Order submission
ACK	OnPendingOrders	Order acknowledgment
EXE	OnPositionsOpened	Full order execution
PRT	OnPositionsModified	Partial fill
CLS	OnPositionsClosed	Position close
MOD	ModifyPosition	SL/TP modification
CXL	CancelPendingOrder	Order cancellation
REJ	OnError (Order)	Order rejection
ERR_*	OnError (System)	System/connection errors

3.3 Events NOT Captured

The following events are explicitly **out of scope** for this reference implementation:

- Internal risk management calculations (platform-internal)
- Broker-side liquidity aggregation (not exposed via API)
- Market data ticks (high volume, not order-related)
- Account balance changes unrelated to trading
- Platform UI interactions (no API visibility)

Note: The scope of captured events can be extended based on deployment requirements and available API access.

4. Cryptographic Processing Boundary

All cryptographic operations occur **outside** the cTrader platform boundary, ensuring that trading performance is unaffected and no platform modifications are required.

4.1 Processing Location

Operation	Location	Impact on Trading
Event capture	API callback (cTrader)	None - passive listener
Event queuing	VCP Sidecar (memory)	None - async buffer
SHA-256 hashing	VCP Sidecar (CPU)	None - separate process
Hash chain linking	VCP Sidecar (CPU)	None - separate process
Ed25519 signing	VCP Sidecar (CPU)	None - separate process
Merkle tree build	VCP Sidecar (CPU)	None - batch operation
External anchoring	VCP Sidecar (Network)	None - async I/O
Evidence storage	VCP Sidecar (Disk)	None - async write

4.2 Why Trading Latency Is Unaffected

The sidecar architecture provides complete isolation between trading operations and cryptographic processing:

- **Async event queue:** Events are captured and immediately queued; control returns to trading logic
- **Separate process:** VCP processing runs in its own thread/process space
- **Non-blocking I/O:** File writes and network anchoring use async patterns
- **Batch processing:** Merkle trees are built periodically, not per-event
- **No platform hooks:** No code injection or platform modification required

4.3 Sidecar Benefits

Benefit	Description
Zero trading latency impact	Cryptographic ops never block order flow
Platform independence	Works with any cTrader broker without modification
Easy deployment	No broker cooperation or platform access required
Upgradeable	VCP version can be updated without touching trading logic
Auditable separation	Clear boundary between trading and audit functions

5. Reproducibility and Constraints

This section documents the conditions, limitations, and broker-dependent factors that affect deployment of this integration approach.

5.1 Demo vs. Live Environment

Aspect	Demo Account	Live Account
API Access	Full (Open API, FIX)	Full (Open API, FIX)
Event capture	Identical behavior	Identical behavior
Order execution	Simulated fills	Real market execution
Timestamps	Platform clock	Platform clock
Slippage/Requotes	None (simulated)	Real market conditions
VCP Hash validity	Fully valid	Fully valid

Note: VCP audit trails generated in demo environments are cryptographically identical in structure to those from live environments. The difference lies only in the underlying market execution, not the audit trail integrity.

5.2 Timestamp Precision

Source	Precision	Sync Method
cTrader server time	Milliseconds	Broker NTP sync
Local system clock	Microseconds	OS-level NTP
VCP event timestamp	Microseconds (stored)	Best-effort sync
VCP specification	MILLISECOND (Silver)	Platform clock

VCP Silver Tier specifies BEST_EFFORT clock synchronization with MILLISECOND precision. Higher tiers (Gold, Platinum) require stricter time synchronization methods.

5.3 API Limitations

- **Rate limits:** cTrader Open API enforces connection and request rate limits (~10 orders/sec)
- **Event ordering:** API events may arrive out-of-order under high load; VCP uses timestamps
- **Reconnection:** Network disconnects require reconnection handling; gaps should be logged
- **Historical data:** Real-time events only; historical backfill requires Reporting API

5.4 Broker-Dependent Factors

Factor	Dependency	Mitigation
API availability	Broker must enable Open API	Verify before deployment
FIX access	Some brokers restrict FIX	Use Open API as primary
Server location	Affects network latency	Deploy sidecar near broker
Account types	Some limit API access	Verify account capabilities
Execution model	STP/ECN vs. Market Maker	Document in audit metadata

6. Summary

This guide has described the non-invasive sidecar approach used by the VCP cTrader reference implementation to generate cryptographically verifiable audit trails for cTrader trading workflows.

Key Takeaways:

- VCP integration requires **no modification** to cTrader or broker infrastructure
- All cryptographic processing occurs **outside** the trading platform boundary
- Trading latency is **unaffected** due to async sidecar architecture
- Event capture uses **official APIs** (Open API, FIX) only
- Generated audit trails are **independently verifiable** without system access
- Demo and live environments produce **structurally identical** audit trails

References

Reference	Location
VCP Specification v1.1	https://github.com/veritaschain/vcp-specification
VCP cTrader Reference Implementation	https://github.com/veritaschain/vcp-ctrader-rta-reference
cTrader Open API Documentation	https://help.ctrader.com/open-api/
cTrader FIX API Documentation	https://help.ctrader.com/fix/
RFC 6962 (Certificate Transparency)	https://datatracker.ietf.org/doc/html/rfc6962
RFC 8785 (JSON Canonicalization)	https://datatracker.ietf.org/doc/html/rfc8785