Getting Started

Getting Started documentation for Nord Doorlock.

- Overview
- Requirements & Compatibility
- Installation
- Quick Start

Overview

Nord Doorlock is a modular FiveM door-locking resource designed for server owners and developers who need in-game door management, persistent state, framework bridges, double-door support, access control and a developer API.

It is standalone at its core and does not require ox_doorlock.

Core Features

- In-game administrator panel opened with `/doors`
- Wizard-based door creation and editing
- Aim-and-capture placement
- Single doors and double doors
- GTA DoorSystem synchronization
- Locked/unlocked state synchronization for all clients
- Persistent doors across resource and server restarts
- Job and minimum-grade access rules
- Identifier access rules
- ACE permission access rules
- Optional lockpicking
- Automatic relocking
- Native world-space locked/unlocked icons
- Custom interaction NUI
- Duplicate physical-door protection
- Automatic duplicate cleanup
- JSON/KVP persistence without a database dependency
- Optional oxmysql persistence
- ESX, QB-Core, QBX, nord-core and standalone framework support
- Notification, Text UI, Target and Inventory bridges
- Server-authoritative access and distance validation
- Client and server exports
- Custom authorization hooks

Design Principles

Door state changes are authoritative on the server. The client may request a lock, unlock or lockpick action, but permissions, distance and persistence are validated server-side.

Door definitions are copied when returned through the developer API. This allows external resources to inspect them without accidentally mutating Nord Doorlock's internal state.

The same physical GTA door object cannot belong to multiple Nord Doorlock entries. This prevents invisible duplicate locks and conflicting DoorSystem registrations.

Requirements & Compatibility

Nord Doorlock can run without a framework or database.

Required

- FiveM server
- GTA V game build supported by your FiveM server
- The resource folder must be named `nord_doorlock` if you use the examples exactly as shown

Optional Integrations

Category	Supported Providers
Framework	Standalone, ESX, QB-Core, QBX, nord-core
Storage	KVP/JSON, oxmysql
Notifications	Standalone fallback, ox_lib, QB-Core, ESX, nord_notify
Text UI	Built-in NUI, ox_lib, nord_textui
Target	ox_target, qb-target, nord_target
Inventory	nord_inventory, ox_inventory, qb-inventory/QB-Core, ESX

Automatic Detection

Most integration settings support:

```
Config.Framework = 'auto'
Config.Notify = 'auto'
Config.TextUI = 'auto'
```

```
Config.Target = 'auto'
Config.Inventory = 'auto'
```

The resource checks running providers and selects the first compatible integration.

For production servers, explicitly setting the provider can make the configuration easier to understand and prevent an unintended provider from being selected after resource changes.

ox_lib

ox_lib is optional. Nord Doorlock only uses it when the relevant global functions exist.

If ox_lib is used for lockpicking, `lib.skillCheck` is used. Without it, the configured fallback success chance is used.

oxmysql

oxmysql is only required when:

```
Config.Storage = 'mysql'
```

If `Config.Storage = 'auto'`, MySQL is selected when oxmysql is started; otherwise Nord Doorlock uses its KVP/JSON persistence mode.

Installation

1. Install the Resource

Place the resource in your FiveM resources directory:

```
resources/[nord]/nord_doorlock
```

Keep the resource folder name as:

```
nord_doorlock
```

2. Start the Resource

Add to `server.cfg`:

```
ensure nord_doorlock
```

When using optional integrations, start them before Nord Doorlock. Example:

```
ensure oxmysql
ensure ox_lib
ensure ox_target
ensure nord_doorlock
```

Only include resources you actually use.

3. Administrator Permission

The default administrator ACE is:

```
Config.AdminAce = 'nord.doorlock.admin'
```

Example:

```
add_ace group.admin nord.doorlock.admin allow
```

Framework administrator groups can also be accepted through:

```
Config.AdminGroups = {
    admin = true,
    superadmin = true,
    god = true
}
```

4. Select Storage

Default:

```
Config.Storage = 'json'
```

This uses FiveM Resource KVP as the authoritative persistence source and mirrors the data to JSON files.

For MySQL:

```
Config.Storage = 'mysql'
```

Start oxmysql before Nord Doorlock. The table is automatically created.

An optional SQL installer is also provided:

```
sql/install.sql
```

5. Start and Verify

Restart the resource:

```
restart nord_doorlock
```

Then check persistence health:

```
doors_storage
```

A healthy result should report that storage is ready and show no active error.

6. Open the Door Manager

In game:

```
/doors
```

If the menu does not open, verify your ACE or framework admin group.

Quick Start

This is the shortest workflow for creating a working door.

Create a Door

Open:

```
/doors
```

Select **New door**.

The wizard contains three steps.

Step 1 — General

Configure:

- Door name
- Interaction distance
- Auto-lock delay
- Default locked state
- Lockpick enabled/disabled

Step 2 — Door Objects

Capture **Door 1**.

Aim the crosshair at the physical GTA door object. When the object is valid it is highlighted and the capture indicator changes state.

Press:

```
E
```

For a double door, repeat the process for **Door 2**.

Step 3 — Access

Optionally add:

- Jobs and minimum grades
- Identifiers
- ACE permissions

Leave all access fields empty for a public door.

Finish and save the door.

Test It

Walk close to the door.

The default standalone interaction key is:

```
E
```

The default lockpick key is:

```
H
```

Lockpicking only appears when it is globally enabled and enabled for that specific door.

Confirm Persistence

After creating the door:

```
doors_storage
```

Then restart:

```
restart nord_doorlock
```

Open `/doors` again. The saved door should still exist.