10 Branches 4 Tags Go to file Add file Code

3 people feat: add libSQL/Turso e…

e843c18 · 1 hour ago

.claude/…	feat: Add Goog…	5 days ago
.github/…	ci: Enabled buil…	yesterday
channel…	fix: flatten tool …	yesterday
deploy	Bump MSRV t…	9 hours ago
docker	Docker file for …	last week
docs	DM pairing + T…	2 days ago
examples	ci: Added CI/C…	2 days ago
migrations	feat: Sandbox j…	3 days ago
scripts	DM pairing + T…	2 days ago
src	feat: add libSQ…	1 hour ago
tests	feat: add libSQ…	1 hour ago
tools-src	Merge pull req…	3 days ago
wit	DM pairing + T…	2 days ago
wix	ci: Added CI/C…	2 days ago
.dockeri…	feat: Sandbox j…	3 days ago
.env.exa…	Fixes build, ad…	4 days ago
.gitignore	feat: Sandbox j…	3 days ago
AGENT…	Codex/feature …	5 days ago
CHANG…	chore: release …	yesterday

About

IronClaw is OpenClaw inspired implementation in Rust focused on privacy and security

Readme
Apache-2.0, MIT licenses found
Contributing
Activity
Custom properties
944 stars
9 watching
98 forks
Report repository

Releases 3

0.1.3 - 2026-02-12 Latest
yesterday

+ 2 releases

Packages

No packages published

Contributors 10

Languages

Rust 95.6% **JavaScript** 2.2%
CSS 1.4% **PLpgSQL** 0.4%

CLAUD…	feat: add libSQ…	1 hour ago
CONTR…	Codex/feature …	5 days ago
Cargo.lock	feat: add libSQ…	1 hour ago
Cargo.t…	feat: add libSQ…	1 hour ago
Dockerfile	Bump MSRV t…	9 hours ago
Dockerfi…	Bump MSRV t…	9 hours ago
FEATU…	Add OpenAI-c…	13 hours ago
LICENS…	Split LICENSE …	last week
LICENS…	Split LICENSE …	last week
READM…	ci: Disabled np…	yesterday
build.rs	ci: Added CI/C…	2 days ago
docker-…	Add OpenAI-c…	13 hours ago
ironclaw…	Add README …	2 weeks ago
release-…	ci: Skip creatin…	yesterday

HTML 0.2% **Shell** 0.2%

README Contributing Apache-2.0 license MIT license

IronClaw

Your secure personal AI assistant, always on your side

Philosophy • Features • Installation • Configuration • Security • Architecture

Philosophy

IronClaw is built on a simple principle: **your AI assistant should work for you, not against you**.

In a world where AI systems are increasingly opaque about data handling and aligned with corporate interests, IronClaw takes a different approach:

- **Your data stays yours** - All information is stored locally, encrypted, and never leaves your control
- **Transparency by design** - Open source, auditable, no hidden telemetry or data harvesting
- **Self-expanding capabilities** - Build new tools on the fly without waiting for vendor updates
- **Defense in depth** - Multiple security layers protect against prompt injection and data exfiltration

IronClaw is the AI assistant you can actually trust with your personal and professional life.

Features

Security First

- **WASM Sandbox** - Untrusted tools run in isolated WebAssembly containers with capability-based permissions
- **Credential Protection** - Secrets are never exposed to tools; injected at the host boundary with leak detection
- **Prompt Injection Defense** - Pattern detection, content sanitization, and policy enforcement
- **Endpoint Allowlisting** - HTTP requests only to explicitly approved hosts and paths

Always Available

- **Multi-channel** - REPL, HTTP webhooks, WASM channels (Telegram, Slack), and web gateway
- **Docker Sandbox** - Isolated container execution with per-job tokens and orchestrator/worker pattern
- **Web Gateway** - Browser UI with real-time SSE/WebSocket streaming
- **Routines** - Cron schedules, event triggers, webhook handlers for background automation

- **Heartbeat System** - Proactive background execution for monitoring and maintenance tasks
- **Parallel Jobs** - Handle multiple requests concurrently with isolated contexts
- **Self-repair** - Automatic detection and recovery of stuck operations

Self-Expanding

- **Dynamic Tool Building** - Describe what you need, and IronClaw builds it as a WASM tool
- **MCP Protocol** - Connect to Model Context Protocol servers for additional capabilities
- **Plugin Architecture** - Drop in new WASM tools and channels without restarting

Persistent Memory

- **Hybrid Search** - Full-text + vector search using Reciprocal Rank Fusion
- **Workspace Filesystem** - Flexible path-based storage for notes, logs, and context
- **Identity Files** - Maintain consistent personality and preferences across sessions

Installation

Prerequisites

- Rust 1.85+
- PostgreSQL 15+ with pgvector extension
- NEAR AI account (authentication handled via setup wizard)

Download or Build

Visit Releases page to see the latest updates.

▶ Install via Windows Installer (Windows)

▶ Install via powershell script (Windows)

▶ Install via shell script (macOS, Linux, Windows/WSL)

▶ Compile the source code (Cargo on Windows, Linux, macOS)

Database Setup

```
# Create database
createdb ironclaw
```

```
# Enable pgvector
psql ironclaw -c "CREATE EXTENSION IF NOT EXISTS vector;"
```

Configuration

Run the setup wizard to configure IronClaw:

```
ironclaw onboard
```

The wizard handles database connection, NEAR AI authentication (via browser OAuth), and secrets encryption (using your system keychain). All settings are saved to `~/.ironclaw/settings.toml`.

Security

IronClaw implements defense in depth to protect your data and prevent misuse.

WASM Sandbox

All untrusted tools run in isolated WebAssembly containers:

- **Capability-based permissions** - Explicit opt-in for HTTP, secrets, tool invocation
- **Endpoint allowlisting** - HTTP requests only to approved hosts/paths
- **Credential injection** - Secrets injected at host boundary, never exposed to WASM code
- **Leak detection** - Scans requests and responses for secret exfiltration attempts
- **Rate limiting** - Per-tool request limits to prevent abuse
- **Resource limits** - Memory, CPU, and execution time constraints

```
WASM ──▶ Allowlist ──▶ Leak Scan ──▶ Credential ──▶ Execute ──▶ Leak
Scan ──▶ WASM
         Validator     (request)     Injector       Request
(response)
```

Prompt Injection Defense

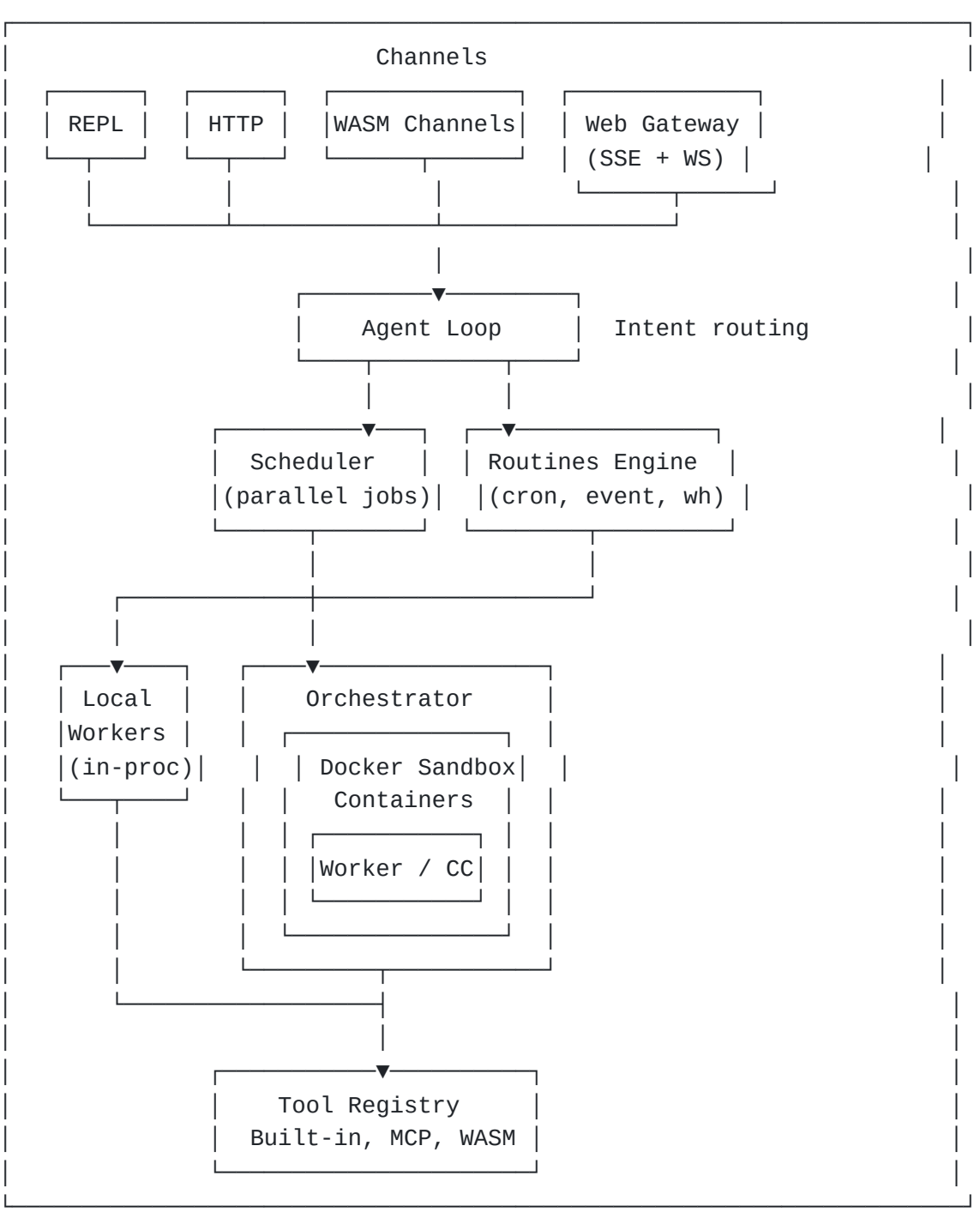
External content passes through multiple security layers:

- Pattern-based detection of injection attempts
- Content sanitization and escaping
- Policy rules with severity levels (Block/Warn/Review/Sanitize)
- Tool output wrapping for safe LLM context injection

Data Protection

- All data stored locally in your PostgreSQL database
- Secrets encrypted with AES-256-GCM
- No telemetry, analytics, or data sharing
- Full audit log of all tool executions

Architecture



Core Components

Component	Purpose
Agent Loop	Main message handling and job coordination
Router	Classifies user intent (command, query, task)
Scheduler	Manages parallel job execution with priorities
Worker	Executes jobs with LLM reasoning and tool calls
Orchestrator	Container lifecycle, LLM proxying, per-job auth
Web Gateway	Browser UI with chat, memory, jobs, logs, extensions, routines
Routines Engine	Scheduled (cron) and reactive (event, webhook) background tasks
Workspace	Persistent memory with hybrid search
Safety Layer	Prompt injection defense and content sanitization

Usage

```
# First-time setup (configures database, auth, etc.)
ironclaw onboard

# Start interactive REPL
cargo run

# With debug logging
RUST_LOG=ironclaw=debug cargo run
```

Development

```
# Format code
cargo fmt

# Lint
cargo clippy --all --benches --tests --examples --all-features

# Run tests
createdb ironclaw_test
cargo test

# Run specific test
cargo test test_name
```

- **Telegram channel**: See docs/TELEGRAM_SETUP.md for setup and DM pairing.

- **Changing channel sources**: Run `./channels-src/telegram/build.sh` before `cargo build` so the updated WASM is bundled.

OpenClaw Heritage

IronClaw is a Rust reimplementation inspired by OpenClaw. See FEATURE_PARITY.md for the complete tracking matrix.

Key differences:

- **Rust vs TypeScript** - Native performance, memory safety, single binary
- **WASM sandbox vs Docker** - Lightweight, capability-based security
- **PostgreSQL vs SQLite** - Production-ready persistence
- **Security-first design** - Multiple defense layers, credential protection

License

Licensed under either of:

- Apache License, Version 2.0 (LICENSE-APACHE)