



Gobli

Free & Pro guide

Version 1.0

13 October 2026

Contents

1	What Gobli is	3
2	Free or Pro?.....	4
3	Helius: free plan vs paid plan	5
4	How Gobli Free works	6
5	How Gobli Pro works	8
	Why watch mode exists	9
6	Setup, step by step	11
	Gobli Free	11
	Gobli Pro	12
7	Creating the Telegram bot	13
8	Commands	14
9	Safety and trust.....	15
10	Limits and risk.....	16

1 What Gobli is

Gobli is a self-hosted Solana sniper bot that you drive from Telegram. It runs on your own PC or VPS, you talk to it through a private Telegram bot, and it buys and sells on your behalf from a wallet that only exists on your machine.

There are two snipers inside it:

- **Manual Sniper.** Paste a token's mint address (its contract address) into the chat and it is bought. Every position then gets sell buttons, a take-profit and a stop-loss.
- **Auto-Sniper.** You write rules in advance (a creator's wallet, or a ticker) and pump.fun launches that match are bought with no human in the loop.

Non-custodial, self-hosted

Gobli is not a service you log in to. You download the source, run it yourself and fill in your own settings file (`.env`). The private key you give it stays in that file on that machine and is used only to sign your own transactions. There is no Gobli server in the middle: the bot talks directly to the Solana data provider (Helius), to Jupiter, to a few public safety-check services and to Telegram. Section 9 lists every one of them.

Two builds

Gobli comes as **Gobli Free**, a free download, and **Gobli Pro**, a \$199 one-time purchase. Both are self-hosted, both run in Telegram, both ship with full source code, and both start in a mode that spends nothing. Section 2 compares them.

What you need for either build

Item	Where it comes from
A Helius API key	dashboard.helius.dev. The free plan runs both builds (Section 3).
A Telegram bot token	@BotFather (Section 7).
Your Telegram user id	@userinfobot (Section 7).
A machine to run it on	A Windows PC or a Linux VPS, with Python 3.10–3.13.
A dedicated hot wallet	Optional for Gobli Free (paper mode needs no key). Required for Gobli Pro, even in dry run.
A Jupiter API key	Optional, free tier from portal.jup.ag (see below).

Jupiter has announced that it will retire keyless API access. Both builds use Jupiter's keyless endpoint when `JUPITER_API_KEY` is blank and the keyed one when it is set, so adding a free key now means nothing breaks when that happens.

A few words used in this guide

Mint address	The address that identifies a token on Solana. It is what you paste into the Manual Sniper.
Bonding curve	Where a new pump.fun coin trades before it "graduates" to a regular market. "SOL in the curve" is how much real SOL buyers have put in.
Paper mode / dry run	Gobli Free calls it paper mode, Gobli Pro calls it dry run. Either way, nothing is spent.
Hot wallet	A wallet whose key sits in a file on a running machine. Use one made only for Gobli, holding only what you can lose.

2 Free or Pro?

	Gobli Free	Gobli Pro
Price	\$0	\$199, once
Snipers	Manual or Auto, one at a time	Manual and Auto, side by side
Open positions	Up to 3	No cap on manual; your own cap on auto
Manual buy route	Jupiter	Straight at the pump.fun curve, Jupiter for the rest
Sending	Standard RPC, fixed fee	Helius Sender, live priority fee
Sell buttons	50% / 100%	25 / 50 / 75 / 100%
Take profit / stop loss	Checked every 5s	Checked every 2s
Safety checks on manual buys	DexScreener, Rugcheck	Adds GoPlus and holder concentration
Auto rules	1 rule, 1 wallet or exact ticker, lasts up to 24h	Unlimited, many wallets and tickers each, contains-matching
Rule filters	Minimum SOL in the curve	Adds a max-SOL ceiling, creator-buy cap, instant ticker buys
Launch detection	Confirmed launches	Up to Helius' pre-execution feed
Extras	None	PnL share cards, bonk.fun alerts, pre-flight check
Paper / dry run, kill switch, daily cap	Yes	Yes
Source code, self-hosted	Yes	Yes

Gobli Free is the real bot on a short leash. It is not a demo with the buttons greyed out: both snipers work, end to end, on real launches. It starts in paper mode and runs with no private key at all, so you can watch exactly how it behaves before paying anything. It is deliberately a little slower and a little smaller: fewer positions, one auto rule, every trade routed through Jupiter and sent the standard way.

Gobli Pro is the full-speed version, with both snipers running side by side. Manual buys take the shortest route the coin allows, every trade goes out through Helius Sender, the Auto-Sniper reads the earliest launch feed your Helius plan offers and builds its pump.fun buys itself, and closed positions post a PnL card.

Moving from Free to Pro

You keep the same Helius key, the same Telegram bot and the same wallet. Two things to know:

- Pro's `.env` has more settings and names a few differently (for example `MANUAL_BUY_AMOUNT_SOL` where Free has `BUY_AMOUNT_SOL`). Open Pro's own `setup.html` and enter the same key, token, chat id and wallet; don't copy Free's `.env` across.
- Run one build at a time on a given Telegram bot. Stop Gobli Free before starting Gobli Pro with the same bot token.

3 Helius: free plan vs paid plan

Helius is the Solana data provider both builds use: the RPC for balances and reads, the launch feeds that tell the Auto-Sniper a coin has been created, and the route transactions are sent through.

The Helius free plan runs both Gobli Free and Gobli Pro. Every launch is seen and every buy goes out. If you are asking "will it still buy without the paid plan?" Yes, it will.

What a paid plan changes

A paid Helius plan (Developer, about \$49 a month, paid to Helius, not to Gobli) lets Gobli Pro also use two faster launch feeds: the **pre-execution feed** and the **transaction stream**. Both see a new launch earlier than the free plan's launch log does. Earlier sight means an earlier entry. Nothing else changes: the same rules, the same caps, the same sending.

Gobli Pro subscribes to every feed your key allows and acts on whichever reports a launch first, ignoring the duplicates that arrive later. On the free plan it simply runs on the launch logs.

The launch feeds

Feed	Helius method	Stage	Helius plan	Used by
Pre-execution	preprocessedSubscribe	Before the transaction has finished executing	Paid	Pro
Transaction stream	transactionSubscribe	After execution	Paid	Pro
pump.fun launch log	logsSubscribe	After execution	Free plan	Pro and Free
bonk.fun launch log	logsSubscribe	After execution	Free plan	Pro (alerts only)

Gobli Free always uses only the pump.fun launch log, whatever plan your key is on, and it reads it at "confirmed": it waits until the network has voted on the launch, so a launch that later drops off a fork is never acted on. Gobli Pro reads its feeds at "processed", without waiting for that vote.

What you see in Telegram

On the free plan, Gobli Pro tells you once which feeds it is using. The message appears a few seconds after the Auto-Sniper is switched on with a rule armed, the first time that happens after the bot starts. With the default setup it reads:

GOBLI PRO, HELIUS FREE PLAN

Your Helius key is on the free plan

Launches are detected through the pump.fun and bonk.fun feeds - your rules work as normal.

The faster pre-execution and transaction stream feeds come with a paid Helius plan (Developer, at dashboard.helius.dev). Upgrade the same key and restart the bot to use them.

This is information, not an error. Your rules are armed and working. `/stats` will also show the two paid feeds as "needs a paid Helius plan".

The message that does mean something is wrong is this one:

GOBLI PRO, NO FEED CONNECTED

No launch feed is connected - auto rules can't see launches.

• (each feed, with the reason it was refused)

Check `HELIUS_API_KEY` in `.env` and restart.

It means not even the free launch log connected, so the Auto-Sniper is blind. The usual cause is the API key: mistyped, revoked, or from a different account. Fix `HELIUS_API_KEY` in `.env` and restart.

A third message, "Some launch feeds were refused", appears if a feed is refused for a reason other than the plan. It lists the reasons and the feeds that are still detecting, so rules keep working through those.

Upgrading

Upgrade the **same key** at `dashboard.helius.dev`, then restart the bot. No setting changes. A feed that was refused is not retried while the bot is running, because the refusal is about the plan; the restart is what makes Pro try it again.

Credits: feeds only run while a rule is armed

Helius meters streaming by the message and by the megabyte, so feeds nobody is listening to would spend your plan's credits for nothing. In both builds the launch feeds run only while an auto rule is armed (in Pro, also only while the Auto-Sniper is switched on and the kill switch is off). An idle bot spends no streaming credits. Rules are armed ahead of a launch, so this never costs a snipe. Pro's menu shows "idle until a rule is armed" when the feeds are off for this reason.

4 How Gobli Free works

Both snipers in one free bot, with limits. You pick one mode at a time from the `/start` menu; positions from either mode stay visible and sellable in both.

Manual mode

Paste a mint address. The buy fires immediately through Jupiter, while DexScreener and Rugcheck run alongside it. Their results show on the BOUGHT message as context and warnings (for example a warning when liquidity is under \$5,000 or the Rugcheck score is under 60), never as a gate. Each position gets **Sell 50%**, **Sell 100%**, **Take profit** and **Stop loss** buttons.

Auto mode

Choose what you know about the coin:

- **Creator wallet.** The first coin that wallet launches on `pump.fun` is bought as soon as Jupiter can route it, usually within a second of the launch being confirmed. Gobli keeps asking Jupiter for up to 15 seconds.
- **Ticker.** Anyone can mint a coin with any ticker for a couple of cents, and people do it on purpose around an anticipated launch. So a ticker rule never buys at launch: it watches each exact match for 10 minutes and buys only once enough real SOL has gone into its bonding curve. Copycats almost never get there. Tickers that mix alphabets (a Cyrillic letter among Latin ones) are refused.

Then you set the buy size, take profit / stop loss, and how long the rule lives: 1 hour, 6 hours or 24 hours. Every rule expires. The launch feed only runs while a rule is armed.

Paper mode

Gobli Free starts in paper mode, and **works with no private key at all**. Paper trades are priced from Jupiter's live quotes, but nothing is ever sent, so you can try both snipers end to end with zero SOL at risk. With no key in `.env`, paper mode is forced on.

If you do put a funded wallet in `.env` and stay in paper mode, every buy is also built, signed and run through Solana's simulator, so a paper buy proves the live path works.

Going live

1. Create a **dedicated hot wallet** (never your main wallet) and fund it only with what you can lose.
2. Put its base58 private key in `.env` as `PRIVATE_KEY` (or add it in `setup.html`) and restart.
3. Send `/live confirm` to switch to real SOL. `/paper` switches back.

A restart always comes back in the mode `.env` declares (`DRY_RUN`, which defaults to `true`).

The menu

`/start` opens the menu and the mode switch: Manual or Auto. `/manual` and `/auto` go straight to each. The full command list is in Section 8.

Every Free limit, in plain words

Limit	Setting in <code>config.py</code>
Open positions at once, across both modes: 3 .	<code>FREE_MAX_OPEN_POSITIONS</code>
Sell buttons: 50% and 100% .	<code>FREE_SELL_PERCENTS</code>
Take-profit / stop-loss levels checked every 5 seconds .	<code>FREE_PRICE_CHECK_SECONDS</code>
One auto rule, with one creator wallet or one exact ticker, and it fires once .	<code>FREE_MAX_RULE_BUYS</code>
A ticker rule watches each match for 10 minutes for its SOL threshold.	<code>FREE_WATCH_WINDOW_SECONDS</code>
At most 10 launches watched at once.	<code>FREE_MAX_WATCHES</code>
Rule lifetimes: 1 h, 6 h or 24 h . Every rule expires.	<code>FREE_RULE_EXPIRY_CHOICES</code>
A creator snipe keeps asking Jupiter for a route for 15 seconds .	<code>FREE_ROUTE_WAIT_SECONDS</code>
Launches are read at " confirmed ".	<code>FEED_COMMITMENT</code>

Buy size is your choice. Gobli Free has no per-buy limit: set it with `BUY_AMOUNT_SOL` in `.env` or `/setbuy`. What bounds your spending is the daily cap on live spend (`MAX_SOL_PER_DAY`) and the wallet ceiling (`HOT_WALLET_MAX_BALANCE_SOL`), which are checked before every buy.

Why Free is deliberately a little slower

- **Jupiter routing.** Every buy goes through Jupiter, an aggregator, which adds a round trip. A brand-new coin also has to be indexed by Jupiter before it can be bought at all.
- **Standard send.** Transactions go out through the normal RPC `sendTransaction`, with a fixed priority fee from `.env` (`PRIORITY_FEE_LAMPORTS`), instead of Helius Sender with a live fee estimate.
- **Confirmed launches.** Free waits until the network has voted on a launch before acting on it, rather than reading the earliest stage a feed offers.

None of these stop it from working; they put it a little behind Pro on the same launch. That is the point of the free build: it shows exactly how Gobli behaves, on a short leash.

5 How Gobli Pro works

Both snipers run at the same time. Every position, whichever sniper opened it, shows up in one list with the same sell buttons, the same take-profit / stop-loss, and a PnL card when it closes.

Two snipers, two switches

`/start` opens the menu: **Manual Sniper**, **Auto-Sniper**, **Positions** and **Settings**. Each sniper's panel has a **Switch on** / **Switch off** button. Which snipers are switched on survives a restart. When the Manual Sniper is off, a pasted address is not bought; when the Auto-Sniper is off, the launch feeds stop.

Manual Sniper

Paste a mint address and it is bought at once with `MANUAL_BUY_AMOUNT_SOL`, by the shortest route the coin allows. Gobli picks the route for you:

- **Still on its pump.fun bonding curve** (most freshly pasted coins are): the buy is built locally and sent straight to the curve. No aggregator in between.
- **Anywhere else**: a coin that has graduated, or one on another DEX. It goes through **Jupiter**, whose quote was already fetched in parallel along with Helius' live priority-fee estimate.

Rugcheck, GoPlus, holder concentration and DexScreener run alongside the buy within a 3-second budget. Their results appear on the BOUGHT message as information and warnings; they never delay or block a buy.

Every position gets **Sell 25% / 50% / 75% / 100%**, plus **Take profit** and **Stop loss**: reply with a percent when asked. Levels are checked every `PRICE_CHECK_INTERVAL` seconds (2 by default) and the position is sold 100% when one is hit. Paste a coin you already hold and the buy is added to the position at a recalculated average entry. Sells you press yourself carry your slippage as a floor.

Auto-Sniper rules

`/newrule`, or **New rule** in the Auto-Sniper panel, walks you through a rule:

Trigger	You give it	When it buys
Creator wallet	One or more wallet addresses	Instantly, the moment that wallet launches anything
Ticker	A symbol, matched exactly or as "anything containing it"	Instantly, or once the coin clears thresholds you set (watch mode)

A ticker rule then asks: **Wait for real volume (safer)** or **Buy instantly (risky)** → minimum SOL in the curve → maximum SOL in the curve (the "too late" ceiling) → maximum creator buy % → buy size → expiry → take profit / stop loss. A creator rule skips straight to the creator-buy cap. Send `skip` to leave a threshold unset.

- **Buy size** defaults to `AUTO_BUY_AMOUNT_SOL` and can never exceed `MAX_SOL_PER_SNIPE`.
- **Expiry** is required: send 6h, 3d, 2w or a date. A forgotten rule that fires months later is how people lose money in their sleep.
- **Take profit / stop loss** are sent as two percentages, e.g. 100/40 for +100% and -40%.

Then **Arm it**. Manage rules with `/rules`, `/rule_on <id>`, `/rule_off <id>` and `/delete <id>`.

Why watch mode exists

Anyone can mint a coin with any ticker for about 0.02 SOL (cents) and people do this deliberately around an anticipated launch. A ticker rule that fires the instant a match appears mostly buys impersonators.

Fakes almost never attract real money. So "wait until the bonding curve has taken in N SOL" is the filter that separates them: watch mode spots the launch, keeps reading its curve, and buys only once real SOL has gone in. If the coin passes your maximum, graduates first, or the watch window runs out, it is dropped and you are told why.

That is why a ticker-only instant rule is something you have to choose explicitly (and you are warned again before arming it), and why tickers that mix alphabets (a Cyrillic "O" inside a Latin ticker) are rejected outright.

Risk caps, cooldown and kill switch

Setting (<code>.env</code>)	Default	What it does
<code>MAX_SOL_PER_SNIPE</code>	0.1	Largest single auto buy
<code>MAX_SOL_PER_DAY</code>	1.0	Total auto spend per day; survives a restart
<code>MAX_CONCURRENT_POSITIONS</code>	5	Auto positions open at once
<code>PER_CREATOR_COOLDOWN_SECONDS</code>	300	Wait before buying from the same creator again, serial rugpulls launch several coins in a row
<code>HOT_WALLET_MAX_BALANCE_SOL</code>	2.0	Refuses to start if the wallet holds more; also catches the wrong key

The caps are enforced immediately before each buy, and changing them needs an `.env` edit and a restart, deliberately. The Auto-Sniper never buys the same mint twice.

Kill switch. `/panic` disables every rule, drops every watch and sells every position. Nothing buys again, manual or auto, until `/resume`. After `/resume`, rules stay paused; re-arm them one by one with `/rule_on <id>`.

Exits never revert on price. Take profit, stop loss and `/panic` sell into the curve with no floor, and through Jupiter with at least 25% slippage: an exit that fails because the price moved is worse than one that fills a little lower. A real position is always really sold, even if you switch to dry run after opening it, and a dry-run position is never really sold.

Dry run first

Pro starts in dry run (`DRY_RUN=true`). Pasted mints and matched launches are built, signed and simulated against the real chain, and open simulated positions you can sell and set exits on, so you see everything behave before it can cost anything. Then:

1. **Pre-flight.** `/preflight <mint>`, with any pump.fun coin still on its curve, builds and signs the real buy and simulates it with signature verification on. Nothing is sent. If it fails, don't go live; the output is the diagnostic.
2. **Watch it in dry run** for a while.
3. **/live confirm**, with small buy sizes. `/dry` goes back. A restart always comes back in the mode `.env` declares.

Helius Sender

Every Pro trade goes out through Helius Sender, which sends the transaction across Helius' staked connections and Jito. Set `SENDER_REGION` to the endpoint nearest the machine (`fra`, `ewr`, `lon`, `ams`, `slc`, `sg`, `tyo`, or `global`). Gobli keeps the connection warm, and if that region can't be reached from your machine it moves to Helius' global endpoint by itself. Sender requires a small tip on every transaction: `TIP_SOL`, 0.001 SOL by default, changeable with `/settip`. `/stats` shows the real timings per route and which feeds are winning.

bonk.fun

bonk.fun (Raydium LaunchLab) launches are detected and reported. Buying there is off (`LAUNCHLAB_BUY_ENABLED` in `config.py`) until one live buy and one live sell have been confirmed on that venue.

PnL cards

Every position that closes for real (take profit, stop loss, a sell button or `/panic`, from either sniper) posts a Gobli share card to your Telegram a few seconds after the last sell lands: green with a celebrating goblin for a win, red with a miserable one for a loss. The numbers are realized, read back from the transactions themselves: what your wallet actually made, tips and fees included. Cards never touch trading, and render in a separate process started without your private key.

<code>.env</code>	Default	Effect
<code>PNL_CARDS</code>	<code>true</code>	Turn cards off entirely
<code>PNL_CARD_USE_TELEGRAM_PROFILE</code>	<code>true</code>	Your @username and profile photo on the card; <code>false</code> shows the default goblin avatar
<code>PNL_CARD_USERNAME</code>	None	Name shown instead of your Telegram username
<code>PNL_CARD_HIDE_TOKEN</code>	<code>false</code>	Blur the ticker

Buy sizes, slippage, the tip and which snipers are switched on survive a restart. Live mode deliberately does not.

6 Setup, step by step

Both builds need Python **3.10–3.13**; the pinned dependencies may not build on newer releases yet. Create the Telegram bot first (Section 7) and get a Helius key from dashboard.helius.dev; the free plan is enough.

Gobli Free

1. **Download and unzip.** Get the .zip from gobli.io (the How It Works page) or from @gobliddev, and from nowhere else (see Section 9). Unzip it to a folder of its own.
2. **Write your .env with setup.html.** Open `setup.html` in Chrome, Edge or Brave, fill in the form, click **Save .env** and pick the bot's folder. It checks every value the same way the bot does on startup, shows which wallet a private key belongs to, and never makes a network request (view its source to confirm). Leave the wallet on **Paper only** to try the bot without a key. Firefox and Safari download the file instead: rename it to exactly `.env` and move it into the folder.
Or by hand: copy `.env.example` to `.env` and fill it in.

3. **Install and run on Windows:**

```
python -m venv venv
venv\Scripts\activate
pip install -r requirements.txt
python main.py
```

On Linux or macOS, activate with `source venv/bin/activate` instead.

4. **Or on a Linux VPS:**

```
bash vps/install.sh
bash vps/start.sh
```

This does the same and locks `.env` down (on Linux the bot refuses to start if `.env` is readable by other users; the fix is `chmod 600 .env`). `vps/gobli-free.service` is a systemd service that keeps it running unattended.

5. **First run.** The bot validates every `.env` value and prints the full list of problems before refusing to start, rather than running on a bad config. When it is up, it messages you in Telegram. Send `/start` and pick a sniper.

Gobli Free .env at a glance

Setting	Default	Notes
HELIUS_API_KEY	None	Required. Free plan is enough
TELEGRAM_BOT_TOKEN	None	Required. From @BotFather
TELEGRAM_CHAT_ID	None	Required. Your numeric id from @userinfobot
PRIVATE_KEY	blank	Only to trade real SOL. Dedicated hot wallet, base58
JUPITER_API_KEY	blank	Optional, free from portal.jup.ag
BUY_AMOUNT_SOL	0.02	Your choice; Free has no per-buy limit
SLIPPAGE_BPS · AUTO_SLIPPAGE_BPS	500 · 2500	Manual · auto slippage (500 = 5%)
PRIORITY_FEE_LAMPORTS	300000	The fixed priority fee
DRY_RUN	true	Paper mode. Forced on when there is no key
MAX_SOL_PER_DAY	0.5	Daily cap on live spend
HOT_WALLET_MAX_BALANCE_SOL	1.0	Refuses to start above this balance

`HTTP_RPC_URL` and `WS_RPC_URL` are optional overrides; leave them blank.

Gobli Pro

1. **Unzip** the Gobli Pro package into a folder of its own.
2. **Install on a Linux VPS** (the recommended way to run it):

```
bash vps/install.sh
```

That creates the virtual environment, installs the pinned dependencies, downloads the headless browser that draws the PnL cards (about 150 MB, kept inside the project folder), locks down `.env`, and runs the tests to confirm the install is sound.

3. **Install manually, or on Windows:**

```
py -3.12 -m venv venv
venv\Scripts\activate
pip install -r requirements.txt
python -m playwright install chromium
```

On Linux/macOS: `python3 -m venv venv`, then `source venv/bin/activate`, and add `--with-deps` to the last line on Linux. Always use the `venv`: dependencies are pinned exactly because this process signs transactions with a live key, and packages installed globally could otherwise shadow the pinned versions.

4. **Write your `.env`** with Pro's `setup.html`, exactly as for Free (Chrome, Edge or Brave; Firefox and Safari download it, rename to `.env`). Set `SENDER_REGION` to the Sender endpoint closest to the machine. A private key is required.
5. **Fund the hot wallet** with only what you accept losing. The bot refuses to start if the wallet holds more than `HOT_WALLET_MAX_BALANCE_SOL`.
6. **Run:**

```
bash vps/start.sh          # or: python main.py
```

To keep it running unattended, install the systemd service:

```
sudo cp vps/gobli.service /etc/systemd/system/
sudo nano /etc/systemd/system/gobli.service # set User= and the paths
sudo systemctl daemon-reload
sudo systemctl enable --now gobli
journalctl -u gobli -f                    # watch the log
```

7. **First run.** Like Free, it validates every value and lists every problem before refusing to start. Send `/start`, leave it in dry run, and run `/preflight` before going live (Section 5).

Gobli Pro `.env` at a glance

Setting	Default	Notes
<code>HELIUS_API_KEY</code>	None	Required. One key covers the RPC, the launch feeds and Sender
<code>SENDER_REGION</code>	<code>fra</code>	<code>slc · ewr · lon · fra · ams · sg · tyo · global</code>
<code>PRIVATE_KEY</code>	None	Required. Dedicated hot wallet, base58
<code>TELEGRAM_BOT_TOKEN</code> · <code>TELEGRAM_CHAT_ID</code>	None	Required
<code>JUPITER_API_KEY</code>	blank	Optional
<code>DRY_RUN</code>	<code>true</code>	Start here
<code>TIP_SOL</code>	0.001	Sender tip per transaction; below 0.000005 Sender drops it
<code>MANUAL_BUY_AMOUNT_SOL</code> · <code>MANUAL_SLIPPAGE_BPS</code>	0.05 · 500	Manual buy size · slippage
<code>PRIORITY_FEE_LAMPORTS</code>	500000	Manual fallback when no live estimate arrives in time

Setting	Default	Notes
MIN_LIQUIDITY_USD · MIN_RUGCHECK_SCORE · MAX_TAX · MAX_HOLDER_PCT	5000 · 60 · 10 · 15	Warnings on manual buys; never block
PRICE_CHECK_INTERVAL	2	Seconds between take-profit / stop-loss checks
AUTO_BUY_AMOUNT_SOL · AUTO_SLIPPAGE_BPS	0.05 · 2500	Default auto buy size · snipe slippage
PRIORITY_FEE_MICRO_LAMPORTS	500000	Auto snipes, per compute unit

Risk caps and PnL-card settings are listed in Section 5. `HTTP_RPC_URL`, `PREPROCESSED_WS_URL`, `LASERSTREAM_WS_URL` and `HELIUS_SENDER_URL` are optional overrides built from your key; leave them blank.

7 Creating the Telegram bot

Each copy of Gobli talks to you through a Telegram bot that you create and own. It takes two minutes.

WARNING: NAME YOUR BOT SOMETHING BORING

Give the bot a boring, unremarkable name and username. Avoid "sniper", "solana", "trade", "pump" and anything similar, in the name, the username and the bio.

Telegram's automated filters have banned trading bots on the name alone. Something forgettable like "Ledger Notes" is the right idea.

1. In Telegram, message **@BotFather** and send `/newbot`.
2. Choose the name and username, **boring**, as above.
3. BotFather replies with a token like `8123456789:AAH...`. That is `TELEGRAM_BOT_TOKEN`. Treat it as a password.
4. Message **@userinfobot**. It replies with your numeric user id. That is `TELEGRAM_CHAT_ID`.
5. Open a chat with your new bot and press **Start**.

Only that one chat id ever gets a reply. Every other chat is ignored silently, so even if someone finds your bot, it will not answer them.

Optional: the command menu

Send BotFather `/setcommands`, choose your bot and paste the list for your build. Telegram then suggests the commands as you type.

Gobli Free

```
start - Menu and mode switch
manual - Manual mode: paste a mint to buy
auto - Auto mode: set or manage your rule
positions - Open positions
balance - Wallet balance
config - Settings and limits
setbuy - Set buy amount
paper - Paper trading
live - Trade real SOL
panic - Pause the rule and sell everything
upgrade - What the full bots add
help - All commands
```

Gobli Pro

```
start - Menu: both snipers, positions, settings
positions - Open positions and watches
newrule - Build an auto-snipe rule
rules - List rules
stats - Speed, feeds and pipeline numbers
config - Every setting
balance - Wallet balance
preflight - Pre-flight check against a real coin
setbuy - Set buy amount
dry - Simulate only
live - Spend real SOL
panic - Disable rules and sell everything
help - All commands
```

8 Commands

Gobli Free

Command	What it does
<code>paste a mint</code>	Buy it (Manual mode)
<code>/start</code>	Menu and mode switch
<code>/manual</code>	Manual mode: paste a mint to buy
<code>/auto</code>	Auto mode: set or manage your rule
<code>/rule</code>	Your auto rule
<code>/positions</code>	Open positions
<code>/sell <mint></code>	Sell a position
<code>/balance</code>	Wallet balance
<code>/config</code>	Settings and limits
<code>/setbuy <sol></code>	Set the buy amount
<code>/setslippage [auto] <bps></code>	Manual slippage, or auto slippage with <code>auto</code>
<code>/paper · /live confirm</code>	Paper trading · trade real SOL
<code>/panic · /resume</code>	Pause the rule, drop every watch and sell everything · release the kill switch
<code>/upgrade</code>	What the paid build adds
<code>/cancel</code>	Abandon a half-finished input
<code>/help</code>	All commands

Gobli Pro

Command	What it does
<code>paste a mint</code>	Buy it (Manual Sniper)
<code>/start</code>	Menu: both snipers, positions, settings
<code>/manual · /auto</code>	Open the Manual Sniper · Auto-Sniper panel
<code>/positions</code>	Every open position with sell buttons, plus watched launches
<code>/sell <mint> [%]</code>	Sell a position (100% if no percent)
<code>/newrule</code>	Build an auto rule
<code>/rules</code>	List rules
<code>/rule_on <id> · /rule_off <id> · /delete <id></code>	Arm · pause · delete a rule
<code>/preflight <mint></code>	Pre-flight check against the chain; run before <code>/live</code> (<code>/testbuy</code> is an older alias)
<code>/stats</code>	Speed per route, feeds and pipeline numbers
<code>/config · /balance</code>	Every setting · wallet balance
<code>/setbuy <sol> · /setbuy auto <sol></code>	Manual buy size · auto default
<code>/setslippage <bps> · /setslippage auto <bps></code>	Manual · auto slippage
<code>/settip <sol></code>	Sender tip per transaction
<code>/dry · /live confirm</code>	Simulate only · spend real SOL
<code>/panic · /resume</code>	Disable every rule and sell everything · release the kill switch
<code>/cancel</code>	Abandon a half-finished input
<code>/help</code>	All commands

`/live` on its own shows the wallet and balance and asks you to send `/live confirm` if you mean it. In both builds, only the chat id in `TELEGRAM_CHAT_ID` gets a reply.

9 Safety and trust

A free bot that asks for a private key is exactly what a wallet drainer looks like. Here is how Gobli is built so you don't have to take that on trust.

Only from gobli.io or @gobliddev

Scammers repackage free bots with key-stealers added. **Download Gobli Free only from gobli.io or t.me/gobliddev.** A copy from anywhere else is not this software, even if it looks identical.

Where your key lives

- The key sits in `.env` on your machine, and nowhere else. On Linux, both builds refuse to start if `.env` is readable by other users.
- **Gobli Free doesn't need a key at all:** paper mode runs without one.
- **Full source, no obfuscation.** In Gobli Free the key is read in `config.py` and used in `trader.py`, which signs with it, and nowhere else (`logger.py` also reads it, only to scrub it out of any log line). A test, `tests/test_network.py`, fails if the key is ever read anywhere but `config.py`.
- Both builds redact secrets from their logs. In Gobli Pro, PnL cards render in a separate process started without the private key in its environment.
- `setup.html` never makes a network request; view its source to confirm.
- Gobli Free has no phone-home: no telemetry, no licence server, no update check.

What leaves your machine: Gobli Free

Host	Why
<code>mainnet.helius-rpc.com</code>	Your RPC: balances, curve reads, sending, the launch feed
<code>lite-api.jup.ag</code> or <code>api.jup.ag</code>	Jupiter: quotes, swap transactions, prices
<code>api.rugcheck.xyz</code>	Rugcheck score on manual buys
<code>api.dexscreener.com</code>	Name, market cap and liquidity on manual buys
<code>api.coingecko.com</code>	SOL price, only if Jupiter's price is unavailable
<code>api.telegram.org</code>	The bot itself

Solscan and DexScreener appear only as links in messages to you. `tests/test_network.py` fails if the code ever contains a host not on this list, or if `setup.html` gains any way to make a request.

What leaves your machine: Gobli Pro

Host	Why
<code>mainnet.helius-rpc.com</code>	Your RPC, the transaction stream and the launch logs
<code>beta.helius-rpc.com</code>	The pre-execution feed (paid Helius plans)
<code><region>-sender.helius-rpc.com</code> , <code>sender.helius-rpc.com</code>	Helius Sender: your chosen region, and the global fallback
<code>lite-api.jup.ag</code> or <code>api.jup.ag</code>	Jupiter: quotes, swaps, prices
<code>api.rugcheck.xyz</code> , <code>api.gopluslabs.io</code> , <code>api.dexscreener.com</code>	Safety checks on manual buys
<code>api.coingecko.com</code>	SOL price for the USD line on PnL cards
<code>api.telegram.org</code>	The bot itself

These are the endpoints defined in Pro's `config.py`. Installing also downloads the pinned Python packages and, for PnL cards, a headless browser.

Dedicated hot wallet only

Never put your main wallet's key in either build. Create a wallet just for Gobli and fund it only with what you can afford to lose entirely. `HOT_WALLET_MAX_BALANCE_SOL` makes the bot refuse to start if the wallet holds more than you said it would, a blast-radius limit that also catches the wrong key pasted into `.env`.

Guard rails

Guard rail	Gobli Free	Gobli Pro
Starts without spending	Paper mode by default	Dry run by default
Kill switch	<code>/panic</code> pauses the rule, drops every watch, sells every position	<code>/panic</code> disables every rule, drops every watch, sells every position
Daily cap	<code>MAX_SOL_PER_DAY</code> on live spend; survives a restart	<code>MAX_SOL_PER_DAY</code> on auto spend; survives a restart
Other caps	3 open positions (built in)	<code>MAX_SOL_PER_SNIPE</code> , <code>MAX_CONCURRENT_POSITIONS</code> , creator cooldown
Wallet ceiling	<code>HOT_WALLET_MAX_BALANCE_SOL</code>	<code>HOT_WALLET_MAX_BALANCE_SOL</code>
No repeat buys	Auto mode never buys the same coin twice	Auto-Sniper never buys the same mint twice

All of them are checked immediately before every buy. Gobli Free also refuses a buy that would leave too little SOL in the wallet to pay for the sell.

10 Limits and risk

Where Gobli Free stops

1. **Positions are in memory.** A restart stops tracking them and their take-profit / stop-loss; the tokens stay in your wallet. The mode, the auto rule, buy size and the daily spend do persist.
2. **A brand-new coin needs Jupiter to index it** before it can be bought. The auto mode keeps trying for 15 seconds.
3. **Ticker rules need a SOL-paired coin**, since the threshold is SOL in the curve. A good share of pump.fun launches are now quoted in other tokens; ticker rules skip those. Creator rules buy either kind.
4. **Paper fills are Jupiter's quote**, not a landed trade. Real fills can be worse, especially on a coin minted seconds ago.
5. **Safety checks are informational** and cover the basics only.

Where Gobli Pro stops

1. **Positions are in memory.** A restart stops tracking them and their take-profit / stop-loss. The tokens stay in your wallet. Rules, caps, settings and daily spend do persist.
2. **Auto snipes buy on bonding curves only:** pump.fun live, bonk.fun detection only. Positions can be sold after a coin graduates (through Jupiter), and the Manual Sniper buys anything Jupiter can route.
3. **Auto-snipe safety checks are the fast ones only:** what can be read from the create transaction itself. The full Rugcheck / GoPlus / holder checks run on manual buys.
4. **A launch's metadata is never fetched**, because that round trip takes seconds. Matching uses the symbol from the create instruction.
5. **pump.fun changes its program** every few months, and a wrong account list fails on-chain, after the snipe is lost. After any pump.fun announcement, run `python tests/test_layout.py`.

Risk

Trading fresh Solana tokens loses money often, and an auto-sniper loses it without asking. Most new coins are rugs or jokes, transactions can fail after paying the fee, and liquidity can vanish faster than an exit can fire. Fund the hot wallet only with what you can afford to lose entirely, and treat `MAX_SOL_PER_DAY` as the amount you are willing to lose in a day, not a target.

Fixed costs decide small trades. In Gobli Pro every trade pays your tip twice, once on the buy and once on the sell, plus roughly 0.0015–0.002 SOL of rent to create the account that holds a new coin (taken back when a curve position is fully sold). At a 0.002 SOL buy size with a 0.001 SOL tip, a position has to about double just to break even; at 0.5 SOL the same costs are noise.

Read Section 4 of the `LICENSE` file in either build before funding anything.

Not financial advice. Gobli is software that does what you configure it to do. Nothing in this guide is a recommendation to buy or sell any token. Every trade, and every loss, is yours.