Module 10

Security Hardening

🦞 Sandboxing, Tool Policies, and Incident Response

🧭 Navigation Chart

By the end of this module, you will be able to:

1. **Explain** the three sandbox modes and three sandbox scopes
2. **Set up Docker** for sandboxing on WSL2
3. **Configure tool policies** to restrict what your agent can do
4. **Harden** your ⛵ gateway authentication
5. **Isolate** your agent on a separate home network
6. **Set up** a 🔑 password manager for credential storage
7. **Run and interpret** a deep security audit
8. **Implement** a weekly security maintenance routine
9. **Handle** a security incident step by step

📜 Ship's Logbook

Term	Definition
Sandbox	An isolated environment that restricts what a program can access
Docker	A platform for running applications in isolated containers
Container	A lightweight, isolated instance of an operating system
Tool policy	A rule that allows or denies specific tools for specific contexts
Elevated mode	A setting that bypasses sandboxing -- extremely dangerous
chmod	A 🐚 Linux command that changes file permissions

Why Harden Now?

- Module 01 was the **security briefing**
- Module 03 was **locking the front door**
- This module is installing the **alarm system, cameras, and vault**

Since initial setup, you have added:

- Messaging channels (Telegram, Discord, WhatsApp)
- 🐠 Skills from ClawHub and custom-built
- Heartbeats and cron jobs
- Potentially browser control

Each one expanded your attack surface. Time to contain the risk.

The Three Sandbox Modes

Mode	What It Does	When to Use
off	No sandboxing -- agent runs commands directly on your system	Only if you fully trust your setup
non-main	Everything except your main session is sandboxed	Good default for most users
all	Everything is sandboxed, including your main session	Maximum security for untrusted content

```
openclaw config sandbox mode non-main
```

What "non-main" means in practice:

- **Your TUI chat:** Full system access
- **Telegram messages:** Sandboxed (restricted)

The Three Sandbox Scopes

Scope	Container Behavior	Isolation Level	Overhead
session	New container for every session	Highest -- nothing persists	High
agent	One container per agent	Medium -- each agent isolated	Medium
shared	All agents share one container	Lowest -- agents can see each other's files	Low

```
openclaw config sandbox scope agent
```

Recommended: `agent` scope -- each agent gets its own sandbox without per-session overhead.

🪸 Workspace access control:

Setting	What the Sandbox Can Do
read/write	Read and write workspace files
read	Read files only

Setting Up Docker for Sandboxing

Step 1: Install Docker in WSL2

```
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io
```

Step 2: Add your user to the Docker group

```
sudo usermod -aG docker $USER
```

Then log out and back in.

Step 3: Verify it works

```
docker run hello-world
```

Step 4-5: Configure 🦞 OpenClaw sandboxing

Testing the Sandbox

From Telegram (or any external channel), send:

```
List the files in my home directory.
```

If sandboxing is working:

The agent should only see the sandbox's limited file system, not your actual home directory.

Expected response:

```
I can only see files within my sandbox environment.
I don't have access to your actual home directory.
```

If it shows your real files, sandboxing is **not active** -- recheck your configuration.

Tool Policies

Tool policies are layered (global > provider > agent > sandbox) and deny always wins.

Tool	What It Does	Risk Level
exec	Execute 🐚 shell commands	Critical
process	Manage system processes	High
browser	Control a web browser	High
file-read	Read files on the system	Medium
file-write	Write/modify files	High
network	Make network requests	Medium

Example: Lock down external channels

```
openclaw config tools deny exec --context channels
openclaw config tools deny browser --context channels
```

🚩 Rough Waters: Elevated Mode

Elevated exec bypasses all sandboxing. The agent runs commands directly on the host system regardless of sandbox config.

NEVER enable elevated mode for:

- Unknown senders
- External channels
- Untrusted 🐠 skills
- Any context where input is not directly from you

Check and disable:

```
# Check status
openclaw config tools elevated

# Disable if enabled
```

⛵ Gateway Authentication Hardening

Verify authentication is required:

```
openclaw config gateway auth
```

Rotate your ⛵ gateway token (periodic maintenance or after incident):

```
openclaw config gateway token rotate
```

Verify bind configuration:

```
openclaw config gateway bind
```

Expected: `loopback` (unless you intentionally configured LAN or Tailscale).

```
# Reset to loopback if unexpected
openclaw config gateway bind loopback
```

Home Network Security

Your agent shares WiFi with your laptop, phone, NAS, and smart TV. **A flat network means a compromise on one device can reach all others.**

The fix: **put your agent on a separate network** (same principle as IoT devices).

Approach	Cost	Difficulty	Isolation Level
Guest network	Free	Easy	Good -- devices can't see each other
VLAN / separate SSID	Free (if router supports it)	Medium	Best -- full network separation
Second cheap router	$15-30	Easy	Good -- physical subnet separation
Do nothing	Free	None	None -- all devices share one network

Guest Network (Easiest -- 5 minutes):

1. Log into your router admin panel (`192.168.1.1`)
2. Enable **Guest Network** with a strong password

🔑 Credential Management: Password Manager

🦞 OpenClaw stores secrets in **plaintext on disk** by default -- API keys, bot tokens, service credentials.

1Password Setup (Recommended):

1. Create a dedicated vault: **"Shared with OpenClaw"**
2. Create a **service account** with access to ONLY that vault
3. Add 1Password CLI commands to `TOOLS.md`
4. Tell the agent: **"Never store secrets in 🪸 memory, notes, or plain text"**

Why it matters:

- Compromised filesystem → attacker gets ONE service account key for ONE vault
- Credentials never appear in session transcripts or chat logs
- **Revoke instantly** -- disable the service account and all credential access stops

File Permissions

Lock down the workspace:

```
chmod 700 ~/.openclaw
chmod 700 ~/.openclaw/workspace
chmod 700 ~/.openclaw/config
chmod 700 ~/.openclaw/memory
chmod 700 ~/.openclaw/sessions
find ~/.openclaw -type f -exec chmod 600 {} \;
```

What these permissions mean:

Permission	Number	Meaning
`700` (directory)	rwx------	Only owner can read, write, and list
`600` (file)	rw-------	Only owner can read and write

Verify:

🔭 Running Security Audits

Run the deep audit:

```
openclaw security audit --deep
```

Understanding the output:

- CRITICAL -- must fix immediately (e.g., ⛵ gateway exposed without auth)
- WARNING -- address soon (e.g., file permissions too broad)
- INFO -- confirmations that things are correct

Auto-fix common issues:

```
openclaw security audit --deep --fix
```

Separate health check:

Browser Control Safety

If you have enabled browser control for your agent:

- **Use a dedicated browser profile** -- never let the agent use your personal profile
- **Never** have banking, personal email, or social media logged in on the agent's profile
- **Restrict browser access** to your main session only:

```
openclaw config tools deny browser --context sandbox
openclaw config tools allow browser --context main
```

If you do not need browser control:

```
openclaw config tools deny browser
```

Most users do not need browser control, especially when starting out.

Prompt Inoculation (ACIP)

⚓ Weekly Security Maintenance

The 10-Minute Weekly Check:

1. Run security audit: `openclaw security audit --deep`
2. Run health check: `openclaw doctor`
3. Fix any issues: `--fix` flags
4. Check API spending: visit your provider dashboard
5. Verify DM modes: `openclaw config channels [channel] dm-mode`
6. Verify sandbox status: `openclaw config sandbox mode`
7. Commit workspace backup: `git add -A && git commit -m "Weekly backup"`

Automate it:

```
Every Sunday at 10 AM, run a security audit.
Send results to Telegram. Alert me immediately for
```

Incident Response: The Five Steps

Step	Action	Time
STOP	Kill the ⛵ gateway immediately	30 seconds
CLOSE	Lock down access -- loopback only, disable DM channels	2 minutes
FREEZE	Rotate all 🔑 secrets -- gateway token, API keys, bot tokens	10-30 minutes
INVESTIGATE	Review logs, sessions, file modifications	1-2 hours
RESTORE	Fix root cause, audit, restart, 🔭 monitor 24-48 hours	30 min - hours

Incident Response: Commands

STOP

```
openclaw service stop
pkill -9 -f openclaw    # if it won't stop gracefully
```

CLOSE

```
openclaw config gateway bind loopback
openclaw config channels telegram dm-mode disabled
```

FREEZE

```
openclaw config gateway token rotate
openclaw config provider key [NEW_KEY]
openclaw config channels telegram token [NEW_TOKEN]
```

🚩 Shoals and Sandbars

Mistake	Fix
Not setting up Docker	Install Docker -- required for sandboxing
Sandbox mode "off"	Set to "non-main" at minimum
Elevated mode enabled	Disable: `openclaw config tools elevated off`
Loose file permissions	`chmod 700` directories, `chmod 600` files
Never running security audit	Run `--deep` weekly
Using personal browser profile	Create a dedicated profile for the agent
⛵ Gateway exposed without auth	Set bind to loopback, enable token auth
Skipping 🔑 secret rotation after incident	Rotate ALL secrets immediately

⚙ Hands on Deck

Part 1: Run the Audit (5 min)

- `opencláw security audit --deep`
- Note every finding: critical, warning, or info

Part 2: Fix All Issues (10 min)

- `openclaw security audit --deep --fix`
- Manually fix anything auto-fix missed

Part 3: Set Up Docker and Sandboxing (15 min)

- Install Docker, configure sandbox mode and scope, restart

Part 4: Test the Sandbox (10 min)

- From Telegram, ask your agent to list files or read `/etc/passwd`

💎 Treasure Chest

1. **Sandbox mode "non-main" is the sweet spot** -- direct sessions have full access, everything else is isolated
2. **Sandbox has NO network by default** -- test workflows from Telegram after enabling
3. **Docker is required for sandboxing** -- install it even if you do not enable sandboxing right away
4. **Tool policies are layered, deny always wins** -- check all levels when debugging
5. **Isolate your agent's network** -- guest WiFi or VLAN keeps a compromise contained
6. **Store credentials in a password manager** -- never leave 🔑 API keys in plaintext on disk
7. **Control outbound calls** -- disable ClawHub telemetry, review usage tracking, control remote embeddings
8. **File permissions matter** -- 700 for directories, 600 for files
9. [illegible]

Next Port of Call

Module 11: Maintenance and Troubleshooting

Day-to-day operations, updating 🦞 OpenClaw, common error solutions, and a printable command cheat sheet.