QWU WORLD · MINECRAFT

Qcord — Privacy notice

What the mod sends, what our relay keeps, and what it never touches.

Last updated **1 September 2026** · Applies to **Qcord 1.2.0** · qwu.world/minecraft/qcord/privacy

Who runs this

Qcord and the relay server it connects to are operated by **QwU World**. Contact: **it.qwuworld@gmail.com**, or the `#bugs-suggestions` channel in the Minecraft section of our Discord: discord.gg/sSENQuz6a7. Either reaches us; the email is the formal route for anything in this notice and for the bundled open-source libraries.

Qcord is a client-side Minecraft mod. It sends video, screen shares and voice between the members of a party through a relay server that QwU World rents and runs:

RELAY	LOCATION	ADDRESS	PROVIDER
The relay	Warsaw, Poland	`57.128.197.25:8797`	OVH Sp. z o.o.

There is one relay and it is in the EU, so your data is not transferred to a third country. The mod ships with no second address: `network.relayFallback` is empty unless you fill it in yourself.

You do not have to use our relay, and two settings in `config/qcord-client.toml` keep you off it:

- `network.relayAddress` — the relay you create parties on. Point it at your own.
- `serverLobby.autoJoin` — leave it off (it is off until you switch it on) or the automatic lobby lookup described below still goes to our relay regardless of the first setting.

`network.lanMode` changes where a party is **hosted** — your own game client becomes the host instead of a relay — but it does not by itself stop the automatic lobby lookup, which always uses the configured relay endpoints. Leave `serverLobby.autoJoin` off as well.

What reaches our relay, and when

Nothing at all until you agree

> On its first launch Qcord shows a welcome screen that says what the mod captures, where the media goes and what the relay sees, and asks whether server lobbies may be joined automatically. Until you press **Get started**, the mod opens no connection of any kind.

Creating or joining a party, the `/qcord` commands and the automatic lobby lookup are all blocked before you answer, and the automatic path stays silent rather than interrupting you. The answer is stored as `privacy.consentVersion` in `config/qcord-client.toml` (0 = never answered); the screen reopens any time with `/qcord privacy`, and setting the value back to 0 makes the mod ask again.

When you join a multiplayer world

About three seconds after you enter a multiplayer world, Qcord opens a TCP connection to the relay and asks whether that Minecraft server has a Qcord lobby — but only if you allowed it. `serverLobby.autoJoin` is **off by default** and is switched on by the toggle on the welcome screen, which explains exactly this before you touch it.

That first question contains **one value**: a "server key", the first 16 hex characters of a SHA-256 hash of the address you connected to. It is not reversible on its own, but the address of a public Minecraft server is guessable, so treat it as "which server you are playing on" rather than as a secret. Your name and UUID are **not** sent at this point.

If the server has no lobby, the connection is closed and nothing else happens. **In single player the mod never contacts the relay on its own** — the automatic lookup is skipped entirely — but it does connect if you create or join a party yourself while in a single-player world.

When you join a party or a lobby

Your client sends, and the relay writes to its log:

- your **IP address, shortened before it is written down** — the relay needs the full address to hold the TCP connection open, but it truncates it to the network (an IPv4 address loses its last octet, an IPv6 address everything after the first 48 bits) and drops the port before anything reaches the log;
- your **Minecraft username**, so the other people in the room can see who you are and whose camera is whose. Your **account UUID travels to the other members** of your room so their clients can match a stream to a player, but it is **not written to the log**;
- the **Qcord version** you are running (`qcord/1.2.0`);
- the **room code** you are in.

When you leave, the relay additionally logs the reason and counters of how much you sent — numbers of video, screen-audio and voice packets, and how many were dropped.

Other events are logged with your username and, where it applies, the room code:

- that you started sharing sound or speaking for the first time in that room;
- that you were kicked, muted or restricted, or promoted to host, and who did it;
- that a room or lobby was created, and that a host switched party voice on or off;
- that a host action of yours was **denied**, and which action it was;
- that you were sending faster than the relay's rate limit;
- that a connection of yours was **refused** — a version mismatch, a full room, a wrong token, a duplicate name — together with your shortened IP and the name you tried to use;
- that a reconnect of yours took over a session the relay had not yet reaped;
- for a connection that never gets as far as joining a room — including every automatic lobby lookup — the shortened IP and the reason it closed.

We do not collect anything about you beyond what is needed to run the relay and debug it. There are no accounts, no analytics, no profiles and no tracking across sessions.

Your camera, screen and voice

Video frames, screen-share audio and voice packets pass **through** the relay and are forwarded to the other people in your room. The relay never decodes them and never writes them to disk.

One exception is worth naming precisely, because a blanket "nothing is stored" would be untrue: the relay keeps the **most recent video keyframe of each live stream in memory**, so that somebody joining halfway through sees a picture immediately instead of a grey box. It is a single still frame, it lives in RAM only, it is replaced by the next keyframe, and it is dropped when the stream stops or the room is deleted.

Capture only runs while you switch it on. Turning a camera or share off, leaving the party, or leaving the world stops it. The microphone is only ever opened inside a private party whose host has switched party voice on — a server-wide lobby carries no voice at all — and never if you set `voice.enabled = false`.

Server lobbies (stored on disk)

If an operator opens a server-wide lobby, one record is written to `lobbies.json` on the relay. It contains the server key, the lobby's room code, its join token, its policy, the account UUID of whoever created it, the account UUIDs of its hosts, when it was created, and the hashed extra addresses (aliases) an operator attached to it — up to 16 per lobby, each one a server key like the one above.

The record is kept until somebody deletes that lobby, or until **60 days** pass without anybody looking it up or joining it, whichever comes first. The sweep runs when the relay starts and once a day after that.

Private parties are not stored: a private room is deleted 60 seconds after its last member leaves, and nothing about it survives a relay restart.

What we do not do

- No accounts, no sign-up, no password, no email address.
- No analytics or usage statistics, no crash reporting, no advertising, no profiling.
- No update check. Qcord's own code contains no HTTP or HTTPS client and no hostname other than the relay address in your config — it speaks only its own TCP protocol. The bundled FFmpeg and WebRTC libraries *do* contain general-purpose network code, because they are the stock upstream builds and we ship them unmodified. Qcord never calls those code paths: it uses FFmpeg only for local capture, encoding and decoding, and webrtc-java only for audio processing. If you run `strings` on the jar you will find those URLs; that is what they are.
- Nothing is downloaded at runtime; the only Qcord code that runs is what ships inside the jar. To list your screens, windows and devices the mod also runs standard tools that are already on your system — PowerShell on Windows, `osascript`, `system_profiler` and `screencapture` on macOS, `xrandr` and `wmctrl` on Linux — and reads their output. It does not modify anything with them.
- Nothing is sold, shared, published, or handed to any third party. The only company involved at all is the hosting provider named above, acting for us and seeing only what any host sees.

Why we process it, and for how long

Our basis is our legitimate interest in operating the service (GDPR Article 6(1)(f)): the username and UUID are what make a party work at all, and the connection log is what lets us keep the relay running, find bugs and deal with abuse.

- **Relay logs:** kept for **14 days**, then deleted.
- `lobbies.json`: kept until that lobby is deleted or goes 60 days unused, because it is what makes the lobby exist.

You can ask us for a copy of what we hold about you, ask us to correct or delete it, object to it, or ask us to restrict it. Write to **it.qwuworld@gmail.com** with your Minecraft username. If you think we have handled your data badly, you can complain to your national data protection authority.

Things you should know before you use it

> **The connection to the relay is not encrypted.** Qcord 1.2.0 uses plain TCP with no TLS and no end-to-end encryption. In practice that means somebody positioned on the network path between you and the relay — a hostile Wi-Fi, an ISP, a hosting provider — could reconstruct the picture and the sound of a call. Treat a Qcord call as an open stream, not as a private channel.

> **Anybody holding your invite code can join your party** and will see and hear everything in it. The code is the credential; there is nothing else. If you use `/qcord invite <player>`, the code is sent as a normal Minecraft whisper through the Minecraft server, so that server's staff and any chat-logging plugin can read it. Copying the code to your clipboard and sending it elsewhere avoids that.

> **A server lobby is effectively public.** The relay hands a lobby's join credentials to anybody who asks it about that Minecraft server's address, so assume that anyone who knows the server exists can walk into its lobby and see the roster and the cameras in it.

> **Technically, we could watch a stream.** Because there is no end-to-end encryption, the operator of a relay is in a position to decode what passes through it. We do not do this, and the relay software does not do it — but you should know that the architecture relies on that promise rather than on mathematics. If that is not good enough for what you are about to share, run your own relay or use LAN mode.

> **Screen sharing shows what is on your screen.** Notifications, tabs, private messages and anything else visible in the window you picked go to everybody in the room. Be aware of who else is in the room with you, and tell people when your camera is on.

What a Minecraft server can see

Nothing. Qcord installs nothing on the Minecraft server and sends nothing over the Minecraft connection — except in one case: the `/qcord invite <player>` command sends the invite code as an ordinary whisper, which goes through the server like any other chat message. Server operators cannot see your camera, your screen, your microphone, or that you are in a party.

What stays on your own computer

- `config/qcord-client.toml` — your settings, and the account UUIDs of people you have adjusted the volume or camera size for. It stays after you close the game.
- `qcord-natives/` in your game directory — the FFmpeg and WebRTC libraries, and the two capture helpers under `qcord-natives/helpers/`, extracted from the jar on first launch.
- Your Minecraft log (`latest.log`) records the title of the window or application you shared and the names of the camera, microphone and audio devices that were opened. Worth knowing before you attach a log to a bug report.

None of this is sent anywhere.

Age

Qcord is not intended for children under 13, and we ask that you do not use it if you are younger than that.

Changes

If this notice changes, the date at the top changes with it, and anything material will be mentioned in the changelog of the release it applies to.