Clawd/Brutus v2.0 — Production Implementation Guide

1. Executive Summary

1.1 Project Overview

The **Clawd/Brutus v2.0 initiative** represents a fundamental architectural transformation of Flo's personal AI infrastructure, evolving from a single monolithic agent to a **16-agent orchestration system** built on OpenClaw's subagent framework. Dated **February 12, 2026**, and owned by Flo (@notabanker1), this project merges an ambitious multi-agent vision with existing production infrastructure—a 4-node WireGuard mesh with heterogeneous compute resources—while maintaining operational continuity and introducing sophisticated automation capabilities.

The transformation's core innovation is the **"Jarvis-style" orchestrator pattern**: BRUTUS evolves from performing all tasks directly to exclusively **classifying, delegating, and relaying** to specialized subagents. This enables parallel specialization across **Finance, Operations, Content, Life, Security, and Meta clusters** without sacrificing the established user relationship. The implementation leverages OpenClaw's architectural insight that **subagents are lightweight configurations** (prompts + memory + tool permissions, not separate processes), allowing dense consolidation on the existing **clawd-16gb node with 14GB free RAM and 434GB available disk**.

A **critical external deadline** compresses the timeline: Flo begins as **Privatkundenberater at BB-Bank eG in Munich on March 1, 2026**—only **17 days from project inception**. This necessitates aggressive prioritization, with the **Finance Cluster accelerated ahead of strict architectural phase ordering** to deliver functional onboarding support. The production plan balances this urgency against foundational stability requirements, enforcing a **48-hour Telegram delivery reliability gate** before any architectural expansion.

1.2 Current State Assessment

The **v1.0 infrastructure** presents a mixed operational picture with **critical blockers requiring immediate Phase 0 remediation**:

Component	Status	Operational Impact
OpenClaw Gateway (clawd-16gb, v2026.2.6-3)	✅ Stable	Foundation for v2.0 expansion
BRUTUS main agent	✅ Functional	Daily driver, but monolithic bottleneck
NeuroSec security agent	⚠️ Degraded	**Running without baselines—blind to anomalies**
WireGuard mesh	⚠️ 3/4 nodes	Nexus SSH refused, Plutos-32gb offline
Ollama cluster	⚠️ 2/3 nodes	No heavy inference (14B+ models)
Cron jobs (2 active)	⚠️ Broken delivery	Mesh check + NewsClawd both fail

Component	Status	Operational Impact
Telegram delivery	🔴 **Critical failure**	**“Chat not found” errors block all automation**
Skills system	✅ 15+ structured	Mature tooling for extension

Six blockers cascade by dependency:

Blocker	Severity	Fix Required Before	Root Cause
Telegram delivery failing	🔴 Critical	**Phase 0 (everything)**	Bot session/auth mismatch
Plutos-32gb offline	🟡 Medium	Phase 2	Unpaid invoice
Nexus SSH refused	🟡 Medium	Phase 1	Likely fail2ban/SSH daemon crash
NeuroSec baselines missing	🟡 Medium	Phase 1	Never generated
Cron contention at :00	🟡 Medium	Phase 1	Resource competition
No subagent architecture	🟡 High	Phase 2	Core v2.0 transformation

The **Telegram delivery failure** is the absolute prerequisite—without reliable notification channels, no automation delivers value regardless of agent sophistication. The **48-hour stability gate** (consecutive successful mesh status and NewsClawd delivery) enforces this discipline before Phase 1 commencement.

1.3 Target Architecture

The **v2.0 target** implements a **hub-and-spoke orchestration topology**:

```
Flo → Telegram → BRUTUS (orchestrator ONLY — "Jarvis" role)

                    FINANCE CLUSTER  → Macro, Alpha, Banker, Ledger
                    OPS CLUSTER    → Sentinel, Cloud, Compliance
                    CONTENT CLUSTER → Scribe, Trendy, Atlas
                    LIFE CLUSTER   → Pitwall, Munich, Gambit, Mentor
                    SECURITY     → NeuroSec (upgraded with baselines)
                    META       → Digest (morning briefing compiler)
```

Node allocation concentrates agent execution on **clawd-16gb** while distributing specialized workloads:

Node	IP	RAM	Function	Agent Hosting
Nexus	10.0.0.1	1GB	WireGuard hub, security bastion	None (infrastructure only)
Clawd	10.0.0.2	16GB (14GB free)	**OpenClaw Gateway, all subagents, cron, Telegram**	**All 16 agents + BRUTUS orchestrator**
Brutus-8gb	10.0.0.3	8GB	Coding agent, Ollama small models	Cloud (remote SSH ops)
Plutos	10.0.0.4	32GB	Heavy inference endpoint	Inference-only (post-recovery)

This concentration is **architecturally sound**: OpenClaw subagents consume ~**50-100MB RAM per configuration** (not resident processes), with LLM inference via **OpenRouter API calls** rather than local execution. The 14GB free RAM provides substantial headroom for concurrent API buffering and lightweight Ollama 3B fallback.

Model tiering strategy optimizes cost-quality tradeoffs:

Tier	Agents	Primary Model	Fallback	Cost Target
Reasoning-heavy	BRUTUS, Banker, Macro, Alpha, Atlas, Scribe	`openrouter/anthropic/claude-sonnet-4-20250514` or `kimi-k2.5`	Cross-tier fallback	~$3-15 per 1K calls
Scanning/ monitoring	Sentinel, Trendy, Compliance, NeuroSec	`openrouter/xiaomi/mimo-v2-flash` or local Ollama 3B	Local Ollama	~$0.10-0.50 per 1K calls

Tier	Agents	Primary Model	Fallback	Cost Target
Lightweight	Pitwall, Munich, Gambit, Mentor, Ledger, Digest	Local Ollama 3B or mimo-v2-flash	API if local unavailable	Minimal to zero

1.4 Critical Success Factors

Four factors determine v2.0 success, ranked by **dependency order**:

Factor	Target	Measurement
1. Telegram delivery reliability	**>95% uptime**	48-hour continuous successful delivery gate
2. Orchestrator delegation accuracy	**>80% correct routing**	BRUTUS classification → successful subagent completion
3. Morning briefing quality	**Flo-rated 8+/10**	Daily 06:45 CET delivery with fresh, actionable intelligence
4. Cost sustainability	**<$150/month incremental**	Sentinel-monitored OpenRouter spend with tier enforcement

Secondary factors include: NeuroSec baseline completion enabling actual security monitoring; WireGuard mesh restoration to 3+ active nodes; and **BBBank onboarding support delivery by March 1, 2026**—the immovable external deadline driving Finance Cluster acceleration.

---

2. Infrastructure Foundation

2.1 WireGuard Mesh Topology

2.1.1 Node Allocation Strategy The **10.0.0.0/24 WireGuard mesh** implements **purpose-heterogeneous design**—each node's hardware configuration directly determines its functional role, maximizing resource efficiency through hardware-function alignment. This topology provides **encrypted, low-latency interconnectivity** for distributed agent operations while maintaining clear failure domains.

Nexus (10.0.0.1, 1GB RAM) serves as **WireGuard hub and security bastion**. Its minimal RAM **intentionally excludes agent hosting**—attempting to run even lightweight subagents would risk OOM kills and gateway instability. Instead, its network position at the mesh hub enables efficient traffic inspection and **NeuroSec alert relay** without computational load. Current **SSH refusal** indicates likely **fail2ban self-lock or SSH daemon crash**, requiring provider console access for recovery.

Clawd-16gb (10.0.0.2) is the **primary compute node**, hosting the **OpenClaw Gateway, BRUTUS orchestrator, all 16 subagent configurations, cron scheduler, Telegram bot, and Ollama 3B models**. The 14GB free RAM (after ~2GB system/Gateway overhead) supports substantial concurrent API call buffering and local inference. The 434GB free disk accommodates extensive memory file logging, skill repositories, and archival. This concentration simplifies backup, migration, and debugging while maintaining theoretical scalability—subagent configs migrate to other nodes with only path updates if resource pressure demands.

Brutus-8gb (10.0.0.3) functions as **coding agent workstation and Ollama small-model host**. Its 8GB RAM supports **3B-7B parameter models** for offline inference when API connectivity fails or cost optimization requires. The **Cloud agent's remote SSH operations** target this node, enabling distributed server management without Gateway resource contention.

Plutos-32gb (10.0.0.4) provides **heavy inference endpoint capacity** for **14B+ parameter models** when restored from current offline status. This enables **local execution of reasoning-heavy tasks** without API dependency, providing cost reduction and data privacy benefits—particularly critical for **Banker agent's client-sensitive operations**.

2.1.2 Resource Distribution Across 4 Nodes

Node	IP Address	RAM	Status	Primary Function	v2.0 Agent Allocation
Nexus	10.0.0.1	1GB	🔴 SSH refused	WireGuard hub, security bastion, NeuroSec alerts	**None** (infrastructure only)
Clawd	10.0.0.2	16GB (14GB free)	✅ Online	**OpenClaw Gateway, BRUTUS, cron, Telegram, all subagents**	**BRUTUS (orch.), Macro, Alpha, Banker, Scribe, Sentinel, Trendy, Digest, Atlas, Compliance, Pitwall, Munich, Gambit, Mentor, Ledger, NeuroSec**
Brutus-8gb	10.0.0.3	8GB	✅ Online	Coding agent, Ollama small models (3B-7B)	**Cloud** (remote SSH ops target)
Plutos	10.0.0.4	32GB	🔴 Offline	Heavy inference, Ollama 14B+ models	**Inference endpoint only** (post-recovery)

The **87.5% agent concentration on clawd-16gb** contradicts distributed systems orthodoxy but aligns with **OpenClaw's architectural reality**: subagents are **configuration objects, not processes**. Each agent comprises approximately **50-100KB in SOUL.md, CONTEXT.md, and config.json files**—loaded on demand during session spawning, not kept resident. The actual compute load is **API call**

volume and response processing, bounded by OpenRouter rate limits rather than local resources.

Ollama cluster tiering enables intelligent model selection:

Node	Models	Use Case	Latency
Clawd	3B (qwen2.5-coder:3b, phi4:3b)	Lightweight agent tasks, scanning fallback	~50-200ms
Brutus-8gb	7B (llama3.1:8b, qwen2.5:7b)	Intermediate reasoning, coding assistance	~200-500ms
Plutos (recovery)	14B+ (qwen2.5:14b, deepseek-r1:14b, llama3.3:70b)	Deep analysis, complex generation, privacy-critical	~500ms-2s

2.1.3 Failover and Recovery Procedures

Node failure scenarios require differentiated responses based on **functional criticality**:

Failure Mode	Impact	Recovery Procedure	Time Target
Clawd-16gb (Gateway)	🔴 **Catastrophic** — all operations halt	Restore from backup or migrate Gateway to Brutus-8gb with DNS/bot token updates	30-60 min
Nexus SSH refusal	🟡 Security hub degraded, mesh routing impaired	Provider console access → check `/var/log/auth.log` → fail2ban whitelist or `systemctl restart sshd`	<30 min
Plutos offline	🟡 Heavy inference eliminated, API costs increase	Invoice payment → verify WireGuard handshake → restore Ollama models	<15 min post-payment
Brutus-8gb failure	🟡 Coding agent lost, SSH operations degrade to Clawd local	Queue Cloud operations for retry, no immediate user impact	N/A (degraded operation)

Recovery prioritization follows user impact: **Telegram delivery > Gateway availability > mesh**

completeness > heavy inference. The Phase 0-1 sequence addresses critical path items before feature development, with explicit verification at each stage.

2.2 OpenClaw Gateway Configuration

2.2.1 Version Compatibility (v2026.2.6-3) The **v2026.2.6-3 Gateway** provides production-stable capabilities for v2.0 implementation: **subagent spawning via** `sessions_spawn`, **cron job persistence with isolated execution modes**, **multi-channel delivery with Telegram topic support**, and **memory-mapped agent configurations**. Version pinning is recommended—automatic updates risk introducing breaking changes during critical development phases.

Capability verification before Phase 1:

```
# Gateway health check

curl -s http://localhost:18789/health | jq .

# Subagent spawn test

openclaw agent spawn macro --task "Test subagent initialization" --dry-run

# Cron persistence verification

openclaw cron list # Should show existing jobs
```

The **upgrade path to v2026.3.x** (expected March 2026) should follow v2.0 stabilization, with isolated environment testing before production deployment.

2.2.2 Memory and Compute Budgeting **Memory model**: Subagent configurations load into Gateway memory **only during active execution**—persistent storage is disk-based. Concurrent execution, not agent count, determines RAM pressure.

Scenario	Estimated RAM	Notes
Gateway idle	512MB-1GB	Base process, no active sessions
BRUTUS main session	1.5GB	Context window, tool definitions, memory index
Single subagent spawn	+800MB	Temporary session, auto-archived post-completion
Three concurrent subagents (recommended max)	**4GB**	Headroom for API buffering

Scenario	Estimated RAM	Notes
Cron job execution	+600MB	Isolated session, freed after delivery
Local Ollama 3B inference	+2GB	Model weights, active generation

The **4GB concurrent subagent limit** (three agents plus BRUTUS main) shapes delegation patterns: **serialize complex multi-agent queries** rather than parallelizing—"ask Macro, then ask Alpha, then synthesize" matches ADHD-aware sequential delivery preference while preventing resource exhaustion.

Compute budgeting emphasizes **API inference over local models**:

Workload Type	Preferred Execution	Cost Impact
Lightweight tasks (Sentinel checks, Trendy scans)	Local Ollama 3B	Zero API cost
Reasoning tasks (Macro analysis, Banker prep)	OpenRouter premium	~$3-15 per 1K calls
Heavy generation (Atlas research, Scribe drafting)	OpenRouter with Plutos fallback	Variable, monitored

2.2.3 Ollama Cluster Integration **Ollama integration** provides **local inference fallback** for cost control, latency reduction, and privacy-sensitive operations. Configuration uses **OpenAI-compatible endpoint specification** with model aliases for consistent reference:

```
// ~/.openclaw/openclaw.json excerpt
{
  "models": {
    "local-3b": {
      "provider": "ollama",
      "baseUrl": "http://localhost:11434",
      "model": "qwen2.5-coder:3b"
    },
    "local-7b": {
      "provider": "ollama",
      "baseUrl": "http://10.0.0.3:11434",
      "model": "llama3.1:8b"
    },
    "local-14b": {
      "provider": "ollama",
      "baseUrl": "http://10.0.0.4:11434",
      "model": "qwen2.5:14b"
    }
  }
}
```

Agent config.json references these aliases—`"model": "local-3b"` resolves to appropriate endpoint. **Fallback chain**: if Plutos remains offline, "local-14b" references fail gracefully to API fallback; if Brutus-8gb unreachable, "local-7b" routes to API or local-3b.

Health monitoring via Sentinel's `ollama-check` cron (every 30 minutes) verifies API responsiveness across all nodes, with **automatic model tier adjustment**—agents configured for unavailable local models receive API fallback without manual intervention.

2.3 Telegram Delivery System

2.3.1 Bot Authentication and Session Management The **@brutusclawdbot Telegram delivery failures** ("chat not found" errors) represent the **single highest-priority blocker** for v2.0. Root cause analysis identifies **two probable failure modes**:

Failure Mode	Diagnostic	Resolution
Bot token invalid/expired	`curl "https://api.telegram.org/bot<TOKEN>/getMe"` returns `{"ok":false}`	Regenerate via @BotFather, update `~/.openclaw/credentials/telegram/bot_token`
Chat_id mismatch	`getMe` succeeds but `sendMessage` to stored chat_id fails	Flo sends `/start` to @brutusclawdbot, extract fresh chat_id from `getUpdates`

Critical discovery: Telegram bots **cannot initiate conversations**—user must send `/start` to establish chat context. Session persistence in `~/.openclaw/telegram/` may cache **stale chat_id values** from previous authorizations. **Nuclear recovery option**: `rm -rf ~/.openclaw/telegram/` forces fresh session establishment.

Session verification protocol:

```
# 1. Verify bot token

curl -s "https://api.telegram.org/bot${TOKEN}/getMe" | jq '.ok'

# 2. Check stored configuration

cat ~/.openclaw/openclaw.json | jq '.channels.telegram'

# 3. Retrieve current chat_id

curl -s "https://api.telegram.org/bot${TOKEN}/getUpdates" | \
  jq '.result[-1].message.chat.id'
```

```
# 4. Test direct delivery

openclaw send-message "Verification test $(date)" --chat @notabanker1
```

2.3.2 Chat ID Verification Protocol Systematic **chat_id verification** prevents recurrence and enables diagnostic automation. Implement **Sentinel health check integration**:

```
#!/bin/bash

# ~/.openclaw/skills/telegram-verify/check.sh

TOKEN=$(cat ~/.openclaw/credentials/telegram/bot_token)
CHAT_ID=$(jq -r '.channels.telegram.chat_id' ~/.openclaw/openclaw.json)

BOT_INFO=$(curl -s "https://api.telegram.org/bot${TOKEN}/getMe")
[[ $(echo "$BOT_INFO" | jq -r '.ok') != "true" ]] && \
  echo "🔴 TELEGRAM_AUTH_FAIL" && exit 1

SEND_RESULT=$(curl -s -X POST "https://api.telegram.org/bot${TOKEN}/sendMessage" \

  -d "chat_id=${CHAT_ID}" -d "text=Verification $(date +%H:%M:%S)")

[[ $(echo "$SEND_RESULT" | jq -r '.ok') != "true" ]] && \
  echo "🔴 TELEGRAM_SEND_FAIL: $(echo "$SEND_RESULT" | jq -r '.description')" && \
  echo "⚠️ Remediation: Flo must send /start to @brutusclawdbot" && exit 1

echo "✅ TELEGRAM_OK"
```

Multi-channel configuration supports future expansion: primary (Flo direct), secondary (monitoring channel for critical alerts), tertiary (log channel for debugging). Each requires explicit `/start` authorization.

2.3.3 Message Routing and Rate Limiting **Telegram Bot API rate limits** (30 messages/second to same chat, 20 messages/minute to same group) require **aggregation strategies** for 16-agent system:

Strategy	Implementation	Benefit
Per-agent message caps	Max 1 message per agent per 5-minute window to Flo	Prevents spam, respects attention
Digest aggregation	Morning briefing compiles 4+ agent outputs into single message	Channel usability, ADHD optimization
Priority queuing	🔴 Critical bypass aggregation; 🟡/🟢 respect quiet hours	Appropriate urgency routing

Strategy	Implementation	Benefit
Length-based splitting	>4000 characters split with “1/3”, “2/3”, “3/3” headers	Technical compliance

Energy-aware delivery windows:

Window	Message Type	Rationale
00:00-12:00 CET	🔴 Critical alerts only	Flo’s low-energy morning
12:00-16:00 CET	Light touches, batched updates	Transition to peak
16:00-02:00 CET	**Full engagement, complex outputs**	**Flo’s peak cognitive window**
02:00+ CET	Queue for tomorrow, “go to bed” nudge	Sleep hygiene support

---

3. Agent Architecture Design

3.1 Orchestrator Pattern: BRUTUS v2.0

3.1.1 Role Transformation from Worker to Router The **BRUTUS v2.0 transformation** is the **most consequential architectural decision** in v2.0—shifting from **generalist task execution** to **specialist coordination** while **preserving established user relationship dynamics**. This pattern, deployed successfully in production systems from Microsoft’s Copilot stack to financial trading infrastructure, separates **human relationship management** from **domain-specific execution**.

Capability surrender is explicit and comprehensive:

BRUTUS v1.0 (Worker)	BRUTUS v2.0 (Orchestrator)
Executes market analysis directly	**Classifies** → spawns **Macro/Alpha**
Drafts emails and documents	**Routes** → spawns **Scribe**
Monitors infrastructure status	**Aggregates** → spawns **Sentinel**
Prepares client meeting materials	**Delegates** → spawns **Banker**
Tracks personal reminders	**Coordinates** → spawns **Munich/Pitwall**

Retained capabilities: quick factual answers (time, calculations), **task routing decisions** (the classification logic itself), **daily digest compilation coordination**, and **personality maintenance** (the “best-buddy” relationship with Gen-Z slang and ADHD-aware communication).

Delegation workflow (five-step):

1. **Receive**: Flo sends message to BRUTUS via Telegram
2. **Classify**: Intent analysis against 16-agent registry
3. **Delegate**: `sessions_spawn` with appropriate context
4. **Collect**: Result aggregation from subagent
5. **Relay**: Formatted response with BRUTUS personality overlay

This introduces **15-30 second latency** (subagent spawn + API call + relay) versus direct execution, trading time for **expertise quality**—acceptable given non-urgent nature of most queries.

3.1.2 Delegation Logic and Classification Rules **Intent classification** implements **decision tree with confidence thresholds**:

Intent Pattern	Primary Agent	Confidence	Fallback	Multi-Agent Trigger
"ECB/Fed/rates/ policy"	**Macro**	0.95	BRUTUS direct	"How do rates affect my mortgage?" → Macro + Banker
"DAX/stocks/ price/target"	**Alpha**	0.90	Macro	"European banks" → Alpha + Macro
"client/meeting/ prep/advice"	**Banker**	0.95	BRUTUS (privacy warning)	Sequential with Macro for context
"server/SSH/ down/error"	**Cloud**	0.90	Sentinel	Cloud execution, Sentinel monitoring
"email/draft/ write/send"	**Scribe**	0.85	BRUTUS (simple only)	Scribe draft → BRUTUS approval relay
"news/scan/ opportunity"	**Trendy**	0.80	Alpha	Trendy scan + Alpha deep-dive on flag
"race/Herberth/ F1/calendar"	**Pitwall**	0.95	BRUTUS	Dual calendar (GT3 + F1) tracking
"tax/Sparen/ finance personal"	**Ledger**	0.90	Banker (if BBBank-related)	Clear personal/professional boundary

Confidence handling:

- **>0.8**: Spawn single agent, proceed
- **0.5-0.8**: Spawn primary with secondary standby, or request clarification

- **<0.5**: Ask clarifying question—"Not sure if you want market analysis (Alpha) or macro context (Macro). Which angle?"

Multi-intent messages: "Prepare me for tomorrow's client meeting about the ECB rate decision" requires **sequential delegation with context passing**—Macro analysis feeds Banker prep, BRUTUS synthesizes final briefing. **Parallel spawning avoided** due to resource constraints and ADHD-preferential sequential presentation.

3.1.3 ADHD-Aware Communication Protocols **ADHD support is architectural requirement, not optional enhancement**. Flo's diagnosed ADHD creates specific interaction patterns that v2.0 must accommodate through **explicit operational rules**:

Pattern	Implementation	Example
Action-first responses	Lead with DO, then context	"**Hold steady on rate guidance** — ECB kept 3.75% as expected. Context: inflation sticky…"
Chunked outputs	Max 5 bullet points, "Want more?" continuation	Break at 5, explicit prompt for expansion
External memory as compensation	Automatic task_log.md logging, proactive deadline surfacing	"You said you'd review BBBank docs by Friday — that's tomorrow"
No repetition penalty	Fresh answers, memory check for changed context	Never "as I mentioned" — just answer
Deadline surfacing	Countdown format with multi-stage reminders	"Due in 7 days" → 3 days → 1 day → day-of
Decision scaffolding	Max 3 options, explicit recommendation	"I'd go with B because…"
Energy-aware scheduling	Critical-only before noon, full engagement 4pm-2am	Queue non-urgent, "go to bed" after 2am
Capture everything	Passing mentions logged as tasks	"Oh I need to call my mom" → auto-logged

BRUTUS-specific ADHD section in SOUL.md:

```
## Your Job as External Brain

Flo has ADHD. You are his external working memory and executive function scaffold.
```

```
ALWAYS:

- Log commitments to task_log.md with deadline calculation

- Surface deadlines proactively ("Hey, that BBBank thing is in 3 days")

- Gently redirect when scattered: "You were working on X — continue or switch?"

- Maintain "parking lot" for half-finished thoughts

NEVER:

- Long unstructured walls of text

- Expect recall from 3+ conversations ago — check memory

- "As I mentioned earlier" phrasing

- Non-urgent messages before noon CET

ENERGY-AWARE SCHEDULING:

- Before 12:00: Only 🔴 CRITICAL (system down, security, money)

- 12:00-16:00: Light touches, queued updates

- 16:00-02:00: Full engagement, complex tasks

- After 02:00: Queue all, "go to bed, bro" if active at 3am
```

3.2 Cluster Organization

3.2.1 Finance Cluster: Macro, Alpha, Banker, Ledger The **Finance Cluster** delivers **highest business value** and **most urgent development priority** due to **March 1, 2026 BBBank deadline**. Four agents cover distinct temporal and functional domains with **accelerated onboarding support**:

Agent	Domain	Key Capability	BBBank Relevance
Macro	Central bank policy, monetary regimes	Daily 05:30 scan, regime assessment, event anticipation	**Rate environment for product positioning**
Alpha	Asset-specific analysis, market research	Watchlist management, valuation metrics, catalyst tracking	**European bank stocks, client portfolio context**

Agent	Domain	Key Capability	BBBank Relevance
Banker	Client advisory, meeting preparation	**Onboarding mode**: DZ Bank ecosystem, cooperative model, regulatory framework	**Direct professional enablement**
Ledger	Personal finance, tax optimization	Sparerpauschbetrag tracking, quarterly reviews, goal monitoring	**Personal/professional boundary maintenance**

Banker agent's ONBOARDING MODE (February 2026 priority):

Day	Task	Output
1-3	BBBank eG research: history, cooperative model, Munich presence	CONTEXT.md foundation
3-5	DZ Bank ecosystem mapping: Union Investment, R+V, Schwäbisch Hall, DZ HYP, TeamBank, VR Smart Finanz	Product universe table
5-7	Quick-reference cards: key facts, fees, risk levels, suitable profiles	RUNBOOK.md procedures
7-10	Regulatory research: § 64 WpHG Anlageberatung, Geeignetheitserklärung, MiFID II suitability	Compliance integration
10-14	"First Week Survival Kit": contacts template, system checklist, compliance must-knows	Scribe-formatted document
Ongoing	Daily 18:00 "BBBank Product Fact" via Telegram	Accelerated knowledge transfer

Macro supports onboarding: "Here's current ECB rate environment and what it means for products you'll sell: Bauspar rates, mortgage rates, bond fund outlook, Tagesgeld money market rates." This gives Flo **pre-built mental model** walking into BBBank March 1.

3.2.2 Operations Cluster: Sentinel, Cloud, Compliance

Agent	Function	Replaces v1.0	Key Capabilities
Sentinel	Unified infrastructure monitoring	`hourly-mesh-confirmation` + `hourly-newsclawd-update`	15-min health checks, Ollama verification, cost tracking, crypto monitoring, severity-graded alerting
Cloud	Remote server operations	Manual SSH	SSH to mesh nodes, read-only default, write-with-approval, operation logging
Compliance	Regulatory monitoring	None (new)	Weekly BaFin/ESMA scan, Friday 16:00 digest, MiFID II/ KWG tracking

Sentinel cron schedule (replaces broken v1.0 jobs):

Job	Schedule	Scope	Severity Handling
`sentinel-health`	`*/15 * * * *`	Mesh pings, process checks, memory/disk	🔴 immediate, 🟡 digest, 🟢 log
`sentinel-ollama`	`*/30 * * * *`	Ollama API health all nodes	Degradation alert, fallback trigger
`sentinel-costs`	`0 */4 * * *`	OpenRouter spend tracking	200% baseline → critical alert
`sentinel-crypto`	`0 */2 * * *`	BTC/ETH + ArbitrageAndy status	Threshold breach alerts
`sentinel-daily-summary`	`0 20 * * *`	Aggregated infrastructure report	Evening digest inclusion

3.2.3 Content Cluster: Scribe, Trendy, Atlas

Agent	Function	Integration	Key Constraint
Scribe	Communications, document drafting	**Gmail OAuth2** (Phase 4a read-only, 4b send-with-approval)	**Never auto-send —human-in-the-loop mandatory**
Trendy	Market opportunity scanning	Polymarket, crypto, AI news, 90-min intervals	Replaces NewsClawd with broader scope
Atlas	Deep research, strategic analysis	On-demand BRUTUS delegation, 20-min time-boxed	Progress updates every 2 min to prevent abandonment

Scribe's phased Gmail integration protects professional liability: Phase 4a enables inbox triage and summarization; Phase 4b adds draft composition with **explicit Flo approval before any send operation**. OAuth2 credential reuse from existing clawdbot accelerates setup.

3.2.4 Life Cluster: Pitwall, Munich, Gambit, Mentor

Agent	Domain	Frequency	Integration
Pitwall	Motorsport logistics (Herberth GT3 + F1)	3-week, 1-week, 3-day, day-before race alerts	Morning briefing, vacation planning coordination
Munich	Local life management	Weekly cron, seasonal triggers	Apartment reminders, administrative deadlines, local events
Gambit	Chess training	Daily puzzle delivery	Lichess API, morning briefing inclusion, rating tracking
Mentor	Learning roadmap	Weekly Sunday 19:00 review	Activity pattern analysis, skill gap identification, resource recommendation

These agents operate at **lower frequency and complexity**, utilizing **lightweight models** (Mimo-v2-Flash or local Ollama 3B) for cost efficiency. Their value is **automated life management reducing cognitive overhead** rather than professional enablement.

3.2.5 Security Layer: NeuroSec **NeuroSec upgrades from v1.0 placeholder to active monitoring** through **baseline completion and enhanced detection**:

Baseline	Generation Command	Detection Scope
`permissions.json`	`find /home/boss/.openclaw -type f -exec stat -c '%n %a %U:%G' {} \;`	File permission changes, ownership drift, unexpected additions
`network.json`	`ss -tlnp | jq -R 'split(" ")| {port: .[3], process: .[6]}'`	New listening ports, missing expected services, process changes
`secrets.json`	`sha256sum ~/.openclaw/credentials/*`	Credential file modification (hash-only, no plaintext storage)

6-hour scan cycle with severity classification: **critical** (immediate Telegram), **warning** (Digest aggregation), **info** (log-only). Clinical persona maintained—factual, structured, actionable: "PERMISSION ANOMALY: ~/.openclaw/openclaw.json mode 644 (expected 600). RECOMMENDED ACTION: chmod 600 ~/.openclaw/openclaw.json."

3.2.6 Meta Layer: Digest **Digest is the critical integration point**—the "killer feature" demonstrating v2.0's distributed intelligence value. **06:45 daily execution** compiles cross-cluster outputs into **ADHD-optimized morning briefing**:

Source Agent	Content	Freshness Requirement
Macro	Overnight markets, central bank news	<6 hours (05:30 scan)
Sentinel	System health, API costs	<30 minutes (15-min checks)
Pitwall	Race weekend proximity	<24 hours
Munich	Daily reminders	<12 hours

TEMPLATE.md enforces strict format:

```
☀️ GUT'N MORGEN FLO — {{DAY}}, {{DATE}}

📅 HEUTE

- {{calendar_items}}

- {{reminders_from_munich}}

- {{race_weekend_alert_from_pitwall}}

💰 MÄRKTE OVERNIGHT

- DAX: {{level}} ({{change}}) | S&P: {{level}} ({{change}})
- EUR/USD: {{level}} | 10Y Bund: {{yield}} | 10Y UST: {{yield}}

- BTC: {{price}} ({{change}}) | ETH: {{price}} ({{change}})

- {{macro_headline_if_any}}

🏦 FÜR DIE ARBEIT

- {{client_meetings_today}}

- {{macro_talking_points}}

- {{compliance_alerts_if_any}}

🤖 SYSTEME

- Mesh: {{mesh_status}} | Ollama: {{ollama_status}}

- API costs yesterday: ${{amount}}

- {{critical_alerts_if_any}}

📈 OPPORTUNITIES

- {{top_trendy_finding}}

- {{polymarket_notable}}

♟ {{daily_puzzle_from_gambit}}

🎯 TOP 3 FÜR HEUTE
1. {{priority_1}}
2. {{priority_2}}
3. {{priority_3}}
```

Design rules: <2 minute read, **bold action items at top**, numbered/bulleted only, 🔴/🟡/🟢 urgency coding, **exactly 3 priorities**.

3.3 Subagent Configuration Schema

3.3.1 Directory Structure Standardization

All 16 agents follow **identical directory structure**:

```
~/.openclaw/workspace/agents/{agent_name}/
   SOUL.md # Required: personality, principles, constraints
   config.json # Required: model, tools, permissions, cron
   CONTEXT.md # Optional: persistent state, current focus
   RUNBOOK.md # Optional: operational procedures, SOPs
   TEMPLATE.md # Optional: output formatting (Digest only)
```

Provisioning command:

```
mkdir -p ~/.openclaw/workspace/agents/{macro,alpha,banker,sentinel,scribe,trendy,cloud,neurosec,
    digest,atlas,compliance,pitwall,munich,gambit,mentor,ledger}
```

Parallel **memory structure**:

```
~/.openclaw/workspace/memory/agents/{agent_name}/
  {domain_specific_files}.md # Persistent state
  {domain_specific_files}.json # Structured data
```

3.3.2 config.json Specification

Field	Type	Required	Description	Example
`name`	string	Yes	Machine identifier	`"macro"`
`display_name`	string	Yes	Human-readable label	`"MACRO — Central Bank Analyst"`
`model`	string	Yes	Primary model	`"openrouter/moonshotai/kimi-k2.5"`
`model_fallback`	string	No	Degradation target	`"openrouter/xiaomi/mimo-v2-flash"`
`tools`	string[]	Yes	Enabled capabilities	`["web_search", "file_read", "file_write"]`
`memory_path`	string	Yes	Base for agent memory	`"memory/agents/macro/"`
`permissions`	object	Yes	Fine-grained access control	See below
`cron`	object[]	No	Scheduled tasks	See below
`security`	object	No	Agent-specific constraints (Banker)	PII rules

Permissions object:

```
{
  "web_search": true,
  "file_read": ["memory/agents/macro/*", "memory/global/*"],
  "file_write": ["memory/agents/macro/*"],
  "telegram_send": false,
  "ssh": false,
  "api_calls": ["coinbase", "coingecko"]
}
```

Cron entry structure:

```
{
  "name": "macro-daily-scan",
  "schedule": "30 5 * * *",
  "task": "Check central bank calendar...",
  "delivery": "memory_only"
}
```

`delivery` options: `memory_only` (silent), `telegram_via_brutus` (alert), `digest_contribution` (flag for compilation).

3.3.3 SOUL.md Personality Framework **Required sections**: Identity, Purpose, Principles, Constraints, Communication Style. **ADHD Support section** mandatory for all agents, with **BRUTUS-specific delegation logic**.

Macro SOUL.md excerpt:

```
# MACRO — Central Bank Analyst

## Identity

You track global monetary policy with obsessive precision.
You think in regimes: tightening, easing, pause, and transitions.

## Purpose

Provide Flo with actionable understanding of monetary policy trajectory
and its implications for asset prices, economic activity, and BBBank decisions.

## Principles

- Lead with regime assessment

- Quantify uncertainty (facts vs pricing vs judgment)

- Connect to Flo's context (BBBank products, client conversations)

## Constraints

- Never predict with false precision

- Distinguish your analysis from consensus
- Flag data limitations explicitly

## Communication

- Professional but accessible

- Numbers-first, then interpretation

- 3-5 paragraphs standard, expandable on request

## ADHD Support

- Chunked regime summaries

- Explicit "what this means for you" translation

- Calendar countdowns for key events
```

3.3.4 CONTEXT.md State Management **Agent-self-maintained persistent state** enabling longitudinal tracking:

Agent	Key CONTEXT.md Sections
Macro	Current regime assessment, this week's calendar, active monitoring topics, recent surprises
Banker	BBBank onboarding progress, product knowledge gaps, anonymized meeting patterns, certification checklist
Sentinel	Threshold configurations, alert history with resolution, known false positives, cost tracking trends

Update protocols: cron execution appends, BRUTUS delegation with explicit "update context" instruction, self-reflection on task completion.

3.3.5 RUNBOOK.md Operational Procedures **Complex agents only**—documenting multi-step workflows:

Agent	Key RUNBOOK.md Procedures
Macro	Rate Decision Response (5-min immediate, 15-30 min analysis), Regime Assessment Update
Banker	Client Meeting Preparation (agenda, research, objections, follow-up), Product Inquiry Handling

Agent	Key RUNBOOK.md Procedures
Sentinel	Alert Escalation Protocol, Threshold Tuning Procedure, Cost Anomaly Investigation
Cloud	Incident Response (diagnose, isolate, remediate, verify, document), SSH Safety Procedures

---

4. Model Strategy and Cost Optimization

4.1 Tiered Model Selection

4.1.1 Reasoning-Heavy Tier (Claude Sonnet, Kimi K2.5)

Agent	Primary	Fallback	Est. Daily Calls	Daily Cost
BRUTUS	`claude-sonnet-4-20250514`	`kimi-k2.5`	50	~$4.00
Banker	`kimi-k2.5`	`claude-sonnet-4-20250514`	10	~$2.00
Macro	`kimi-k2.5`	`mimo-v2-flash`	15	~$2.25
Alpha	`kimi-k2.5`	`mimo-v2-flash`	15	~$2.25
Atlas	`claude-sonnet-4-20250514`	`kimi-k2.5`	5	~$2.50
Scribe	`kimi-k2.5`	`mimo-v2-flash`	10	~$1.75

Total reasoning-tier estimate: ~$14.75/day, ~**$445/month** at full utilization. Actual lower due to fallback triggering and query batching. **Kimi K2.5 as default** balances capability and cost (~33% savings vs Claude Sonnet).

4.1.2 Scanning/Monitoring Tier (Mimo-v2-Flash, Ollama 3B)

Model	Cost	Use Case	Agents
`mimo-v2-flash`	~$0.10-0.50/1K calls	High-frequency scanning, structured extraction	Sentinel, Trendy, Compliance
`ollama/qwen2.5-coder:3b`	Zero (compute only)	Private data processing, deterministic tasks	NeuroSec, Sentinel (fallback)

10x cost reduction enables aggressive scheduling: 15-minute Sentinel checks, 90-minute Trendy scans without budget impact.

4.1.3 Lightweight Tier (Local Ollama Models)

Agent	Model	Task	Cost
Digest	`phi3:mini` or `qwen2.5:0.5b`	Template compilation, data aggregation	~$0-0.05
Pitwall, Munich, Gambit, Mentor, Ledger	`qwen2.5-coder:3b`	Reminders, puzzles, tracking	$0 (local)

Fallback to mimo-v2-flash on Ollama unavailability maintains service continuity.

4.2 OpenRouter Integration

4.2.1 API Key Management

```
// ~/.openclaw/openclaw.json
{
  "env": {
    "OPENROUTER_API_KEY": "sk-or-v1-..."
  },
  "models": {
    "providers": {
      "openrouter": {
        "baseUrl": "https://openrouter.ai/api/v1",
        "apiKey": "${OPENROUTER_API_KEY}",
        "models": [
          {"id": "anthropic/claude-sonnet-4-20250514", "alias": "sonnet"},
          {"id": "moonshotai/kimi-k2.5", "alias": "kimi"},
          {"id": "xiaomi/mimo-v2-flash", "alias": "mimo"}
        ]
      }
    }
  }
}
```

Security: Key in environment variable enables rotation without config modification. **OpenRouter dashboard**: monthly spend caps, per-model rate limits, IP allowlisting.

4.2.2 Fallback Chain Configuration

Tier	Primary	Fallback 1	Fallback 2	Final
Reasoning	Kimi K2.5	Claude Sonnet	Local Ollama 7B	Explicit failure + BRUTUS notification
Scanning	Mimo-v2-flash	Local Ollama 3B	Cached last result	Stale warning

Tier	Primary	Fallback 1	Fallback 2	Final
Lightweight	Local Ollama 3B	Mimo-v2-flash	—	BRUTUS degraded service notice

Fallback triggers: HTTP 429/5xx, >30s timeout, content policy rejection. Events logged to `memory/cost_tracker.md` for pattern analysis.

4.2.3 Cost Tracking and Alerting **Sentinel cost-check cron** (every 4 hours):

Threshold	Alert	Action
50% daily budget ($10)	Warning in Digest	Monitor trend
80% daily budget ($16)	Immediate Telegram	Investigate anomaly
100% daily budget ($20)	Critical + model tier downgrade	Switch to local inference only
200% baseline (weekly)	Emergency review	Full model rebalancing

---

5. Memory and Persistence Architecture

5.1 Shared Memory Layer

5.1.1 Global State: mesh_status, agent_registry, task_log

File	Maintainer	Update Frequency	Purpose
`mesh_status_latest.json`	Sentinel	15 minutes	Current node health, Ollama status, latency
`agent_registry.md`	BRUTUS	On change	Agent list, capabilities, status for delegation
`task_log.md`	BRUTUS	Real-time	Commitments, deadlines, completions for ADHD support

File	Maintainer	Update Frequency	Purpose
`cost_tracker.md`	Sentinel	4 hours	API spend, projections, optimization opportunities

5.1.2 Per-Agent Memory Isolation Each agent's `memory_path` contains domain-specific persistent state:

Agent	Key Memory Files
Macro	`regime_assessment.md`, `central_bank_calendar.md`, `rate_history.md`
Banker	`meeting_notes.md` (anonymized), `product_knowledge.md`, `onboarding_progress.md`
Sentinel	`alert_history.md`, `thresholds.json`, `cost_trends.md`
Pitwall	`race_calendar_2026.md` (GT3 + F1), `travel_bookings.md`

5.1.3 Cross-Agent Read Permissions **Digest's unique broad read access**: `"file_read": ["memory/agents/*/"]` enables morning briefing compilation. **Write isolation enforced**: agents modify only own memory, preventing cross-contamination.

5.2 File-Based Persistence

5.2.1 JSON State Files Machine-readable, schema-versioned: `mesh_status_latest.json`, `thresholds.json`, `secrets.json`. Pretty-printed for emergency human inspection.

5.2.2 Markdown Knowledge Bases Human-readable, Git-friendly: all `.md` files. `~/.openclaw/workspace/` should be **Git-initialized** with `.gitignore` for `credentials/`, `alerts/`.

5.2.3 Log Rotation and Archival

Data Type	Retention	Rotation Trigger
Daily logs	30 days	Size >10MB or age >7 days
Compressed archives	90 days	Weekly cron
Deep archival	1 year	Monthly to off-mesh storage

5.3 NeuroSec Baseline System

Baseline	Generation	Update Frequency	Alert Condition
`permissions.json`	`find /home/boss/.openclaw -type f -exec stat -c '%n %a %U:%G' {} \;`	Weekly + after intentional changes	Permission change, new file in sensitive path, ownership drift
`network.json`	`ss -tlnp | jq ...`	Every 6 hours	New listening port, missing expected service, process change
`secrets.json`	`sha256sum ~/.openclaw/credentials/*`	Weekly + after rotation	Hash mismatch (modification), file disappearance, unexpected new credential file

---

6. Cron Automation Framework

6.1 Scheduling Philosophy

6.1.1 Timezone-Aware Execution (CET) All schedules use `Europe/Berlin` (CET/CEST automatic). Critical schedules avoid 02:00-03:00 window (DST transition ambiguity).

6.1.2 Offset Strategy to Prevent Contention

Minute	Jobs	Rationale
:00	(reserved—lightest only)	Base tick, minimal load
:05, :10	(buffer for expansion)	Post-:00 recovery
:15, :30, :45	`sentinel-health`	Regular distribution
:30, :35	`macro-scan`, `alpha-scan`	Pre-briefing data gathering
:45	`morning-briefing` (06:45 only)	Compilation after sources complete

No two jobs share minute marks—eliminates v1.0 contention.

6.1.3 Energy-Aware Windows (Flo’s 4pm-2am Peak)

Window	Job Types	Examples
00:00-12:00	Background monitoring only	sentinel-health, sentinel-ollama, sentinel-crypto, sentinel-costs
12:00-16:00	Light touches, non-urgent	munich-reminders, pitwall-calendar-check
16:00-02:00	**Full engagement, complex outputs**	**BRUTUS delegation, Atlas research, Scribe drafting**
02:00-00:00	Queue for tomorrow, sleep prompts	evening-digest, "go to bed, bro" if active at 03:00

6.2 Job Categories

6.2.1 High-Frequency Monitoring (15-min intervals)

'sentine

Clawd/Brutus v2.0 — Production Implementation Guide

1. Executive Summary

1.1 Project Overview

The **Clawd/Brutus v2.0 initiative** represents a fundamental architectural transformation of Flo's personal AI infrastructure, evolving from a single monolithic agent (BRUTUS v1.0) to a **16-agent orchestration system** built on OpenClaw's subagent framework. This project, dated **February 12, 2026**, and owned by Flo (@notabanker1), addresses critical operational limitations while establishing a scalable foundation for autonomous task delegation, specialized domain expertise, and ADHD-aware human-AI collaboration.

The transformation's core innovation is the **"Jarvis-style" orchestrator pattern**: BRUTUS evolves from performing all tasks directly to exclusively **classifying, delegating, and relaying** to specialized subagents organized across **six functional clusters**—Finance, Operations, Content, Life, Security, and Meta. This design preserves Flo's established relationship with BRUTUS (single Telegram interface, consistent personality, trusted communication style) while dramatically expanding cognitive capabilities through invisible specialization.

The implementation leverages **OpenClaw's lightweight subagent architecture**, where agents exist as **configuration bundles** (prompts + memory paths + tool permissions, not separate processes), enabling dense consolidation on the existing **clawd-16gb node with 14GB free RAM and 434GB available disk**. LLM inference occurs primarily via **OpenRouter API calls** rather than local execution, with intelligent tiering optimizing cost-quality tradeoffs.

A **critical external deadline** compresses the timeline: Flo begins as **Privatkundenberater at BB-Bank eG in Munich on March 1, 2026**—only **17 days from project inception**. This necessitates aggressive prioritization, with the **Finance Cluster accelerated ahead of strict architectural phase ordering** to deliver functional onboarding support. The production plan balances this urgency against foundational stability requirements, enforcing a **48-hour Telegram delivery reliability gate** before any architectural expansion.

1.2 Current State Assessment

The **v1.0 infrastructure** presents a mixed operational picture with **critical blockers requiring immediate Phase 0 remediation**:

Component	Status	Operational Impact
OpenClaw Gateway (clawd-16gb, v2026.2.6-3)	✅ Stable	Foundation for v2.0 expansion
BRUTUS main agent	✅ Functional	Daily driver, but monolithic bottleneck
NeuroSec security agent	⚠️ **Degraded**	**Running without baselines—blind to anomalies**
WireGuard mesh	⚠️ **3/4 nodes**	Nexus SSH refused, Plutos-32gb offline
Ollama cluster	⚠️ **2/3 nodes**	No heavy inference (14B+ models)
Cron jobs (2 active)	⚠️ **Broken delivery**	Mesh check + NewsClawd both fail
Telegram delivery	🔴 **Critical failure**	**"Chat not found" errors block all automation**
Skills system	✅ 15+ structured	Mature tooling for extension

Six blockers cascade by dependency:

Blocker	Severity	Fix Required Before	Root Cause / Resolution
Telegram delivery failing	🔴 **Critical**	**Phase 0 (everything)**	Bot session/auth mismatch —requires `/start` from Flo, chat_id verification
Plutos-32gb offline	🟡 Medium	Phase 2	Unpaid invoice—payment restores heavy inference
Nexus SSH refused	🟡 Medium	Phase 1	Likely fail2ban self-lock or SSH daemon crash—provider console access required

Blocker	Severity	Fix Required Before	Root Cause / Resolution
NeuroSec baselines missing	🟡 Medium	Phase 1	Never generated—create permissions.json, network.json, secrets.json
Cron contention at :00	🟡 Medium	Phase 1	Resource competition—offset scheduling to :05, :10, :15, etc.
No subagent architecture	🟡 High	Phase 2	Core v2.0 transformation —BRUTUS refactor, 16-agent directory structure

The **Telegram delivery failure** is the **absolute prerequisite**—without reliable notification channels, no automation delivers value regardless of agent sophistication. The **48-hour stability gate** (consecutive successful mesh status and NewsClawd delivery) enforces this discipline before Phase 1 commencement.

1.3 Target Architecture

The **v2.0 target** implements a **hub-and-spoke orchestration topology** with BRUTUS as the exclusive human-facing interface:

```
Flo → Telegram → BRUTUS (orchestrator ONLY — "Jarvis" role)

                    **FINANCE CLUSTER**  → Macro, Alpha, Banker, Ledger
                    **OPS CLUSTER**    → Sentinel, Cloud, Compliance
                    **CONTENT CLUSTER** → Scribe, Trendy, Atlas
                    **LIFE CLUSTER**   → Pitwall, Munich, Gambit, Mentor
                    **SECURITY**      → NeuroSec (upgraded with baselines)
                    **META**        → Digest (morning briefing compiler)
```

Node allocation concentrates agent execution on **clawd-16gb** while distributing specialized workloads:

Node	IP	RAM	Function	Agent Hosting
Nexus	10.0.0.1	1GB	WireGuard hub, security bastion	**None** (infrastructure only)
Clawd	10.0.0.2	16GB (14GB free)	**OpenClaw Gateway, all subagents, cron, Telegram**	**All 16 agents + BRUTUS orchestrator**

Node	IP	RAM	Function	Agent Hosting
Brutus-8gb	10.0.0.3	8GB	Coding agent, Ollama small models	Cloud (remote SSH ops)
Plutos	10.0.0.4	32GB	Heavy inference endpoint	Inference-only (post-recovery)

This concentration is **architecturally sound**: OpenClaw subagents consume ~**50-100MB RAM per configuration** (not resident processes), with LLM inference via **OpenRouter API calls**. The 14GB free RAM provides substantial headroom for concurrent API buffering and lightweight Ollama 3B fallback.

Model tiering strategy optimizes cost-quality tradeoffs:

Tier	Agents	Primary Model	Fallback	Cost Target
Reasoning-heavy	BRUTUS, Banker, Macro, Alpha, Atlas, Scribe	`openrouter/anthropic/claude-sonnet-4-20250514` or `kimi-k2.5`	Cross-tier fallback	~$3-15 per 1K calls
Scanning/monitoring	Sentinel, Trendy, Compliance, NeuroSec	`openrouter/xiaomi/mimo-v2-flash` or local Ollama 3B	Local Ollama	~$0.10-0.50 per 1K calls
Lightweight	Pitwall, Munich, Gambit, Mentor, Ledger, Digest	Local Ollama 3B or mimo-v2-flash	API if local unavailable	Minimal to zero

1.4 Critical Success Factors

Four factors determine v2.0 success, ranked by **dependency order**:

Factor	Target	Measurement
1. Telegram delivery reliability	**>95% uptime**	48-hour continuous successful delivery gate
2. Orchestrator delegation accuracy	**>80% correct routing**	BRUTUS classification → successful subagent completion

Factor	Target	Measurement
3. Morning briefing quality	**Flo-rated 8+/10**	Daily 06:45 CET delivery with fresh, actionable intelligence
4. Cost sustainability	**<$150/month incremental**	Sentinel-monitored OpenRouter spend with tier enforcement

Secondary factors include: NeuroSec baseline completion enabling actual security monitoring; WireGuard mesh restoration to 3+ active nodes; and **BBBank onboarding support delivery by March 1, 2026**—the immovable external deadline driving Finance Cluster acceleration.

---

2. Infrastructure Foundation

2.1 WireGuard Mesh Topology

2.1.1 Node Allocation Strategy The **10.0.0.0/24 WireGuard mesh** implements **purpose-heterogeneous design**—each node's hardware configuration directly determines its functional role, maximizing resource efficiency through hardware-function alignment. This topology provides encrypted, low-latency interconnectivity for distributed agent operations while maintaining clear failure domains.

Nexus (10.0.0.1, 1GB RAM) serves as **WireGuard hub and security bastion**. Its minimal RAM **intentionally excludes agent hosting**—attempting to run even lightweight subagents would risk OOM kills and gateway instability. Instead, its network position at the mesh hub enables efficient traffic inspection and **NeuroSec alert relay** without computational load. Current **SSH refusal** indicates likely **fail2ban self-lock or SSH daemon crash**, requiring provider console access for recovery.

Clawd-16gb (10.0.0.2) is the **primary compute node**, hosting the **OpenClaw Gateway, BRUTUS orchestrator, all 16 subagent configurations, cron scheduler, Telegram bot, and Ollama 3B models**. The 14GB free RAM (after ~2GB system/Gateway overhead) supports substantial concurrent API call buffering and local inference. The 434GB free disk accommodates extensive memory file logging, skill repositories, and archival. This concentration simplifies backup, migration, and debugging while maintaining theoretical scalability—subagent configs migrate to other nodes with only path updates if resource pressure demands.

Brutus-8gb (10.0.0.3) functions as **coding agent workstation and Ollama small-model host**. Its 8GB RAM supports **3B-7B parameter models** for offline inference when API connectivity fails or cost optimization requires. The **Cloud agent's remote SSH operations** target this node, enabling distributed server management without Gateway resource contention.

Plutos-32gb (10.0.0.4) provides **heavy inference endpoint capacity** for **14B+ parameter models** when restored from current offline status. This enables **local execution of reasoning-heavy tasks** without API dependency, providing cost reduction and data privacy benefits—particularly critical for **Banker agent's client-sensitive operations**.

2.1.2 Resource Distribution Across 4 Nodes

Node	IP Address	RAM	Status	Primary Function	v2.0 Agent Allocation
Nexus	10.0.0.1	1GB	🔴 SSH refused	WireGuard hub, security bastion, NeuroSec alerts	**None** (infrastructure only)
Clawd	10.0.0.2	16GB (14GB free)	✅ Online	**OpenClaw Gateway, BRUTUS, cron, Telegram, all subagents**	**BRUTUS (orch.), Macro, Alpha, Banker, Scribe, Sentinel, Trendy, Digest, Atlas, Compliance, Pitwall, Munich, Gambit, Mentor, Ledger, NeuroSec**
Brutus-8gb	10.0.0.3	8GB	✅ Online	Coding agent, Ollama small models (3B-7B)	**Cloud** (remote SSH ops target)
Plutos	10.0.0.4	32GB	🔴 Offline	Heavy inference, Ollama 14B+ models	**Inference endpoint only** (post-recovery)

The **87.5% agent concentration on clawd-16gb** contradicts distributed systems orthodoxy but aligns with **OpenClaw's architectural reality**: subagents are **configuration objects, not processes**. Each agent comprises approximately **50-100KB in SOUL.md, CONTEXT.md, and config.json files**—loaded on demand during session spawning, not kept resident. The actual compute load is **API call volume and response processing**, bounded by OpenRouter rate limits rather than local resources.

Ollama cluster tiering enables intelligent model selection:

Node	Models	Use Case	Latency
Clawd	3B (qwen2.5-coder:3b, phi4:3b)	Lightweight agent tasks, scanning fallback	~50-200ms
Brutus-8gb	7B (llama3.1:8b, qwen2.5:7b)	Intermediate reasoning, coding assistance	~200-500ms
Plutos (recovery)	14B+ (qwen2.5:14b, deepseek-r1:14b, llama3.3:70b)	Deep analysis, complex generation, privacy-critical	~500ms-2s

2.1.3 Failover and Recovery Procedures **Node failure scenarios** require differentiated responses based on **functional criticality**:

Failure Mode	Impact	Recovery Procedure	Time Target
Clawd-16gb (Gateway)	🔴 **Catastrophic** — all operations halt	Restore from backup or migrate Gateway to Brutus-8gb with DNS/bot token updates	30-60 min
Nexus SSH refusal	🟡 Security hub degraded, mesh routing impaired	Provider console access → check `/var/log/auth.log` → fail2ban whitelist or `systemctl restart sshd`	<30 min
Plutos offline	🟡 Heavy inference eliminated, API costs increase	Invoice payment → verify WireGuard handshake → restore Ollama models	<15 min post-payment
Brutus-8gb failure	🟡 Coding agent lost, SSH operations degrade to Clawd local	Queue Cloud operations for retry, no immediate user impact	N/A (degraded operation)

Recovery prioritization follows user impact: **Telegram delivery > Gateway availability > mesh completeness > heavy inference**. The Phase 0-1 sequence addresses critical path items before feature development, with explicit verification at each stage.

2.2 OpenClaw Gateway Configuration

2.2.1 Version Compatibility (v2026.2.6-3) The **v2026.2.6-3 Gateway** provides production-stable capabilities for v2.0 implementation: **subagent spawning via `sessions_spawn`**, **cron job persistence with isolated execution modes**, **multi-channel delivery with Telegram topic support**, and **memory-mapped agent configurations**. Version pinning is recommended—automatic updates risk introducing breaking changes during critical development phases.

Capability verification before Phase 1:

```
# Gateway health check

curl -s http://localhost:18789/health | jq .

# Subagent spawn test

openclaw agent spawn macro --task "Test subagent initialization" --dry-run
```

```
# Cron persistence verification

openclaw cron list # Should show existing jobs
```

The **upgrade path to v2026.3.x** (expected March 2026) should follow v2.0 stabilization, with isolated environment testing before production deployment.

2.2.2 Memory and Compute Budgeting **Memory model**: Subagent configurations load into Gateway memory **only during active execution**—persistent storage is disk-based. Concurrent execution, not agent count, determines RAM pressure.

Scenario	Estimated RAM	Notes
Gateway idle	512MB-1GB	Base process, no active sessions
BRUTUS main session	1.5GB	Context window, tool definitions, memory index
Single subagent spawn	+800MB	Temporary session, auto-archived post-completion
Three concurrent subagents (recommended max)	**4GB**	Headroom for API buffering
Cron job execution	+600MB	Isolated session, freed after delivery
Local Ollama 3B inference	+2GB	Model weights, active generation

The **4GB concurrent subagent limit** (three agents plus BRUTUS main) shapes delegation patterns: **serialize complex multi-agent queries** rather than parallelizing—"ask Macro, then ask Alpha, then synthesize" matches ADHD-aware sequential delivery preference while preventing resource exhaustion.

Compute budgeting emphasizes **API inference over local models**:

Workload Type	Preferred Execution	Cost Impact
Lightweight tasks (Sentinel checks, Trendy scans)	Local Ollama 3B	Zero API cost
Reasoning tasks (Macro analysis, Banker prep)	OpenRouter premium	~$3-15 per 1K calls

Workload Type	Preferred Execution	Cost Impact
Heavy generation (Atlas research, Scribe drafting)	OpenRouter with Plutos fallback	Variable, monitored

2.2.3 Ollama Cluster Integration **Ollama integration** provides **local inference fallback** for cost control, latency reduction, and privacy-sensitive operations. Configuration uses **OpenAI-compatible endpoint specification** with model aliases for consistent reference:

```
// ~/.openclaw/openclaw.json excerpt
{
  "models": {
    "local-3b": {
      "provider": "ollama",
      "baseUrl": "http://localhost:11434",
      "model": "qwen2.5-coder:3b"
    },
    "local-7b": {
      "provider": "ollama",
      "baseUrl": "http://10.0.0.3:11434",
      "model": "llama3.1:8b"
    },
    "local-14b": {
      "provider": "ollama",
      "baseUrl": "http://10.0.0.4:11434",
      "model": "qwen2.5:14b"
    }
  }
}
```

Agent config.json references these aliases—`"model": "local-3b"` resolves to appropriate endpoint. **Fallback chain**: if Plutos remains offline, "local-14b" references fail gracefully to API fallback; if Brutus-8gb unreachable, "local-7b" routes to API or local-3b.

Health monitoring via Sentinel's `ollama-check` cron (every 30 minutes) verifies API responsiveness across all nodes, with **automatic model tier adjustment**—agents configured for unavailable local models receive API fallback without manual intervention.

2.3 Telegram Delivery System

2.3.1 Bot Authentication and Session Management The **@brutusclawdbot Telegram delivery failures** ("chat not found" errors) represent the **single highest-priority blocker** for v2.0. Root cause analysis identifies **two probable failure modes**:

Failure Mode	Diagnostic	Resolution
Bot token invalid/expired	`curl "https://api.telegram.org/bot<TOKEN>/getMe"` returns `{"ok":false}`	Regenerate via @BotFather, update `~/.openclaw/credentials/telegram/bot_token`
Chat_id mismatch	`getMe` succeeds but `sendMessage` to stored chat_id fails	Flo sends `/start` to @brutusclawdbot, extract fresh chat_id from `getUpdates`

Critical discovery: Telegram bots **cannot initiate conversations**—user must send `/start` to establish chat context. Session persistence in `~/.openclaw/telegram/` may cache **stale chat_id values** from previous authorizations. **Nuclear recovery option**: `rm -rf ~/.openclaw/telegram/` forces fresh session establishment.

Session verification protocol:

```
# 1. Verify bot token

curl -s "https://api.telegram.org/bot${TOKEN}/getMe" | jq '.ok'

# 2. Check stored configuration

cat ~/.openclaw/openclaw.json | jq '.channels.telegram'

# 3. Retrieve current chat_id

curl -s "https://api.telegram.org/bot${TOKEN}/getUpdates" | \
  jq '.result[-1].message.chat.id'

# 4. Test direct delivery

openclaw send-message "Verification test $(date)" --chat @notabanker1
```

2.3.2 Chat ID Verification Protocol Systematic **chat_id verification** prevents recurrence and enables diagnostic automation. Implement **Sentinel health check integration**:

```
#!/bin/bash

# ~/.openclaw/skills/telegram-verify/check.sh

TOKEN=$(cat ~/.openclaw/credentials/telegram/bot_token)
CHAT_ID=$(jq -r '.channels.telegram.chat_id' ~/.openclaw/openclaw.json)

BOT_INFO=$(curl -s "https://api.telegram.org/bot${TOKEN}/getMe")
[[ $(echo "$BOT_INFO" | jq -r '.ok') != "true" ]] && \
  echo "🔴 TELEGRAM_AUTH_FAIL" && exit 1
```

```
SEND_RESULT=$(curl -s -X POST "https://api.telegram.org/bot${TOKEN}/sendMessage" \

  -d "chat_id=${CHAT_ID}" -d "text=Verification $(date +%H:%M:%S)")

[[ $(echo "$SEND_RESULT" | jq -r '.ok') != "true" ]] && \
  echo "🔴 TELEGRAM_SEND_FAIL: $(echo "$SEND_RESULT" | jq -r '.description')" && \
  echo "⚠️ Remediation: Flo must send /start to @brutusclawdbot" && exit 1

echo "✅ TELEGRAM_OK"
```

Multi-channel configuration supports future expansion: primary (Flo direct), secondary (monitoring channel for critical alerts), tertiary (log channel for debugging). Each requires explicit `/start` authorization.

2.3.3 Message Routing and Rate Limiting **Telegram Bot API rate limits** (30 messages/second to same chat, 20 messages/minute to same group) require **aggregation strategies** for 16-agent system:

Strategy	Implementation	Benefit
Per-agent message caps	Max 1 message per agent per 5-minute window to Flo	Prevents spam, respects attention
Digest aggregation	Morning briefing compiles 4+ agent outputs into single message	Channel usability, ADHD optimization
Priority queuing	🔴 Critical bypass aggregation; 🟡/🟢 respect quiet hours	Appropriate urgency routing
Length-based splitting	>4000 characters split with "1/3", "2/3", "3/3" headers	Technical compliance

Energy-aware delivery windows:

Window	Message Type	Rationale
00:00-12:00 CET	🔴 Critical alerts only	Flo's low-energy morning
12:00-16:00 CET	Light touches, batched updates	Transition to peak
16:00-02:00 CET	**Full engagement, complex outputs**	**Flo's peak cognitive window**
02:00+ CET	Queue for tomorrow, "go to bed" nudge	Sleep hygiene support

3. Agent Architecture Design

3.1 Orchestrator Pattern: BRUTUS v2.0

3.1.1 Role Transformation from Worker to Router The **BRUTUS v2.0 transformation** is the **most consequential architectural decision** in v2.0—shifting from **generalist task execution** to **specialist coordination** while **preserving established user relationship dynamics**. This pattern, deployed successfully in production systems from Microsoft's Copilot stack to financial trading infrastructure, separates **human relationship management** from **domain-specific execution**.

Capability surrender is explicit and comprehensive:

BRUTUS v1.0 (Worker)	BRUTUS v2.0 (Orchestrator)
Executes market analysis directly	**Classifies** → spawns **Macro/Alpha**
Drafts emails and documents	**Routes** → spawns **Scribe**
Monitors infrastructure status	**Aggregates** → spawns **Sentinel**
Prepares client meeting materials	**Delegates** → spawns **Banker**
Tracks personal reminders	**Coordinates** → spawns **Munich/Pitwall**

Retained capabilities: quick factual answers (time, calculations), **task routing decisions** (the classification logic itself), **daily digest compilation coordination**, and **personality maintenance** (the "best-buddy" relationship with Gen-Z slang and ADHD-aware communication).

Delegation workflow (five-step):

1. **Receive**: Flo sends message to BRUTUS via Telegram
2. **Classify**: Intent analysis against 16-agent registry
3. **Delegate**: `sessions_spawn` with appropriate context
4. **Collect**: Result aggregation from subagent
5. **Relay**: Formatted response with BRUTUS personality overlay

This introduces **15-30 second latency** (subagent spawn + API call + relay) versus direct execution, trading time for **expertise quality**—acceptable given non-urgent nature of most queries.

3.1.2 Delegation Logic and Classification Rules **Intent classification** implements **decision tree with confidence thresholds**:

Intent Pattern	Primary Agent	Confidence	Fallback	Multi-Agent Trigger
"ECB/Fed/rates/ policy"	**Macro**	0.95	BRUTUS direct	"How do rates affect my mortgage?" → Macro + Banker

Intent Pattern	Primary Agent	Confidence	Fallback	Multi-Agent Trigger
"DAX/stocks/ price/target"	**Alpha**	0.90	Macro	"European banks" → Alpha + Macro
"client/meeting/ prep/advice"	**Banker**	0.95	BRUTUS (privacy warning)	Sequential with Macro for context
"server/SSH/ down/error"	**Cloud**	0.90	Sentinel	Cloud execution, Sentinel monitoring
"email/draft/ write/send"	**Scribe**	0.85	BRUTUS (simple only)	Scribe draft → BRUTUS approval relay
"news/scan/ opportunity"	**Trendy**	0.80	Alpha	Trendy scan + Alpha deep-dive on flag
"race/Herberth/ F1/calendar"	**Pitwall**	0.95	BRUTUS	Dual calendar (GT3 + F1) tracking
"tax/Sparen/ finance personal"	**Ledger**	0.90	Banker (if BBBank-related)	Clear personal/professional boundary

Confidence handling:

- **>0.8**: Spawn single agent, proceed
- **0.5-0.8**: Spawn primary with secondary standby, or request clarification
- **<0.5**: Ask clarifying question—"Not sure if you want market analysis (Alpha) or macro context (Macro). Which angle?"

Multi-intent messages: "Prepare me for tomorrow's client meeting about the ECB rate decision" requires **sequential delegation with context passing**—Macro analysis feeds Banker prep, BRUTUS synthesizes final briefing. **Parallel spawning avoided** due to resource constraints and ADHD-preferential sequential presentation.

3.1.3 ADHD-Aware Communication Protocols **ADHD support is architectural requirement, not optional enhancement**. Flo's diagnosed ADHD creates specific interaction patterns that v2.0 must accommodate through **explicit operational rules**:

Pattern	Implementation	Example
Action-first responses	Lead with DO, then context	“**Hold steady on rate guidance** — ECB kept 3.75% as expected. Context: inflation sticky…”
Chunked outputs	Max 5 bullet points, “Want more?” continuation	Break at 5, explicit prompt for expansion
External memory as compensation	Automatic task_log.md logging, proactive deadline surfacing	“You said you’d review BBBank docs by Friday — that’s tomorrow”
No repetition penalty	Fresh answers, memory check for changed context	Never “as I mentioned” — just answer
Deadline surfacing	Countdown format with multi-stage reminders	“Due in 7 days” → 3 days → 1 day → day-of
Decision scaffolding	Max 3 options, explicit recommendation	“I’d go with B because…”
Energy-aware scheduling	Critical-only before noon, full engagement 4pm-2am	Queue non-urgent, “go to bed” after 2am
Capture everything	Passing mentions logged as tasks	“Oh I need to call my mom” → auto-logged

BRUTUS-specific ADHD section in SOUL.md:

```
## Your Job as External Brain

Flo has ADHD. You are his external working memory and executive function scaffold.

ALWAYS:

- Log commitments to task_log.md with deadline calculation

- Surface deadlines proactively ("Hey, that BBBank thing is in 3 days")

- Gently redirect when scattered: "You were working on X — continue or switch?"

- Maintain "parking lot" for half-finished thoughts

NEVER:
```

```
- Long unstructured walls of text

- Expect recall from 3+ conversations ago — check memory

- "As I mentioned earlier" phrasing

- Non-urgent messages before noon CET

ENERGY-AWARE SCHEDULING:

- Before 12:00: Only 🔴 CRITICAL (system down, security, money)

- 12:00-16:00: Light touches, queued updates

- 16:00-02:00: Full engagement, complex tasks

- After 02:00: Queue all, "go to bed, bro" if active at 3am
```

3.2 Cluster Organization

3.2.1 Finance Cluster: Macro, Alpha, Banker, Ledger The **Finance Cluster** delivers **highest business value** and **most urgent development priority** due to **March 1, 2026 BBBank deadline**. Four agents cover distinct temporal and functional domains with **accelerated onboarding support**:

Agent	Domain	Key Capability	BBBank Relevance
Macro	Central bank policy, monetary regimes	Daily 05:30 scan, regime assessment, event anticipation	**Rate environment for product positioning**
Alpha	Asset-specific analysis, market research	Watchlist management, valuation metrics, catalyst tracking	**European bank stocks, client portfolio context**
Banker	Client advisory, meeting preparation	**Onboarding mode**: DZ Bank ecosystem, cooperative model, regulatory framework	**Direct professional enablement**
Ledger	Personal finance, tax optimization	Sparerpauschbetrag tracking, quarterly reviews, goal monitoring	**Personal/professional boundary maintenance**

Banker agent's ONBOARDING MODE (February 2026 priority):

Day	Task	Output
1-3	BBBank eG research: history, cooperative model, Munich presence	CONTEXT.md foundation
3-5	DZ Bank ecosystem mapping: Union Investment, R+V, Schwäbisch Hall, DZ HYP, TeamBank, VR Smart Finanz	Product universe table
5-7	Quick-reference cards: key facts, fees, risk levels, suitable profiles	RUNBOOK.md procedures
7-10	Regulatory research: § 64 WpHG Anlageberatung, Geeignetheitserklärung, MiFID II suitability	Compliance integration
10-14	"First Week Survival Kit": contacts template, system checklist, compliance must-knows	Scribe-formatted document
Ongoing	Daily 18:00 "BBBank Product Fact" via Telegram	Accelerated knowledge transfer

Macro supports onboarding: "Here's current ECB rate environment and what it means for products you'll sell: Bauspar rates, mortgage rates, bond fund outlook, Tagesgeld money market rates." This gives Flo **pre-built mental model** walking into BBBank March 1.

3.2.2 Operations Cluster: Sentinel, Cloud, Compliance

Agent	Function	Replaces v1.0	Key Capabilities
Sentinel	Unified infrastructure monitoring	`hourly-mesh-confirmation` + `hourly-newsclawd-update`	15-min health checks, Ollama verification, cost tracking, crypto monitoring, severity-graded alerting
Cloud	Remote server operations	Manual SSH	SSH to mesh nodes, read-only default, write-with-approval, operation logging

Agent	Function	Replaces v1.0	Key Capabilities
Compliance	Regulatory monitoring	None (new)	Weekly BaFin/ESMA scan, Friday 16:00 digest, MiFID II/ KWG tracking

Sentinel cron schedule (replaces broken v1.0 jobs):

Job	Schedule	Scope	Severity Handling
`sentinel-health`	`*/15 * * * *`	Mesh pings, process checks, memory/disk	🔴 immediate, 🟡 digest, 🟢 log
`sentinel-ollama`	`*/30 * * * *`	Ollama API health all nodes	Degradation alert, fallback trigger
`sentinel-costs`	`0 */4 * * *`	OpenRouter spend tracking	200% baseline → critical alert
`sentinel-crypto`	`0 */2 * * *`	BTC/ETH + ArbitrageAndy status	Threshold breach alerts
`sentinel-daily-summary`	`0 20 * * *`	Aggregated infrastructure report	Evening digest inclusion

3.2.3 Content Cluster: Scribe, Trendy, Atlas

Agent	Function	Integration	Key Constraint
Scribe	Communications, document drafting	**Gmail OAuth2** (Phase 4a read-only, 4b send-with-approval)	**Never auto-send —human-in-the-loop mandatory**
Trendy	Market opportunity scanning	Polymarket, crypto, AI news, 90-min intervals	Replaces NewsClawd with broader scope
Atlas	Deep research, strategic analysis	On-demand BRUTUS delegation, 20-min time-boxed	Progress updates every 2 min to prevent abandonment

Scribe's phased Gmail integration protects professional liability: Phase 4a enables inbox triage and summarization; Phase 4b adds draft composition with **explicit Flo approval before any send operation**. OAuth2 credential reuse from existing clawdbot accelerates setup.

3.2.4 Life Cluster: Pitwall, Munich, Gambit, Mentor

Agent	Domain	Frequency	Integration
Pitwall	Motorsport logistics (Herberth GT3 + F1)	3-week, 1-week, 3-day, day-before race alerts	Morning briefing, vacation planning coordination
Munich	Local life management	Weekly cron, seasonal triggers	Apartment reminders, administrative deadlines, local events
Gambit	Chess training	Daily puzzle delivery	Lichess API, morning briefing inclusion, rating tracking
Mentor	Learning roadmap	Weekly Sunday 19:00 review	Activity pattern analysis, skill gap identification, resource recommendation

These agents operate at **lower frequency and complexity**, utilizing **lightweight models** (Mimo-v2-Flash or local Ollama 3B) for cost efficiency. Their value is **automated life management reducing cognitive overhead** rather than professional enablement.

3.2.5 Security Layer: NeuroSec **NeuroSec upgrades from v1.0 placeholder to active monitoring** through **baseline completion and enhanced detection**:

Baseline	Generation Command	Detection Scope
`permissions.json`	`find /home/boss/.openclaw -type f -exec stat -c '%n %a %U:%G' {} \;`	File permission changes, ownership drift, unexpected additions
`network.json`	`ss -tlnp | jq -R 'split(" ")| {port: .[3], process: .[6]}'`	New listening ports, missing expected services, process changes
`secrets.json`	`sha256sum ~/.openclaw/credentials/*`	Credential file modification (hash-only, no plaintext storage)

6-hour scan cycle with severity classification: **critical** (immediate Telegram), **warning** (Digest aggregation), **info** (log-only). Clinical persona maintained—factual, structured, actionable: "PERMISSION

ANOMALY: ~/.openclaw/openclaw.json mode 644 (expected 600). RECOMMENDED ACTION: chmod 600 ~/.openclaw/openclaw.json."

3.2.6 Meta Layer: Digest **Digest is the critical integration point**—the "killer feature" demonstrating v2.0's distributed intelligence value. **06:45 daily execution** compiles cross-cluster outputs into **ADHD-optimized morning briefing**:

Source Agent	Content	Freshness Requirement
Macro	Overnight markets, central bank news	<6 hours (05:30 scan)
Sentinel	System health, API costs	<30 minutes (15-min checks)
Pitwall	Race weekend proximity	<24 hours
Munich	Daily reminders	<12 hours

TEMPLATE.md enforces strict format:

```
☀️ GUT'N MORGEN FLO — {{DAY}}, {{DATE}}

📅 HEUTE

- {{calendar_items}}

- {{reminders_from_munich}}

- {{race_weekend_alert_from_pitwall}}

💰 MÄRKTE OVERNIGHT

- DAX: {{level}} ({{change}}) | S&P: {{level}} ({{change}})

- EUR/USD: {{level}} | 10Y Bund: {{yield}} | 10Y UST: {{yield}}

- BTC: {{price}} ({{change}}) | ETH: {{price}} ({{change}})

- {{macro_headline_if_any}}

🏦 FÜR DIE ARBEIT

- {{client_meetings_today}}

- {{macro_talking_points}}

- {{compliance_alerts_if_any}}

🤖 SYSTEME

- Mesh: {{mesh_status}} | Ollama: {{ollama_status}}
- API costs yesterday: ${{amount}}

- {{critical_alerts_if_any}}

📈 OPPORTUNITIES

- {{top_trendy_finding}}

- {{polymarket_notable}}

♟ {{daily_puzzle_from_gambit}}

🎯 TOP 3 FÜR HEUTE
1. {{priority_1}}
2. {{priority_2}}
3. {{priority_3}}
```

Design rules: <2 minute read, **bold action items at top**, numbered/bulleted only, 🔴/🟡/🟢 urgency coding, **exactly 3 priorities**.

3.3 Subagent Configuration Schema

3.3.1 Directory Structure Standardization All 16 agents follow **identical directory structure**:

```
~/.openclaw/workspace/agents/{agent_name}/
   SOUL.md # Required: personality, principles, constraints
   config.json # Required: model, tools, permissions, cron
   CONTEXT.md # Optional: persistent state, current focus
   RUNBOOK.md # Optional: operational procedures, SOPs
   TEMPLATE.md # Optional: output formatting (Digest only)
```

Provisioning command:

```
mkdir -p ~/.openclaw/workspace/agents/{macro,alpha,banker,sentinel,scribe,trendy,cloud,neurosec,
    digest,atlas,compliance,pitwall,munich,gambit,mentor,ledger}
```

Parallel **memory structure**:

```
~/.openclaw/workspace/memory/agents/{agent_name}/
   {domain_specific_files}.md # Persistent state
   {domain_specific_files}.json # Structured data
```

3.3.2 config.json Specification

Field	Type	Required	Description	Example
`name`	string	Yes	Machine identifier	`"macro"`
`display_name`	string	Yes	Human-readable label	`"MACRO — Central Bank Analyst"`
`model`	string	Yes	Primary model	`"openrouter/moonshotai/kimi-k2.5"`
`model_fallback`	string	No	Degradation target	`"openrouter/xiaomi/mimo-v2-flash"`
`tools`	string[]	Yes	Enabled capabilities	`["web_search", "file_read", "file_write"]`
`memory_path`	string	Yes	Base for agent memory	`"memory/agents/macro/"`
`permissions`	object	Yes	Fine-grained access control	See below
`cron`	object[]	No	Scheduled tasks	See below
`security`	object	No	Agent-specific constraints (Banker)	PII rules

Permissions object:

```
{
  "web_search": true,
  "file_read": ["memory/agents/macro/*", "memory/global/*"],
  "file_write": ["memory/agents/macro/*"],
  "telegram_send": false,
  "ssh": false,
  "api_calls": ["coinbase", "coingecko"]
}
```

Cron entry structure:

```
{
  "name": "macro-daily-scan",
  "schedule": "30 5 * * *",
  "task": "Check central bank calendar...",
  "delivery": "memory_only"
}
```

`delivery` options: `memory_only` (silent), `telegram_via_brutus` (alert), `digest_contribution` (flag for compilation).

3.3.3 SOUL.md Personality Framework **Required sections**: Identity, Purpose, Principles, Constraints, Communication Style. **ADHD Support section** mandatory for all agents, with **BRUTUS-specific delegation logic**.

Macro SOUL.md excerpt:

```

# MACRO — Central Bank Analyst

## Identity

You track global monetary policy with obsessive precision.
You think in regimes: tightening, easing, pause, and transitions.

## Purpose

Provide Flo with actionable understanding of monetary policy trajectory
and its implications for asset prices, economic activity, and BBBank decisions.

## Principles

- Lead with regime assessment

- Quantify uncertainty (facts vs pricing vs judgment)

- Connect to Flo's context (BBBank products, client conversations)

## Constraints

- Never predict with false precision

- Distinguish your analysis from consensus

- Flag data limitations explicitly

## Communication

- Professional but accessible

- Numbers-first, then interpretation

- 3-5 paragraphs standard, expandable on request

## ADHD Support

- Chunked regime summaries

- Explicit "what this means for you" translation

- Calendar countdowns for key events
```

3.3.4 CONTEXT.md State Management **Agent-self-maintained persistent state** enabling longitudinal tracking:

Agent	Key CONTEXT.md Sections
Macro	Current regime assessment, this week's calendar, active monitoring topics, recent surprises
Banker	BBBank onboarding progress, product knowledge gaps, anonymized meeting patterns, certification checklist
Sentinel	Threshold configurations, alert history with resolution, known false positives, cost tracking trends

Update protocols: cron execution appends, BRUTUS delegation with explicit "update context" instruction, self-reflection on task completion.

3.3.5 RUNBOOK.md Operational Procedures **Complex agents only**—documenting multi-step workflows:

Agent	Key RUNBOOK.md Procedures
Macro	Rate Decision Response (5-min immediate, 15-30 min analysis), Regime Assessment Update
Banker	Client Meeting Preparation (agenda, research, objections, follow-up), Product Inquiry Handling
Sentinel	Alert Escalation Protocol, Threshold Tuning Procedure, Cost Anomaly Investigation
Cloud	Incident Response (diagnose, isolate, remediate, verify, document), SSH Safety Procedures

---

4. Model Strategy and Cost Optimization

4.1 Tiered Model Selection

4.1.1 Reasoning-Heavy Tier (Claude Sonnet, Kimi K2.5)

Agent	Primary	Fallback	Est. Daily Calls	Daily Cost
BRUTUS	`claude-sonnet-4-20250514`	`kimi-k2.5`	50	~$4.00
Banker	`kimi-k2.5`	`claude-sonnet-4-20250514`	10	~$2.00
Macro	`kimi-k2.5`	`mimo-v2-flash`	15	~$2.25
Alpha	`kimi-k2.5`	`mimo-v2-flash`	15	~$2.25
Atlas	`claude-sonnet-4-20250514`	`kimi-k2.5`	5	~$2.50
Scribe	`kimi-k2.5`	`mimo-v2-flash`	10	~$1.75

Total reasoning-tier estimate: ~$14.75/day, ~**$445/month** at full utilization. Actual lower due to fallback triggering and query batching. **Kimi K2.5 as default** balances capability and cost (~33% savings vs Claude Sonnet).

4.1.2 Scanning/Monitoring Tier (Mimo-v2-Flash, Ollama 3B)

Model	Cost	Use Case	Agents
`mimo-v2-flash`	~$0.10-0.50/1K calls	High-frequency scanning, structured extraction	Sentinel, Trendy, Compliance
`ollama/qwen2.5-coder:3b`	Zero (compute only)	Private data processing, deterministic tasks	NeuroSec, Sentinel (fallback)

10x cost reduction enables aggressive scheduling: 15-minute Sentinel checks, 90-minute Trendy scans without budget impact.

4.1.3 Lightweight Tier (Local Ollama Models)

Agent	Model	Task	Cost
Digest	`phi3:mini` or `qwen2.5:0.5b`	Template compilation, data aggregation	~$0-0.05
Pitwall, Munich, Gambit, Mentor, Ledger	`qwen2.5-coder:3b`	Reminders, puzzles, tracking	$0 (local)

Fallback to mimo-v2-flash on Ollama unavailability maintains service continuity.

4.2 OpenRouter Integration

4.2.1 API Key Management

```
// ~/.openclaw/openclaw.json
{
 "env": {
   "OPENROUTER_API_KEY": "sk-or-v1-..."
 },
 "models": {
   "providers": {
     "openrouter": {
       "baseUrl": "https://openrouter.ai/api/v1",
       "apiKey": "${OPENROUTER_API_KEY}",
       "models": [
         {"id": "anthropic/claude-sonnet-4-20250514", "alias": "sonnet"},
         {"id": "moonshotai/kimi-k2.5", "alias": "kimi"},
         {"id": "xiaomi/mimo-v2-flash", "alias": "mimo"}
       ]
     }
   }
 }
}
```

Security: Key in environment variable enables rotation without config modification. **OpenRouter dashboard**: monthly spend caps, per-model rate limits, IP allowlisting.

4.2.2 Fallback Chain Configuration

Tier	Primary	Fallback 1	Fallback 2	Final
Reasoning	Kimi K2.5	Claude Sonnet	Local Ollama 7B	Explicit failure + BRUTUS notification
Scanning	Mimo-v2-flash	Local Ollama 3B	Cached last result	Stale warning
Lightweight	Local Ollama 3B	Mimo-v2-flash	—	BRUTUS degraded service notice

Fallback triggers: HTTP 429/5xx, >30s timeout, content policy rejection. Events logged to `memory/cost_tracker.md` for pattern analysis.

4.2.3 Cost Tracking and Alerting

Sentinel cost-check cron (every 4 hours):

Threshold	Alert	Action
50% daily budget ($10)	Warning in Digest	Monitor trend
80% daily budget ($16)	Immediate Telegram	Investigate anomaly

Threshold	Alert	Action
100% daily budget ($20)	Critical + model tier downgrade	Switch to local inference only
200% baseline (weekly)	Emergency review	Full model rebalancing

---

5. Memory and Persistence Architecture

5.1 Shared Memory Layer

5.1.1 Global State: mesh_status, agent_registry, task_log

File	Maintainer	Update Frequency	Purpose
`mesh_status_latest.json`	Sentinel	15 minutes	Current node health, Ollama status, latency
`agent_registry.md`	BRUTUS	On change	Agent list, capabilities, status for delegation
`task_log.md`	BRUTUS	Real-time	Commitments, deadlines, completions for ADHD support
`cost_tracker.md`	Sentinel	4 hours	API spend, projections, optimization opportunities

5.1.2 Per-Agent Memory Isolation Each agent's `memory_path` contains domain-specific persistent state:

Agent	Key Memory Files
Macro	`regime_assessment.md`, `central_bank_calendar.md`, `rate_history.md`
Banker	`meeting_notes.md` (anonymized), `product_knowledge.md`, `onboarding_progress.md`

Agent	Key Memory Files
Sentinel	`alert_history.md`, `thresholds.json`, `cost_trends.md`
Pitwall	`race_calendar_2026.md` (GT3 + F1), `travel_bookings.md`

5.1.3 Cross-Agent Read Permissions **Digest's unique broad read access**: `"file_read": ["memory/agents/*/"]` enables morning briefing compilation. **Write isolation enforced**: agents modify only own memory, preventing cross-contamination.

5.2 File-Based Persistence

5.2.1 JSON State Files Machine-readable, schema-versioned: `mesh_status_latest.json`, `thresholds.json`, `secrets.json`. Pretty-printed for emergency human inspection.

5.2.2 Markdown Knowledge Bases Human-readable, Git-friendly: all `.md` files. `~/.openclaw/workspace/` should be **Git-initialized** with `.gitignore` for `credentials/`, `alerts/`.

5.2.3 Log Rotation and Archival

Data Type	Retention	Rotation Trigger
Daily logs	30 days	Size >10MB or age >7 days
Compressed archives	90 days	Weekly cron
Deep archival	1 year	Monthly to off-mesh storage

5.3 NeuroSec Baseline System

Baseline	Generation	Update Frequency	Alert Condition
`permissions.json`	`find /home/boss/.openclaw -type f -exec stat -c '%n %a %U:%G' {} \;`	Weekly + after intentional changes	Permission change, new file in sensitive path, ownership drift
`network.json`	`ss -tlnp | jq ...`	Every 6 hours	New listening port, missing expected service, process change
`secrets.json`	`sha256sum ~/.openclaw/credentials/*`	Weekly + after rotation	Hash mismatch (modification), file disappearance, unexpected new credential file

6. Cron Automation Framework

6.1 Scheduling Philosophy

6.1.1 Timezone-Aware Execution (CET) All schedules use `Europe/Berlin` (CET/CEST automatic). Critical schedules avoid 02:00-03:00 window (DST transition ambiguity).

6.1.2 Offset Strategy to Prevent Contention

Minute	Jobs	Rationale
:00	(reserved—lightest only)	Base tick, minimal load
:05, :10	(buffer for expansion)	Post-:00 recovery
:15, :30, :45	`sentinel-health`	Regular distribution
:30, :35	`macro-scan`, `alpha-scan`	Pre-briefing data gathering
:45	`morning-briefing` (06:45 only)	Compilation after sources complete

No two jobs share minute marks—eliminates v1.0 contention.

6.1.3 Energy-Aware Windows (Flo's 4pm-2am Peak)

Window	Job Types	Examples
00:00-12:00	Background monitoring only	sentinel-health, sentinel-ollama, sentinel-crypto, sentinel-costs
12:00-16:00	Light touches, non-urgent	munich-reminders, pitwall-calendar-check
16:00-02:00	**Full engagement, complex outputs**	**BRUTUS delegation, Atlas research, Scribe drafting**
02:00-00:00	Queue for tomorrow, sleep prompts	evening-digest, "go to bed, bro" if active at 03:00

6.2 Job Categories

6.2.1 High-Frequency Monitoring (15-min intervals) 'sentine

Clawd/Brutus v2.0 — Production Implementation Guide

1. Executive Summary

1.1 Project Overview

The **Clawd/Brutus v2.0 initiative** represents a fundamental architectural transformation of Flo's personal AI infrastructure, evolving from a single monolithic agent (BRUTUS v1.0) to a **16-agent orchestration system** built on OpenClaw's subagent framework. This project, dated **February 12, 2026**, and owned by Flo (@notabanker1), addresses critical operational limitations while establishing a scalable foundation for autonomous task delegation, specialized domain expertise, and ADHD-aware human-AI collaboration.

The transformation's core innovation is the **"Jarvis-style" orchestrator pattern**: BRUTUS evolves from performing all tasks directly to exclusively **classifying, delegating, and relaying** to specialized subagents organized across **six functional clusters**—Finance, Operations, Content, Life, Security, and Meta. This design preserves Flo's established relationship with BRUTUS (single Telegram interface, consistent personality, trusted communication style) while dramatically expanding cognitive capabilities through invisible specialization.

The implementation leverages **OpenClaw's lightweight subagent architecture**, where agents exist as **configuration bundles** (prompts + memory paths + tool permissions, not separate processes), enabling dense consolidation on the existing **clawd-16gb node with 14GB free RAM and 434GB available disk**. LLM inference occurs primarily via **OpenRouter API calls** rather than local execution, with intelligent tiering optimizing cost-quality tradeoffs.

A **critical external deadline** compresses the timeline: Flo begins as **Privatkundenberater at BB-Bank eG in Munich on March 1, 2026**—only **17 days from project inception**. This necessitates aggressive prioritization, with the **Finance Cluster accelerated ahead of strict architectural phase ordering** to deliver functional onboarding support. The production plan balances this urgency against foundational stability requirements, enforcing a **48-hour Telegram delivery reliability gate** before any architectural expansion.

1.2 Current State Assessment

The **v1.0 infrastructure** presents a mixed operational picture with **critical blockers requiring immediate Phase 0 remediation**:

Component	Status	Operational Impact
OpenClaw Gateway (clawd-16gb, v2026.2.6-3)	✅ Stable	Foundation for v2.0 expansion
BRUTUS main agent	✅ Functional	Daily driver, but monolithic bottleneck
NeuroSec security agent	⚠️ **Degraded**	**Running without baselines—blind to anomalies**
WireGuard mesh	⚠️ **3/4 nodes**	Nexus SSH refused, Plutos-32gb offline

Component	Status	Operational Impact
Ollama cluster	⚠️ **2/3 nodes**	No heavy inference (14B+ models)
Cron jobs (2 active)	⚠️ **Broken delivery**	Mesh check + NewsClawd both fail
Telegram delivery	🔴 **Critical failure**	**"Chat not found" errors block all automation**
Skills system	✅ 15+ structured	Mature tooling for extension

Six blockers cascade by dependency:

Blocker	Severity	Fix Required Before	Root Cause / Resolution
Telegram delivery failing	🔴 **Critical**	**Phase 0 (everything)**	Bot session/auth mismatch—requires `/start` from Flo, chat_id verification
Plutos-32gb offline	🟡 Medium	Phase 2	Unpaid invoice—payment restores heavy inference
Nexus SSH refused	🟡 Medium	Phase 1	Likely fail2ban self-lock or SSH daemon crash—provider console access required
NeuroSec baselines missing	🟡 Medium	Phase 1	Never generated—create permissions.json, network.json, secrets.json
Cron contention at :00	🟡 Medium	Phase 1	Resource competition—offset scheduling to :05, :10, :15, etc.
No subagent architecture	🟡 High	Phase 2	Core v2.0 transformation—BRUTUS refactor, 16-agent directory structure

The **Telegram delivery failure** is the **absolute prerequisite**—without reliable notification channels, no automation delivers value regardless of agent sophistication. The **48-hour stability gate** (consecutive successful mesh status and NewsClawd delivery) enforces this discipline before Phase 1 commencement.

1.3 Target Architecture

The **v2.0 target** implements a **hub-and-spoke orchestration topology** with BRUTUS as the exclusive human-facing interface:

```
Flo → Telegram → BRUTUS (orchestrator ONLY — "Jarvis" role)

                    **FINANCE CLUSTER**  → Macro, Alpha, Banker, Ledger
                    **OPS CLUSTER**    → Sentinel, Cloud, Compliance
                    **CONTENT CLUSTER** → Scribe, Trendy, Atlas
                    **LIFE CLUSTER**   → Pitwall, Munich, Gambit, Mentor
                    **SECURITY**     → NeuroSec (upgraded with baselines)
                    **META**       → Digest (morning briefing compiler)
```

Node allocation concentrates agent execution on **clawd-16gb** while distributing specialized workloads:

Node	IP	RAM	Function	Agent Hosting
Nexus	10.0.0.1	1GB	WireGuard hub, security bastion	**None** (infrastructure only)
Clawd	10.0.0.2	16GB (14GB free)	**OpenClaw Gateway, all subagents, cron, Telegram**	**All 16 agents + BRUTUS orchestrator**
Brutus-8gb	10.0.0.3	8GB	Coding agent, Ollama small models	Cloud (remote SSH ops)
Plutos	10.0.0.4	32GB	Heavy inference endpoint	Inference-only (post-recovery)

This concentration is **architecturally sound**: OpenClaw subagents consume ~**50-100MB RAM per configuration** (not resident processes), with LLM inference via **OpenRouter API calls**. The 14GB free RAM provides substantial headroom for concurrent API buffering and lightweight Ollama 3B fallback.

Model tiering strategy optimizes cost-quality tradeoffs:

Tier	Agents	Primary Model	Fallback	Cost Target
Reasoning-heavy	BRUTUS, Banker, Macro, Alpha, Atlas, Scribe	`openrouter/anthropic/claude-sonnet-4-20250514` or `kimi-k2.5`	Cross-tier fallback	~$3-15 per 1K calls

Tier	Agents	Primary Model	Fallback	Cost Target
Scanning/ monitoring	Sentinel, Trendy, Compliance, NeuroSec	`openrouter/xiaomi/mimo-v2-flash` or local Ollama 3B	Local Ollama	~$0.10-0.50 per 1K calls
Lightweight	Pitwall, Munich, Gambit, Mentor, Ledger, Digest	Local Ollama 3B or mimo-v2-flash	API if local unavailable	Minimal to zero

1.4 Critical Success Factors

Four factors determine v2.0 success, ranked by **dependency order**:

Factor	Target	Measurement
1. Telegram delivery reliability	**>95% uptime**	48-hour continuous successful delivery gate
2. Orchestrator delegation accuracy	**>80% correct routing**	BRUTUS classification → successful subagent completion
3. Morning briefing quality	**Flo-rated 8+/10**	Daily 06:45 CET delivery with fresh, actionable intelligence
4. Cost sustainability	**<$150/month incremental**	Sentinel-monitored OpenRouter spend with tier enforcement

Secondary factors include: NeuroSec baseline completion enabling actual security monitoring; WireGuard mesh restoration to 3+ active nodes; and **BBBank onboarding support delivery by March 1, 2026**—the immovable external deadline driving Finance Cluster acceleration.

---

2. Infrastructure Foundation

2.1 WireGuard Mesh Topology

2.1.1 Node Allocation Strategy The **10.0.0.0/24 WireGuard mesh** implements **purpose-heterogeneous design**—each node's hardware configuration directly determines its functional role, maximizing resource efficiency through hardware-function alignment. This topology provides encrypted, low-latency interconnectivity for distributed agent operations while maintaining clear failure domains.

Nexus (10.0.0.1, 1GB RAM) serves as **WireGuard hub