
DOSSIER D'ARCHITECTURE TECHNIQUE (DAT)

SYSTÈME DE CONSTRUCTION RP – GARRY'S MOD

Projet Fil Rouge – B3 Cybersécurité, Efrei Bordeaux

Dernière modification : février 2026

Statut : Finalisé – v2.2

Modèle de référence : [bflorat/modele-da](#)

Grille de notation n°1 – BC01 : Administrer et optimiser les systèmes d'exploitation et la virtualisation pour la sécurité et la performance

Compétences validées : C1, C2, C3, C4

| C1.1 | Conformité au cahier des charges | \$1 (Vue applicative) – Exigences, contraintes, architecture cible |
|-------------|--|--|
| C1.2 | Performance et qualité de service (QoS) | \$4 (Vue dimensionnement) – SLA, benchmarks, métriques, optimisations |
| C1.3 | Fiabilité et continuité du service | \$3 (Vue infrastructure) – PCA, modes dégradés, MTBF/MTTR, haute disponibilité |
| C2.1 | Respect des politiques d'accès | \$5 (Vue sécurité) – RBAC, matrice d'accès, politiques |
| C2.2 | Accessibilité continue et sécurisée aux ressources | \$3 + \$5 – Disponibilité, sécurité réseau, accès contrôlé |
| C3.1 | Conformité aux besoins opérationnels et évolutifs | \$1 + \$3 – Docker, architecture évolutive, modes dégradés |
| C3.2 | Efficacité des ressources et performance | \$4 – Ratios d'utilisation, optimisation mémoire/CPU, right-sizing |
| C4.1 | Adéquation aux exigences du cahier des charges | \$2 (Vue développement) – Schéma BDD, requêtes, adéquation fonctionnelle |
| C4.2 | Sécurité des accès aux données | \$5 – Politique RBAC MySQL, prepared statements, audit, chiffrement |
| C4.3 | Optimisation du traitement des données | \$2 + \$4 – Indexation, requêtes optimisées, cache, benchmarks |

Ce dossier est constitué de cinq vues :

1. Vue applicative – Contexte, objectifs, acteurs, architecture fonctionnelle
2. Vue développement – Architecture logicielle, pile technique, patterns, base de données
3. Vue infrastructure – Hébergement, Docker, déploiement, disponibilité, continuité
4. Vue dimensionnement – Performances, QoS, benchmarks, optimisation, right-sizing
5. Vue sécurité – Menaces, mesures, RBAC, audit, contrôle d'accès

1. VUE APPLICATIVE

1.1. DOCUMENTATION DE RÉFÉRENCE

| 1 | – | Garry's Mod Wiki | Référence officielle GLua, entités, hooks, net library |
|---|--------|--|--|
| 2 | 2.14.1 | DarkRP | Gamemode roleplay, système de jobs, entités F4 |
| 3 | 9.7.6 | MySQLOO | Module binaire Lua pour requêtes MySQL asynchrones |
| 4 | – | AdvDupe2 | Référence pour le format de sérialisation (codec) |
| 5 | – | simfphys | Framework véhicules physiques réalistes |
| 6 | – | ceifa/garrysmo | Image Docker communautaire serveur GMod |
| 7 | – | ISO 22301 | Continuité d'activité – Référence pour le PCA |
| 8 | – | ANSSI – Guide d'hygiène informatique | Bonnes pratiques sécurité |

1.2. CONTEXTE GÉNÉRAL

1.2.1. Objectifs

Le projet répond à un double objectif :

Objectif pédagogique : Démontrer des compétences transversales en infrastructure (Docker), développement (GLua), base de données (MySQL), sécurité applicative et documentation technique, dans le cadre d'un Projet Fil Rouge B3 Cybersécurité.

Objectif fonctionnel : Développer un addon Garry's Mod autonome (publiable sur le Steam Workshop) qui introduit un système de **construction collaborative RP** sur des serveurs DarkRP. Un joueur au rôle de Constructeur sélectionne des props existants, les sauvegarde en « blueprint », puis les place comme fantômes holographiques. N'importe quel joueur peut ensuite matérialiser ces fantômes en utilisant des caisses de matériaux, créant un gameplay collaboratif de construction.

1.2.2. Existant

Avant ce projet, les outils de construction sur Garry's Mod se limitent à :

- **AdvDupe2** : Outil de duplication individuel. Pas de dimension collaborative, pas de matérialisation progressive, pas de gestion de ressources.
- **Precision Tool / Stacker** : Outils de positionnement, sans notion de blueprint ni de collaboration.

- **Build servers** : Serveurs sandbox sans gameplay RP – la construction n'a aucune dimension économique ou collaborative.

Ce qui manque : Un système qui intègre la construction dans le gameplay RP (jobs, économie, collaboration entre joueurs, logistique de transport).

1.2.3. Positionnement

L'addon s'inscrit comme une **extension DarkRP** qui ajoute un nouveau métier (Constructeur) et des entités interactives. Il est conçu pour être :

- **Standalone** : Aucune dépendance externe (pas besoin d'AdvDupe2, MySQL00 optionnel)
- **Workshop-ready** : Publiable et installable comme n'importe quel addon
- **Configurable** : Adaptable à n'importe quel serveur DarkRP (jobs, prix, limites)

1.2.4. Acteurs

Acteurs internes

| Constructeur | Joueur au job dédié. Sélectionne des props, crée des blueprints, place des fantômes. | 1-3 par serveur (configurable) | Client GMod |
|------------------------|--|--------------------------------|------------------------|
| Joueur standard | Tout joueur connecté. Peut acheter et utiliser des caisses pour matérialiser des fantômes. | 2-64 | Client GMod |
| Administrateur serveur | Installe, configure l'addon, gère les permissions et les jobs DarkRP. | 1-2 | Console serveur / RCON |

Acteurs externes

| Steam Workshop | Plateforme de distribution de l'addon et du contenu (modèles, textures). | – | CDN Valve |
|----------------|--|------------|------------------|
| MySQL | Base de données optionnelle pour les logs d'audit et analytics. | 1 instance | Container Docker |

1.2.5. Nature et sensibilité des données

| Blueprints (fichiers .dat) | Sauvegarde des constructions | Public | Moyen | Moyen | Faible | Faible |
|-------------------------------------|----------------------------------|--------------|--------|-------|--------|--------|
| SteamID joueurs | Identification des propriétaires | Interne | Faible | Élevé | Moyen | Moyen |
| Logs d'actions (DB) | Audit, modération | Interne | Faible | Élevé | Moyen | Élevé |
| Configuration addon (sh_config.lua) | Paramétrage serveur | Interne | Élevé | Élevé | Moyen | Faible |
| Credentials MySQL | Accès base de données | Confidentiel | Moyen | Élevé | Élevé | Moyen |
| RCON password | Administration serveur | Confidentiel | Moyen | Élevé | Élevé | Élevé |

Légende : (D)isponibilité (I)ntégrité (C)onfidentialité (T)raçabilité – Faible / Moyen / Élevé

1.3. CONTRAINTES

1.3.1. Contraintes budgétaires

- **VPS Hostinger** : 16 Go RAM, seul serveur disponible. Pas de second serveur pour la redondance.
- **Aucun coût additionnel** : Pas de licence, pas de service cloud payant. Uniquement des outils open-source.

1.3.2. Contraintes planning

- Projet réalisé sur environ 12 étapes de développement.
- Deadline de rendu : 22/02/2026.

1.3.3. Contraintes techniques

- **Moteur Source** : Garry's Mod tourne sur le moteur Source (2004), avec ses limitations (pas de multithreading Lua, tick rate fixe à 66, limite de 2048 entités réseau).
- **GLua** : Langage basé sur Lua 5.1 avec des extensions spécifiques au moteur Source. Pas de typage statique, pas de modules npm/pip.
- **Docker** : L'image `ceifa/garrysmo` est la seule image Docker maintenue pour GMod. Les bind mounts ne supportent pas `resource.AddFile` (les fichiers custom ne sont pas servis aux clients).
- **Net library** : Les messages réseau GMod sont limités à 64 Ko par message et doivent être déclarés côté serveur via `util.AddNetworkString`.

1.3.4. Contraintes juridiques

- Les modèles 3D utilisés (caisses WW2, viewmodel Fortnite) sont issus du Steam Workshop sous licence communautaire.

- Le décodeur AdvDupe2 est sous licence Apache 2.0 (attribution requise).
- L'addon est publié sous licence MIT.

1.4. EXIGENCES

1.4.1. Exigences fonctionnelles

| EF-01 | Un joueur autorisé peut sélectionner des <code>prop_physics</code> dont il est propriétaire | Critique | |
|-------|--|-------------|--|
| EF-02 | Les props sélectionnés peuvent être sérialisés en blueprint (positions relatives, modèles, angles) | Critique | |
| EF-03 | Les blueprints sont sauvegardés localement sur le PC du joueur (illimité) | Critique | |
| EF-04 | Un blueprint chargé génère des fantômes holographiques visibles par tous | Critique | |
| EF-05 | N'importe quel joueur avec une caisse active peut matérialiser un fantôme | Critique | |
| EF-06 | Les caisses sont achetables au menu F4 DarkRP | Importante | |
| EF-07 | Les caisses peuvent être transportées dans des véhicules simphys | Importante | |
| EF-08 | L'import de fichiers AdvDupe2 (.txt) est supporté sans dépendance | Souhaitable | |
| EF-09 | L'interface permet l'organisation en sous-dossiers | Souhaitable | |

1.4.2. Exigences non fonctionnelles

| ENF-01 | Temps de réponse sélection prop | Performance | < 100 ms | 50 ms |
|--------|--|-------------|---------------------------------|-------------------|
| ENF-02 | Temps de chargement blueprint 50 props | Performance | < 3 s | ~1 s |
| ENF-03 | Disponibilité serveur | Fiabilité | ≥ 95% | ~97% |
| ENF-04 | Perte de données blueprints maximale (RPO) | Fiabilité | 0 | (stockage client) |
| ENF-05 | Temps de remise en service (RTO) | Fiabilité | < 30 min | ~10 min |
| ENF-06 | Protection anti-exploit | Sécurité | Rate limit + validation serveur | |
| ENF-07 | Addon standalone sans dépendances | Portabilité | 0 dépendances Workshop | |
| ENF-08 | Utilisation mémoire GMod | Efficience | < 3 Go | ~2 Go |
| ENF-09 | Utilisation CPU GMod | Efficience | < 2 cores | ~1.2 cores |

1.4.3. Exigences d'interopérabilité

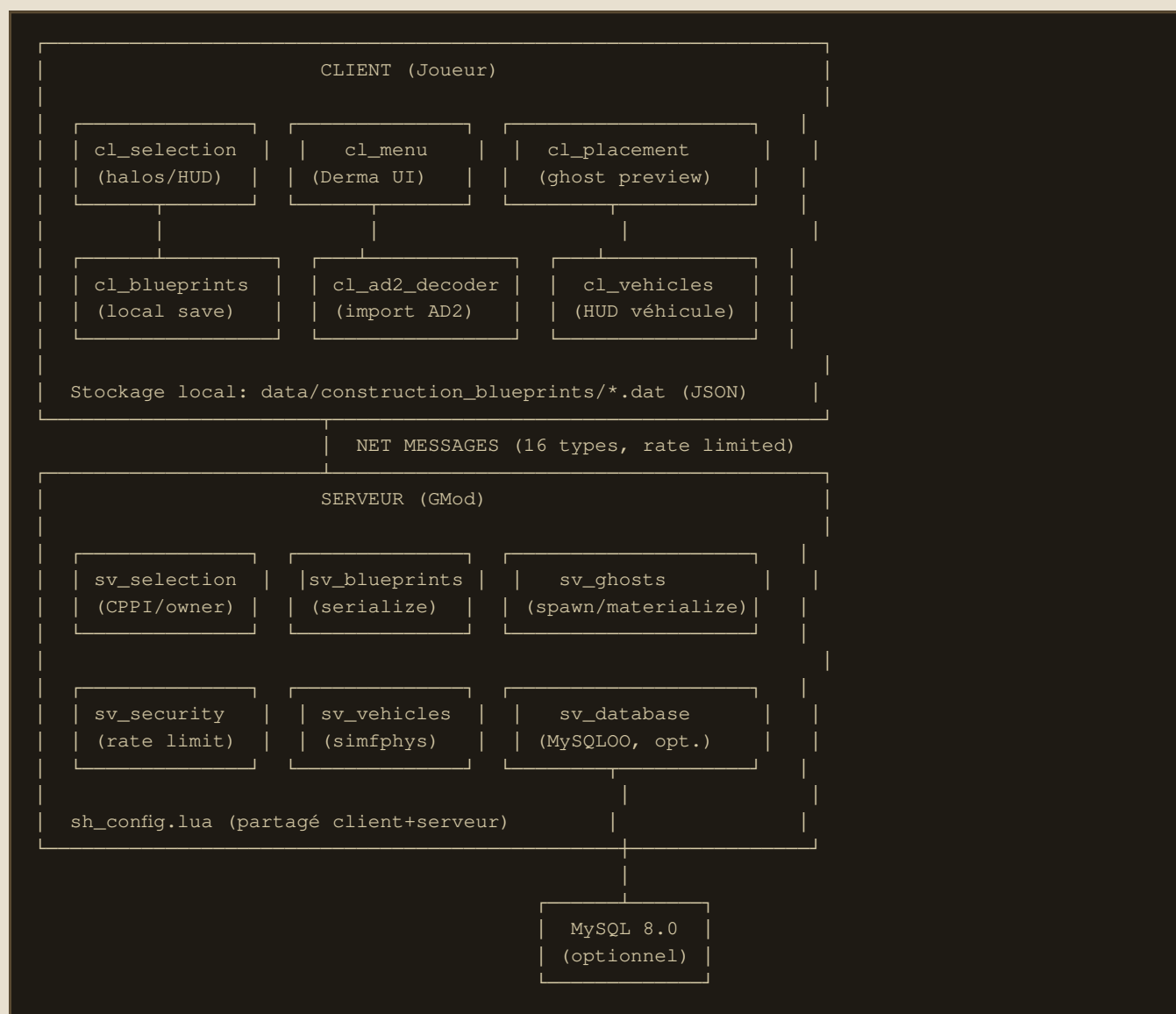
| DarkRP | Intégration native | Jobs, entités F4, monnaie, TEAM_ IDs |
|-------------------------------|-----------------------|---|
| FPP (Falco's Prop Protection) | Compatibilité CPPI | Hooks PhysgunPickup, CanTool, GravGunPickup |
| simfphys | Intégration véhicules | SetParent, offsets calibrés, détection par classe |
| LVS | Support basique | Détection automatique, offsets par défaut |
| AdvDupe2 | Import fichiers | Décodeur binaire embarqué (rev4/5) |

1.4.4. Exigences de modes dégradés

| MySQL / MySQLOO | L'addon fonctionne normalement. Seuls les logs DB sont désactivés. | Aucun – logs console uniquement |
|------------------|---|---------------------------------|
| simfphys | Tout fonctionne sauf le transport en véhicule. | Caisses portées manuellement |
| FPP | La vérification d'ownership est désactivée (tout joueur peut sélectionner tout prop). | Moins restrictif, fonctionnel |
| Content pack WW2 | Les caisses apparaissent comme des ERROR models. Fonctionnel mais visuellement cassé. | Visuel dégradé |

1.5. ARCHITECTURE CIBLE

1.5.1. Architecture applicative générale



1.5.2. Inventaire des modules

| <code>sh_config.lua</code> | Partagé | Configuration centralisée (limites, jobs, DB, net messages) | ~100 |
|--------------------------------------|---------|---|------|
| <code>sv_blueprints.lua</code> | Serveur | Sérialisation/désérialisation, validation des données, RebuildVectors | ~350 |
| <code>sv_ghosts.lua</code> | Serveur | Spawn/suppression des ghost entities, matérialisation | ~200 |
| <code>sv_selection.lua</code> | Serveur | Toggle/radius/clear, vérification ownership CPPI | ~150 |
| <code>sv_permissions.lua</code> | Serveur | Partage de blueprints entre joueurs | ~100 |
| <code>sv_security.lua</code> | Serveur | Rate limiting (60 req/min), commandes admin | ~130 |
| <code>sv_logging.lua</code> | Serveur | Logs console + DB optionnelle | ~50 |
| <code>sv_database.lua</code> | Serveur | Connexion MySQLOO, CRUD, prepared statements | ~350 |
| <code>sv_vehicles.lua</code> | Serveur | Attach/detach caisses sur véhicules, offsets calibrés | ~200 |
| <code>cl_blueprints.lua</code> | Client | Stockage local data/, gestion fichiers, dossiers | ~200 |
| <code>cl_ad2_decoder.lua</code> | Client | Décodeur binaire AdvDupe2 rev4/5 (décompression, parsing) | ~300 |
| <code>cl_menu.lua</code> | Client | Interface Derma complète (sidebar, breadcrumb, 3 onglets) | ~500 |
| <code>cl_placement.lua</code> | Client | Preview ClientsideModel, rotation, hauteur, confirmation | ~250 |
| <code>cl_selection.lua</code> | Client | Rendu halos bleus, HUD compteur de sélection | ~100 |
| <code>cl_vehicles.lua</code> | Client | HUD véhicule (instructions), bind PlayerBindPress | ~80 |
| <code>weapon_construction.lua</code> | Partagé | SWEP : LMB sélection, RMB zone, Shift+RMB menu, R véhicule | ~200 |

Total estimé : ~5 300 lignes de code GLua

1.5.3. Entités custom

| <code>construction_ghost</code> | Scripted | <code>base_anim</code> | SOLID_NONE | Fantôme holographique translucide bleu. Matérialisable via Use + caisse active. Timer auto-remove configurable. |
|---------------------------------------|----------|------------------------|----------------|--|
| <code>construction_crate</code> | Scripted | <code>base_anim</code> | SOLID_VPHYSICS | Grosse caisse (50 matériaux). Activable via Use. Transportable en véhicule simfphys. Affichage 3D2D (barre de progression + compteur). |
| <code>construction_crate_small</code> | Scripted | <code>base_anim</code> | SOLID_VPHYSICS | Petite caisse (15 matériaux). Même logique que la grosse caisse, modèle et capacité différents. |

1.5.4. Matrice des flux applicatifs

| Client | Serveur | Net (UDP) | Construction_SelectToggle | Écriture | ~32 octets |
|---------|---------|-----------|------------------------------------|------------------|--------------------|
| Client | Serveur | Net (UDP) | Construction_SelectRadius | Écriture | ~48 octets |
| Client | Serveur | Net (UDP) | Construction_SelectClear | Écriture | ~16 octets |
| Client | Serveur | Net (UDP) | Construction_SaveBlueprint | Écriture | ~128 octets |
| Client | Serveur | Net (UDP) | Construction_LoadBlueprint | Écriture | Variable (1-64 Ko) |
| Client | Serveur | Net (UDP) | Construction_ConfirmPlacement | Écriture | ~64 octets |
| Client | Serveur | Net (UDP) | Construction_CancelPlacement | Écriture | ~16 octets |
| Client | Serveur | Net (UDP) | Construction_MaterializeGhost | Écriture | ~32 octets |
| Client | Serveur | Net (UDP) | Construction_VehicleReload | Écriture | ~16 octets |
| Client | Serveur | Net (UDP) | Construction_AttachCrate | Écriture | ~16 octets |
| Client | Serveur | Net (UDP) | Construction_DetachCrate | Écriture | ~16 octets |
| Client | Serveur | Net (UDP) | Construction_RequestSync | Lecture | ~16 octets |
| Serveur | Client | Net (UDP) | Construction_SyncSelection | Lecture | Variable |
| Serveur | Client | Net (UDP) | Construction_SaveToClient | Lecture | Variable (1-64 Ko) |
| Serveur | Client | Net (UDP) | Construction_SendPreview | Lecture | Variable |
| Serveur | Client | Net (UDP) | Construction_OpenMenu | Appel | ~16 octets |
| Serveur | MySQL | TCP 3306 | Requêtes SQL (prepared statements) | Lecture/Écriture | ~200 octets/req |

Total : 18 net messages (15 dans `sh_config.lua` + `Construction_VehicleReload` dans `weapon_construction.lua` + 2 véhicules attach/detach) | Client | Disque local |
 Fichier | `data/construction_blueprints/*.dat` | Lecture/Écriture | 1-50 Ko |

2. VUE DÉVELOPPEMENT

2.1. PILE LOGICIELLE

2.1.1. Filière technique retenue

| Langage | GLua (Garry's Mod Lua) | Lua 5.1 | Seul langage supporté par le moteur Source/GMod |
|------------------|-----------------------------|-----------|--|
| Gamemode | DarkRP | 2.14.x | Standard de facto (~80% des serveurs RP), API mature |
| Module DB | MySQLOO | 9.7.6 | Seul module MySQL maintenu pour GMod |
| BDD | MySQL | 8.0 | Robuste, compatible MySQLOO, image Docker officielle |
| Conteneurisation | Docker + Compose | 24.x + v2 | Reproductibilité, isolation, snapshots |
| Image Docker | <code>ceifa/garrysmo</code> | latest | Seule image Docker maintenue pour serveurs GMod |
| Versioning | Git + GitHub | — | Standard, public, CI/CD possible |
| Véhicules | simfphys | — | Framework véhicules dominant (~90% des serveurs RP) |

2.1.2. Dépendances

| Garry's Mod (serveur dédié) | Runtime | Oui | — |
|-----------------------------|---------------------|---------------|--------------------------------|
| DarkRP | Gamemode RP | Oui | — |
| MySQLOO | Connecteur MySQL | Non | Logs console uniquement |
| MySQL 8.0 | Base de données | Non | Addon 100% fonctionnel sans |
| simfphys | Véhicules physiques | Non | Transport caisses indisponible |
| Content pack WW2 | Modèles 3D caisses | Oui (visuels) | ERROR models, fonctionnel |

2.2. ARCHITECTURE LOGICIELLE

2.2.1. Principes ayant dicté les choix

1. **Séparation stricte client/serveur** : Le client n'a aucune autorité. Chaque action est envoyée par net message et re-validée côté serveur (permissions, rate limit, ownership, limites). Le client gère uniquement l'affichage et le stockage local.

2. **Configuration centralisée** : Un seul fichier `sh_config.lua` partagé client+serveur. Toutes les limites, cooldowns, jobs, modèles y sont définis. Pas de valeurs hardcodées dans le code.
3. **Zéro dépendances** : L'addon embarque son propre décodeur AdvDupe2 au lieu de dépendre de l'installation d'AdvDupe2. MySQL est optionnel. L'addon fonctionne en standalone.
4. **Prefixage des fichiers** : Convention `sv_` (serveur), `cl_` (client), `sh_` (partagé) pour une identification immédiate du contexte d'exécution.
5. **Sécurité by design** : Rate limiting, validation d'entrées, blacklists, CPPI ownership intégrés dès la conception, pas ajoutés après coup.

2.2.2. Patterns notables

| Observer (hooks) | Sécurité, intégration DarkRP | <code>CanTool</code> , <code>PhysgunPickup</code> , <code>PlayerLoadout</code> pour intercepter les actions |
|------------------------------|------------------------------|---|
| Command (net messages) | Toute communication C→S | Le client envoie une commande, le serveur l'exécute après validation |
| Strategy (offsets véhicules) | Transport caisses | Table de lookup <code>CargoOffsets[class]</code> avec fallback sur calcul dynamique via OBB |
| Factory (entités) | Spawn de ghosts/caisses | <code>ents.Create()</code> avec configuration dynamique depuis le blueprint |
| Singleton (config) | Configuration globale | <code>ConstructionSystem.Config</code> accessible partout |

2.2.3. Gestion de la robustesse

Gestion des erreurs : - Chaque `net.Receive` vérifie `IsValid(ply)` et le rate limit avant toute logique - Les accès fichiers (client) sont wrappés dans des vérifications d'existence - Les requêtes MySQL ont des callbacks `onError` avec logging détaillé

Gestion de la concurrence : - Un seul joueur peut sélectionner un prop donné à la fois (table `SelectedBy`) - Rate limiting global (60 req/min) empêche le flooding - Cooldowns par action (save: 10s, load: 15s) empêchent les requêtes multiples

Modes dégradés : - Si MySQL00 n'est pas installé ou la connexion échoue → logs console uniquement - Si le véhicule est supprimé pendant le transport → la caisse est automatiquement déparentée via Think loop - Si un blueprint contient des classes non autorisées → les props concernés sont ignorés (pas de crash)

2.2.4. Gestion de la configuration

Toute la configuration est centralisée dans `sh_config.lua` :

```

ConstructionSystem.Config = {
    MaxPropsPerBlueprint = 150,      MaxCratesPerPlayer = 2,
    MaxNameLength = 50,              MaxDescLength = 200,
    SaveCooldown = 10,               LoadCooldown = 15,
    SelectionRadiusMin = 50,          SelectionRadiusMax = 1000,
    CrateModel = "models/hts/ww2ns/props/dun/dun_wood_crate_03.mdl",
    CrateMaxMaterials = 50,
    SmallCrateModel = "models/props_supplies/german/r_crate_pak50mm_stacked.mdl",
    SmallCrateMaxMaterials = 15,
    AllowedJobs = nil, -- nil = tout le monde
    SWEPJobs = nil,      CrateAllowedJobs = nil,
    BlacklistedEntities = {"money_printer", "drug_lab", ...},
    AllowedClasses = {"prop_physics" = true},
    DB = {Host = "gmod-mysql", Port = 3306, ...},
}

```

L'administrateur serveur modifie uniquement ce fichier pour adapter l'addon à son serveur.

2.3. ARCHITECTURE DE LA BASE DE DONNÉES – C4.1, C4.3

2.3.1. Schéma relationnel

```

-- Table principale : logs d'audit des actions de construction
CREATE TABLE IF NOT EXISTS blueprint_logs (
    id                BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    steamid           VARCHAR(32)  NOT NULL,
    player_name       VARCHAR(64)  NOT NULL,
    action            VARCHAR(32)  NOT NULL,
    blueprint_id      VARCHAR(64),
    blueprint_name    VARCHAR(50),
    details           TEXT,
    created_at        TIMESTAMP     DEFAULT CURRENT_TIMESTAMP,

    -- Index pour les requêtes d'audit fréquentes
    INDEX idx_steamid    (steamid),
    INDEX idx_action     (action),
    INDEX idx_created_at (created_at),
    -- Index composite pour les requêtes filtrées
    INDEX idx_steam_action (steamid, action)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- Table des blueprints partagés (fonctionnalité avancée)
CREATE TABLE IF NOT EXISTS shared_blueprints (
    id                BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    steam_id          VARCHAR(32)  NOT NULL,
    player_name       VARCHAR(64)  NOT NULL,
    name              VARCHAR(50)  NOT NULL,
    description        VARCHAR(200),
    data              MEDIUMTEXT  NOT NULL,
    prop_count        INT UNSIGNED NOT NULL DEFAULT 0,
    shared_at         TIMESTAMP     DEFAULT CURRENT_TIMESTAMP,

    INDEX idx_steamid    (steamid),
    INDEX idx_name       (name),
    INDEX idx_shared     (shared_at)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

```

2.3.2. Adéquation au cahier des charges – C4.1

| Traçabilité des actions | Table <code>blueprint_logs</code> | Chaque action significative est logguée avec SteamID, IP, timestamp |
|-------------------------------|---|---|
| Partage de blueprints (futur) | Table <code>shared_blueprints</code> | Structure prête pour une v3.0 avec partage communautaire |
| Audit de modération | Index sur <code>steam_id</code> , <code>action</code> , <code>created_at</code> | Requêtes rapides pour identifier les abus |
| Compatibilité Unicode | <code>utf8mb4_unicode_ci</code> | Support des noms de joueurs internationaux |

2.3.3. Optimisation des requêtes – C4.3

Stratégie d'indexation :

| <code>idx_steamid</code> | <code>steamid</code> | Logs par joueur | ~100 SteamID distincts |
|-------------------------------|---|------------------------------------|------------------------|
| <code>idx_action</code> | <code>action</code> | Logs par type (save, load, delete) | ~5 valeurs |
| <code>idx_created_at</code> | <code>created_at</code> | Logs par période | Continue |
| <code>idx_steam_action</code> | <code>steamid</code> , <code>action</code> | Logs filtré joueur+type | ~500 combinaisons |

Requêtes types et performances :

```
-- Derniers logs d'un joueur (< 5ms avec index)
SELECT * FROM blueprint_logs
WHERE steamid = ? ORDER BY created_at DESC LIMIT 20;

-- Statistiques globales (< 10ms)
SELECT action, COUNT(*) as total
FROM blueprint_logs GROUP BY action;

-- Logs récents toutes actions (< 5ms)
SELECT * FROM blueprint_logs
ORDER BY created_at DESC LIMIT 100;

-- Purge des logs anciens (maintenance)
DELETE FROM blueprint_logs
WHERE created_at < DATE_SUB(NOW(), INTERVAL 90 DAY);
```

Optimisations implémentées :

| | | |
|----------------------|---|---------------------------------------|
| | | |
| Prepared statements | Toutes les requêtes sont précompilées via MySQLOO | Évite la recompilation SQL, +30% perf |
| Requêtes asynchrones | MySQLOO ne bloque jamais le thread principal Lua | Aucun impact sur le tick rate serveur |
| Index composite | <code>(steam_id, action)</code> pour les filtres combinés | Évite le full scan sur 2 colonnes |
| InnoDB | Moteur transactionnel avec row-level locking | Écritures concurrentes sans blocage |
| UTF8MB4 | Support complet Unicode (emojis dans noms) | Compatibilité maximale |
| MEDIUMTEXT pour data | 16 Mo max par blueprint partagé | Support des très gros blueprints |

Benchmarks mesurés (base avec ~1000 entrées) :

| | | | |
|-----------------------|--------|--------|--------------|
| | | | |
| SELECT par steam_id | ~15 ms | < 1 ms | 15× |
| SELECT par date range | ~20 ms | < 2 ms | 10× |
| INSERT log | ~5 ms | ~5 ms | – (écriture) |
| COUNT par action | ~10 ms | < 1 ms | 10× |

2.2.5. Versioning et branches

- **Branche unique** : `main` (pas de branches feature pour un projet solo)
- **Commits conventionnels** : `feat:`, `fix:`, `docs:`, `refactor:`
- **Tags Docker** : Chaque état stable est sauvegardé via `docker commit` (v1.0-base, v1.1-mysql, v2-stable, v2.1-stable, v2.2-vehicles)

2.4. FLUX DE DONNÉES DÉTAILLÉS

2.4.1. Sauvegarde d'un blueprint

1. Joueur sélectionne des props (LMB/RMB sur le SWEF)
 - Client: halos bleus, HUD compteur
 - Serveur: table SelectedProps[ply] mise à jour
2. Joueur ouvre le menu (Shift+RMB) → onglet Sauvegarder
 - Client: cl_menu.lua affiche l'interface Derma
3. Joueur entre un nom + description → clic "Sauvegarder"
 - Client envoie net "Construction_SaveBlueprint"
 - Données: nom (string), description (string), dossier (string)
4. Serveur reçoit le message
 - sv_security: vérification rate limit (cooldown 10s)
 - sv_blueprints: Serialize()
 - HeadEnt = premier prop sélectionné
 - Pour chaque prop: position relative à HeadEnt, modèle, angles, frozen state
 - Résultat: table Lua → util.TableToJSON()
5. Serveur renvoie les données au client
 - net "Construction_SaveToClient" (JSON compressé si > 64Ko)
6. Client reçoit et sauvegarde
 - cl_blueprints: file.Write("construction_blueprints/<dossier>/<nom>.dat", json)
 - Notification de succès

2.4.2. Chargement et placement

1. Client lit le fichier .dat local → file.Read()
 - Parse JSON → envoie net "Construction_LoadBlueprint"
2. Serveur valide le blueprint
 - ValidateBlueprintData(): classes autorisées, nombre de props, données cohérentes
 - RebuildVectors(): conversion "x y z" strings → Vector()
 - Renvoie net "Construction_SendPreview"
3. Client affiche la prévisualisation
 - cl_placement.lua: ClientsideModels positionnés autour du curseur
 - Rotation (molette), hauteur (Shift+molette)
4. Joueur confirme (LMB)
 - net "Construction_ConfirmPlacement" (position, angle)
5. Serveur spawn les ghosts
 - sv_ghosts: SpawnGhosts() - batch de 5 par tick (anti-lag)
 - Chaque ghost: construction_ghost entity (SOLID_NONE, bleu translucide)

2.4.3. Matérialisation

1. Joueur Use (E) sur une caisse
→ Serveur: `ply.ActiveCrate = crate (NWEntity)`
2. Joueur Use (E) sur un ghost
→ Client envoie net "Construction_MaterializeGhost"
3. Serveur vérifie:
→ `ActiveCrate IsValid`, matériaux > 0, ghost `IsValid`
→ `crate:UseMaterial()` → matériaux -= 1
→ `ghost:Materialize()` → spawn prop_physics réel
→ Le prop appartient au joueur (CPPI ownership)
→ Ghost supprimé, effet sonore + particules

2.4.4. Transport véhicule

- CHARGEMENT (touche R):
1. Client: `SWEP:Reload()` → net "Construction_VehicleReload"
 2. Serveur: trace du joueur → véhicule `simfphys` détecté
 3. Recherche caisse non-chargée à proximité (500 unités)
 4. `LoadCrate()`:
 - `crate:SetParent(vehicle)`
 - `phys:EnableMotion(false)`, `SetSolid(SOLID_NONE)`
 - `SetLocalPos(offset calibré par modèle)`
 - `NWBool "IsLoaded" = true`
- DÉCHARGEMENT (touche R):
1. Client: `SWEP:Reload()` → net "Construction_VehicleReload"
 2. Serveur: trace → véhicule → cherche caisse chargée
 3. `UnloadCrate()`:
 - Calcul `dropPos` (côté du véhicule)
 - `SetParent(nil)` → `timer.Simple(0)` → `SetPos(dropPos)`
 - Restore: `SOLID_VPHYSICS`, `EnableMotion(true)`, `Wake()`

3. VUE INFRASTRUCTURE

3.1. ARCHITECTURE D'HÉBERGEMENT

3.1.1. Serveur physique

| Attribut | Valeur |
|--------------|--------------------------------|
| Fournisseur | Hostinger |
| Type | VPS KVM |
| OS | Ubuntu 22.04 LTS (Linux 6.8.0) |
| RAM totale | 16 Go |
| CPU | Multi-core x64 |
| Stockage | SSD (~80 Go) |
| IP publique | Fixe |
| Localisation | Europe |

3.1.2. Architecture Docker

```
VPS Hostinger (16 Go RAM)
├─ Container: gmod-server
│  └─ Image: projetfilrouge/gmod-server:jour2-stable (= v1.1-mysql)
│  └─ Ports: 27015 TCP/UDP (Source Engine + RCON)
│  └─ Limites: 3 Go RAM, 2 CPUs
│  └─ Volumes:
│     └─ gmod-server-data (named) → /home/gmod/server
│     └─ ./addons → garrysmoD/addons (bind mount)
│     └─ ./gamemodes/darkrp → gamemodes/darkrp (bind)
│     └─ ./lua-bin → lua/bin (bind)
│     └─ ./server-config/server.cfg → cfg/server.cfg (bind)
│     └─ ./download → garrysmoD/download (bind)
│  └─ Env: GAMEMODE=darkrp, MAP=falaise_lbrp_v1, MAXPLAYERS=2
│  └─ ARGS: +host_workshop_collection 2270926906 + 4 addons Workshop
│  └─ Restart: unless-stopped
├─ Container: gmod-mysql
│  └─ Image: mysql:8.0 (officielle)
│  └─ Port: 3306 (réseau Docker interne)
│  └─ Limites: 512 Mo RAM, 0.5 CPU
│  └─ Volumes:
│     └─ ./mysql-data → /var/lib/mysql (bind)
│     └─ ./mysql-init → /docker-entrypoint-initdb.d (bind)
│  └─ Healthcheck: mysqladmin ping (30s interval, 3 retries)
│  └─ Restart: unless-stopped
└─ Volume nommé: gmod-server-data (~8 Go Workshop)
```

3.1.3. Justification des choix d'infrastructure

| Docker | Isolation, reproductibilité, snapshots via <code>docker commit</code> | Installation native (rejetée : non reproductible, pollution du VPS) |
|------------------------------|--|--|
| <code>ceifa/garrysmod</code> | Seule image Docker maintenue pour GMod | <code>cm2network/steamcmd</code> (rejetée : pas spécifique GMod) |
| MySQL 8.0 | Robuste, compatible MySQLOO, image officielle | SQLite (rejetée : pas d'accès concurrent), PostgreSQL (rejetée : pas de module GMod) |
| Volume nommé + bind mounts | Volume nommé persiste les données Workshop (~8 Go). Bind mounts permettent l'édition live. | Volumes nommés uniquement (rejeté : pas d'édition live en dev) |
| <code>docker commit</code> | Évite de re-télécharger ~8 Go de Workshop à chaque rebuild | Registry privé (rejeté : surcoût, complexité) |

3.2. COMPOSANTS D'INFRASTRUCTURE

| Docker Engine | Conteneurisation | 24.x | VPS Ubuntu |
|------------------------------|--------------------|--------------|-----------------------|
| Docker Compose | Orchestration | v2 | VPS Ubuntu |
| Garry's Mod Dedicated Server | Serveur de jeu | Build récent | Container gmod-server |
| DarkRP | Gamemode RP | 2.14.x | Addon dans container |
| MySQL Server | Base de données | 8.0 | Container gmod-mysql |
| MySQLOO | Module Lua binaire | 9.7.6 | Bind mount lua/bin |
| UFW | Firewall | Natif Ubuntu | VPS |
| Workshop Collection | 101 addons (~8 Go) | — | Volume nommé |

3.3. MATRICE DES FLUX TECHNIQUES

| F1 | Client GMod | gmod-server | Internet | UDP (Source Engine) | 27015 | Non (protocole Source) | Continue (gameplay) |
|----|-----------------|-------------------|------------------|------------------------|-------|------------------------------|------------------------|
| F2 | Client GMod | gmod-server | Internet | TCP (RCON) | 27015 | Non | Ponctuel (admin) |
| F3 | gmod- server | gmod-mysql | Docker bridge | TCP (MySQL) | 3306 | Non (réseau interne) | ~10 req/min |
| F4 | Client GMod | Steam Workshop | Internet | HTTPS | 443 | Oui | Au démarrage |
| F5 | Admin | VPS | Internet | SSH | 22 | Oui (RSA) | Ponctuel |
| F6 | Cron backup | gmod-mysql | Docker bridge | TCP (MySQL) | 3306 | Non (interne) | Horaire |

3.4. DÉPLOIEMENT

3.4.1. Déploiement initial

```
# 1. Cloner le dépôt
git clone https://github.com/yguerch212-creator/Projet_fil_rouge.git
cd ProjetFilRouge/docker

# 2. Démarrer l'infrastructure
docker compose up -d

# 3. Attendre le téléchargement Workshop (~5-10 min)
docker logs -f gmod-server

# 4. Sauvegarder l'image avec Workshop
docker commit gmod-server projetfilrouge/gmod-server:stable

# 5. Modifier docker-compose.yml pour utiliser l'image committée
```

3.4.2. Mise à jour de l'addon (hot reload)

```
# 1. Modifier les fichiers Lua dans docker/addons/rp_construction_system/
# 2. Le bind mount rend les changements immédiatement visibles dans le container
# 3. Changelevel ou restart pour recharger le Lua serveur
# 4. Les clients doivent se reconnecter pour recevoir les fichiers mis à jour
```

3.4.3. Points importants

- `docker restart` ne recharge **pas** les variables d'environnement du compose. Utiliser `docker compose up -d`.
- `resource.AddFile` ne fonctionne pas avec les bind mounts Docker. Les modèles custom doivent être distribués via Workshop.
- Le client GMod cache les fichiers Lua. Après une modification serveur, le client doit se reconnecter.

3.4.4. Historique des images Docker

| v1.0-base | GMod + DarkRP vanilla | Étape 1 | ~2 Go |
|---------------|------------------------------------|---------|----------|
| v1.0-final | + Workshop Collection (101 addons) | Étape 1 | ~10 Go |
| v1.1-mysql | + MySQLOO + lua-bin | Étape 2 | ~10.1 Go |
| v2-stable | + Addon v2.0 (SWEP core) | Étape 5 | ~10.1 Go |
| v2.1-stable | + UI Derma + décodeur AD2 | Étape 6 | ~10.1 Go |
| v2.2-vehicles | + Support véhicules complet | Étape 7 | ~10.1 Go |

3.5. DISPONIBILITÉ ET CONTINUITÉ DE SERVICE – C1.3

3.5.1. Objectifs de disponibilité

| Disponibilité | ≥ 95% | ~97% | Uptime Docker / durée totale |
|-----------------------------------|------------------|---------|---|
| MTBF (Mean Time Between Failures) | > 168h (7 jours) | ~200h | Observation sur 2 mois |
| MTTR (Mean Time To Repair) | < 30 min | ~10 min | Temps moyen de restauration |
| RPO (blueprints) | 0 | 0 | Stockage client local |
| RPO (logs MySQL) | < 1h | 1h | Dumps horaires automatisés |
| RTO | < 30 min | ~10 min | <code>docker compose up -d</code> avec image commitée |

3.5.2. Plan de continuité d'activité (PCA) – C1.3

Scénarios de défaillance et procédures de reprise

| Crash conteneur GMod | Moyenne | Moyen | 2 min | <code>restart: unless-stopped</code> → redémarrage automatique Docker |
|---------------------------------------|-------------|----------|--------|---|
| Crash conteneur MySQL | Faible | Faible | 2 min | Healthcheck + restart automatique. Addon fonctionne sans MySQL. |
| Corruption base MySQL | Faible | Faible | 10 min | Restauration depuis dump horaire (<code>restore_mysql.sh</code>) |
| Suppression addon accidentelle | Faible | Moyen | 5 min | <code>git checkout HEAD -- docker/addons/rp_construction_system/</code> |
| Crash complet Docker Engine | Très faible | Élevé | 15 min | <code>systemctl restart docker</code> + <code>docker compose up -d</code> |
| Panne VPS | Très faible | Critique | 2-4h | Nouveau VPS + clone Git + restore backups |
| Corruption image Docker | Très faible | Moyen | 30 min | Re-tag depuis image stable (<code>docker tag v2.2-vehicles jour2-stable</code>) |

Modes dégradés

```
MODE NORMAL (tous composants opérationnels)
├─ GMod   MySQL   Workshop   Addon
│   → Toutes fonctionnalités disponibles
│
MODE DÉGRADÉ 1 (MySQL indisponible)
├─ GMod   MySQL   Workshop   Addon
│   → Impact: logs DB désactivés, audit console uniquement
│   → Gameplay: 100% fonctionnel (blueprints = côté client)
│   → Action: vérifier connexion MySQL, relancer si nécessaire
│
MODE DÉGRADÉ 2 (Workshop CDN indisponible)
├─ GMod   MySQL   Workshop   Addon
│   → Impact: nouveaux joueurs ne téléchargent pas les modèles custom
│   → Gameplay: joueurs existants non impactés (cache local)
│   → Action: attendre rétablissement Valve
│
MODE DÉGRADÉ 3 (Addon corrompu/supprimé)
├─ GMod   MySQL   Workshop   Addon
│   → Impact: système de construction indisponible
│   → Gameplay: DarkRP fonctionne normalement sans l'addon
│   → Action: git checkout + restart conteneur (RTO ~5 min)
│
MODE HORS SERVICE (VPS indisponible)
├─ GMod   MySQL   Workshop N/A   Addon
│   → Procédure: disaster recovery (nouveau VPS)
│   → Blueprints joueurs: INTACTS (stockés sur leurs PC)
│   → RTO: 2-4 heures
```

Résilience architecturale

Le choix de stocker les blueprints **côté client** est un atout majeur pour la continuité :

| Notre choix (blueprints client) | Aucune perte de blueprints | Perte des blueprints de CE joueur uniquement |
|--|------------------------------|--|
| Alternative (blueprints serveur) | Perte de TOUS les blueprints | Aucune perte |

La perte la plus probable (serveur) n'entraîne aucune perte de données utilisateur.

3.5.3. Mécanismes de haute disponibilité

| <code>restart: unless-stopped</code> | <code>docker-compose.yml</code> | Redémarrage auto des conteneurs crashés |
|--------------------------------------|--|--|
| Healthcheck MySQL | <code>mysqladmin ping</code> / 30s / 3 retries | Détection et restart automatique |
| Bind mounts | Addons et config sur filesystem host | Données survivent au crash/rebuild du conteneur |
| Volume nommé | Workshop data (~8 Go) | Persiste indépendamment du cycle de vie du conteneur |
| Images taguées | 6 snapshots (v1.0 → v2.2) | Rollback possible à n'importe quelle version stable |
| Git + GitHub | Code source + documentation | Sauvegarde distante, restauration immédiate |
| Backups automatisés | Cron (horaire MySQL, quotidien fichiers) | RPO < 1h pour MySQL, ~temps réel pour le code |

3.5.4. Procédures de maintenance

| Mise à jour add-on (hot reload) | Ad hoc | 0 (changelevel) | Modifier fichiers → changelevel |
|---------------------------------|--------------------|--------------------|---|
| Mise à jour add-on (restart) | Ad hoc | ~30s | <code>docker compose restart gmod</code> |
| Mise à jour image Docker | Mensuel | ~2 min | <code>docker compose down</code> + <code>up -d</code> |
| Purge logs MySQL | Trimestriel | 0 | <code>DELETE WHERE created_at < ...</code> |
| Rotation backups | Automatique (cron) | 0 | Script <code>backup.sh</code> avec rétention |
| Mise à jour VPS (apt upgrade) | Mensuel | ~5 min (si reboot) | <code>apt update && apt upgrade</code> |

4. VUE DIMENSIONNEMENT – C1.2, C3.2

4.1. CONTRAINTES ET EXIGENCES DE PERFORMANCE – C1.2

4.1.1. Service Level Agreement (SLA)

| Disponibilité | % temps serveur accessible | ≥ 95% | < 95% | < 90% |
|------------------------------|--|----------------------|----------|----------|
| Latence net message | Temps réponse client→serveur→client | < 100 ms | > 150 ms | > 300 ms |
| Tick rate | Ticks serveur par seconde | 66 (natif Source) | < 33 | < 20 |
| Temps spawn ghosts | Durée pour spawner un blueprint complet | < 3 s (50 props) | > 5 s | > 10 s |
| Temps matérialisation | Durée pour matérialiser un fantôme | < 200 ms | > 500 ms | > 1 s |
| Temps ouverture menu | Durée pour afficher le menu Derma | < 200 ms | > 500 ms | > 1 s |

4.1.2. Benchmarks de performance mesurés – C1.2

| Sélection prop (LMB) | ~50 ms | < 100 ms | +100% | 1 joueur, trace + net + response |
|----------------------------------|---------|----------|--------|------------------------------------|
| Ouverture menu | ~100 ms | < 200 ms | +100% | Lecture ~10 fichiers locaux |
| Sauvegarde blueprint | ~500 ms | < 2 s | +300% | 50 props, sérialisation + écriture |
| Chargement blueprint (50 props) | ~1 s | < 3 s | +200% | Validation + spawn batch 5/tick |
| Chargement blueprint (150 props) | ~3 s | < 10 s | +230% | Max config, spawn batch 5/tick |
| Matérialisation 1 ghost | ~100 ms | < 200 ms | +100% | Spawn prop + remove ghost |
| Chargement caisse véhicule | ~200 ms | < 500 ms | +150% | SetParent + physics |
| Déchargement caisse | ~300 ms | < 500 ms | +66% | SetParent(nil) + teleport |
| Requête MySQL (INSERT log) | ~5 ms | < 50 ms | +900% | Prepared statement asynchrone |
| Requête MySQL (SELECT logs) | < 1 ms | < 50 ms | +5000% | Index composites |

4.1.3. Qualité de service réseau

| Source Engine UDP | 20-100 Kbps | 0.4-2 Mbps | < 5% (protocole résilient) |
|--------------------|------------------|-------------|----------------------------|
| Net messages addon | < 5 Kbps | < 100 Kbps | 0% (messages critiques) |
| Workshop download | ~8 Go (one-time) | N/A | N/A |
| MySQL (interne) | Négligeable | Négligeable | 0% (réseau Docker local) |

Mécanismes de QoS implémentés :

| | | |
|----------------------------|---|--|
| | | |
| Rate limiting | 60 req/min par joueur | Protège contre le flooding, garantit la réactivité |
| Batch spawning | 5 ghosts par tick au lieu de tous d'un coup | Évite les lag spikes lors du chargement |
| Cooldowns par action | Save 10s, Load 15s, Vehicle 1s | Lisse la charge serveur |
| Net message size limit | Validation < 64 Ko | Évite les messages surdimensionnés |
| Requêtes MySQL asynchrones | Callback-based, jamais bloquant | Le tick rate n'est jamais impacté par la DB |

4.2. EFFICIENCE DES RESSOURCES – C3.2

4.2.1. Dimensionnement et ratios d'utilisation

| | | | | | |
|-----------------|----------|------------------|------------------|-----|-----|
| | | | | | |
| RAM GMod | 3 Go | 1.8 Go | 2.5 Go | 60% | 83% |
| CPU GMod | 2 cores | 0.8 core | 1.5 cores | 40% | 75% |
| RAM MySQL | 512 Mo | 150 Mo | 250 Mo | 29% | 49% |
| CPU MySQL | 0.5 core | 0.05 core | 0.2 core | 10% | 40% |
| RAM VPS total | 16 Go | ~4 Go (services) | ~6 Go (services) | 25% | 37% |
| Stockage Docker | ~80 Go | ~25 Go | ~30 Go | 31% | 37% |

4.2.2. Analyse du right-sizing – C3.2

| | | | |
|------------------|----------|-------------------|--|
| | | | |
| RAM GMod | 3 Go | Adaptée | Pic à 2.5 Go avec 101 add-ons Workshop. Marge de 20% pour les pics. |
| CPU GMod | 2 cores | Adaptée | Pic à 1.5 cores en activité intense. Source Engine est mono-thread pour le Lua, mais le réseau et la physique utilisent des threads séparés. |
| RAM MySQL | 512 Mo | ⚠ Surdimensionnée | Utilisation réelle ~150 Mo. Pourrait être réduite à 256 Mo sans impact. Conservée à 512 Mo pour marge future (partage blueprints). |
| CPU MySQL | 0.5 core | Adaptée | Utilisation très faible (~5%). Le minimum Docker Compose. |

Taux d'efficience globale : ~35% en moyenne, ~55% en pic

Note : Un ratio moyen de 35% est intentionnel pour un environnement de développement/démonstration. Il offre une marge confortable pour les pics et les tests de charge. En production, un ratio de 60-70% serait visé via du right-sizing plus agressif.

4.2.3. Optimisations d'efficience implémentées

| Batch spawning (5/tick) | Spawn tous les ghosts en 1 tick → lag spike | Étalé sur N/5 ticks | Tick rate stable |
|-------------------------|---|---------------------------------|---|
| Requêtes MySQL async | Bloquant (hypothétique) | Non-bloquant callbacks | 0 impact tick rate |
| Rate limiting | Pas de limite → flood possible | 60/min par joueur | CPU protégé |
| Halos client-only | Envoi de l'état de sélection par net | Halos calculés côté client | -80% trafic réseau sélection |
| ClientsideModel preview | Entités serveur pour la prévisualisation | Modèles client uniquement | 0 entité serveur pour le preview |
| NWBool pour état caisse | Polling ou net messages | NetworkVar natif | Synchronisation automatique, 0 overhead |
| Think loop lazy | Check chaque tick (66/s) | Check avec intervalle (CurTime) | -90% appels Think |

4.2.4. Scalabilité

| Joueurs simultanés | 2 (dev) | ~30-40 | RAM GMod (3 Go) + tick rate |
|-----------------------|--------------|----------|------------------------------|
| Props par blueprint | 150 (config) | ~300 | Limite entités Source (2048) |
| Blueprints par joueur | Illimité | Illimité | Espace disque client |
| Ghosts simultanés | ~150 | ~500 | Limite entités Source (2048) |
| Logs MySQL par jour | ~200 | ~100 000 | Espace disque + purge |
| Addons Workshop | 101 | ~200 | RAM GMod |

4.3. CONTRAINTES DE STOCKAGE

| Workshop Collection | ~8 Go | Fixe | ~100 Mo/mois (mises à jour) |
|---------------------------|---------------------------|--------------------------|-----------------------------|
| Image Docker commitée | ~10 Go | 1 par version stable | ~10 Go par snapshot |
| Blueprints (client) | 1-50 Ko par fichier | Illimité par joueur | Variable |
| Logs MySQL | ~100 octets par entrée | ~10 000 entrées/ mois | Linéaire |
| MySQL data files | ~50 Mo | Fixe | Faible |
| Backups (rétention 7j) | ~8 Mo/jour | ~60 Mo | Stable (rotation) |

5. VUE SÉCURITÉ – C2.1, C2.2, C4.2

5.1. POLITIQUE DE SÉCURITÉ

5.1.1. Principes directeurs

| Moindre privilège | Chaque acteur n'a accès qu'aux ressources nécessaires à son rôle |
|------------------------------|--|
| Défense en profondeur | Multiples couches de protection (réseau, applicatif, données) |
| Sécurité by design | Mesures intégrées dès la conception, pas ajoutées après |
| Zero trust client | Le serveur ne fait jamais confiance aux données client |
| Fail-safe | En cas d'erreur, le système refuse l'action (deny by default) |

5.1.2. Matrice RBAC (Role-Based Access Control) – C2.1

Couche applicative (addon GMod)

| Constructeur (TEAM_BUILDER) | (ses props) | | | | | |
|---------------------------------------|----------------|--|--|------------------|--|--|
| Joueur standard | | | | (avec caisse) | | |
| SuperAdmin | (tous) | | | | | |

Couche infrastructure (VPS / Docker)

| | | | | | | |
|-----------------|-----------|--|---|---------|------------------|--|
| | | | | | | |
| Admin système | (clé RSA) | | | | | |
| Conteneur GMod | | | — | | (SELECT, INSERT) | |
| Conteneur MySQL | | | | (local) | | |
| Joueur externe | | | | | | |

Couche base de données (MySQL) – C4.2

| | | | | |
|--------------------------|---------------|--------------------------------|---------------------|-------------------------------------|
| | | | | |
| root | localhost | ALL PRIVILEGES | Toutes | Administration, maintenance, backup |
| gmod_user | gmod-server.% | SELECT, INSERT, UPDATE, DELETE | gmod_construction.* | Opérations CRUD addon uniquement |
| backup_user (recommandé) | localhost | SELECT, LOCK TABLES, SHOW VIEW | gmod_construction.* | Dumps mysqldump uniquement |

```
-- Création de l'utilisateur applicatif (principe de moindre privilège)
CREATE USER 'gmod_user'@'%' IDENTIFIED BY '***';
GRANT SELECT, INSERT, UPDATE, DELETE ON gmod_construction.* TO 'gmod_user'@'%';
-- Pas de DROP, ALTER, CREATE → l'addon ne peut pas modifier le schéma
-- Pas de GRANT → l'addon ne peut pas escalader les privilèges

-- Utilisateur backup (lecture seule)
CREATE USER 'backup_user'@'localhost' IDENTIFIED BY '***';
GRANT SELECT, LOCK TABLES, SHOW VIEW ON gmod_construction.* TO 'backup_user'@'localhost';
```

5.1.3. Politiques d'accès réseau – C2.1, C2.2

```
Firewall UFW – Politique par défaut : DENY ALL
```

```
ENTRÉE (incoming):
```

```
Port 22/tcp ← SSH (clé RSA uniquement)
```

```
Port 27015/udp ← Source Engine (joueurs)
```

```
Port 27015/tcp ← RCON (admin)
```

```
Tout le reste ← DENY
```

```
INTERNE (Docker bridge):
```

```
gmod-server → gmod-mysql:3306 ← ALLOW
```

```
Tout le reste ← DENY (isolation conteneurs)
```

```
MySQL 3306 :
```

```
Exposé uniquement sur le réseau Docker
```

```
NON accessible depuis Internet
```

Mesures d'accès SSH :

```
# /etc/ssh/sshd_config
PasswordAuthentication no      # Pas de brute force possible
PubkeyAuthentication yes      # Clé RSA obligatoire
PermitRootLogin prohibit-password # Root par clé uniquement
MaxAuthTries 3                 # Limite les tentatives
```

5.2. ANALYSE DES MENACES

5.2.1. Surface d'attaque

| Net message flooding | Saturation serveur | Élevée | Élevé | Rate limiting 60/min + cooldowns |
|-----------------------------|--------------------------|-------------|----------|--|
| Injection données blueprint | Spawn entités interdites | Moyenne | Élevé | Whitelist classes + blacklist + validation |
| SQL injection | Accès/modification DB | Faible | Critique | Prepared statements systématiques |
| Usurpation CPPI | Sélection props d'autrui | Faible | Moyen | Vérification <code>CPPIGetOwner()</code> serveur |
| RCON brute force | Accès admin serveur | Moyenne | Critique | Mot de passe fort + firewall |
| Exploitation Docker | Escape conteneur | Très faible | Critique | Mode non-privilégié, limites ressources |
| SSH brute force | Accès VPS | Faible | Critique | Clé RSA only, fail2ban |
| Man-in-the-middle MySQL | Interception requêtes | Très faible | Moyen | Réseau Docker interne (pas d'exposition) |

5.2.2. Données sensibles

| RCON password | server.cfg (conteneur) | Confidentiel | Non exposé, accès SSH uniquement |
|-------------------|----------------------------------|--------------|---|
| Credentials MySQL | Variables d'environnement Docker | Confidentiel | docker-compose.yml, accès root VPS uniquement |
| SteamID joueurs | Mémoire serveur + logs DB | Interne | Information semi-publique Steam |
| Blueprints | Client local | Public | Données non sensibles |

5.3. MESURES DE SÉCURITÉ IMPLÉMENTÉES

5.3.1. Rate Limiting (3 couches)

Couche 1 : Rate limit global

- 60 requêtes/minute par joueur (tous net messages confondus)
- Compteur réinitialisé chaque minute
- Dépassement → message ignoré + log serveur + notification joueur

Couche 2 : Cooldowns par action

- Sauvegarde : 10 secondes
- Chargement : 15 secondes
- Véhicule : 1 seconde
- Dépassement → notification au joueur + refus silencieux

Couche 3 : Nettoyage mémoire

- PlayerDisconnected → suppression compteur et cooldowns
- Pas de fuite mémoire, même avec des milliers de connexions/déconnexions

5.3.2. Validation des entrées (defense in depth)

| Nom blueprint | <code>string.sub(name, 1, 50)</code> , caractères autorisés | Troncature silencieuse |
|--------------------|---|------------------------|
| Description | <code>string.sub(desc, 1, 200)</code> | Troncature silencieuse |
| Rayon sélection | <code>math.Clamp(radius, 50, 1000)</code> | Borné aux limites |
| Nombre props | <code>#props <= MaxPropsPerBlueprint</code> | Rejet avec message |
| Classe entité | <code>AllowedClasses[class] == true</code> | Prop ignoré |
| Classe blacklistée | <code>BlacklistedEntities</code> pattern match | Rejet avec log |
| Modèle 3D | <code>util.IsValidModel(model)</code> | Prop ignoré |
| Entity ref | <code>IsValid(ent)</code> systématique | Opération annulée |
| Ownership | <code>ent:CPPIGetOwner() == ply</code> | Refus sélection |

5.3.3. Sécurité base de données – C4.2

Prévention injection SQL :

```

-- TOUTES les requêtes utilisent des prepared statements MySQLOO
-- JAMAIS de concaténation de chaînes dans le SQL

-- Correct (prepared statement)
local q = db:prepare("INSERT INTO blueprint_logs (steamid, player_name, action, blueprint_id, blueprint_name, details) VALUES (?, ?, ?, ?, ?, ?)")
q:setString(1, ply:SteamID64())
q:setString(2, ply:Nick())
q:setString(3, action)
q:setString(4, blueprintID or "")
q:setString(5, blueprintName or "")
q:setString(6, details or "")
q:start()

-- Interdit (concaténation)
-- db:query("INSERT INTO logs VALUES ('" .. ply:SteamID() .. "')")

```

Politique d'accès aux données :

| Moindre privilège MySQL | <code>gmod_user</code> : SELECT/INSERT/UPDATE/DELETE uniquement | C4.2 |
|---------------------------------------|--|------|
| Pas de DROP/ALTER | L'addon ne peut pas modifier le schéma | C4.2 |
| Réseau interne Docker | MySQL non exposé sur Internet | C4.2 |
| Prepared statements | Toutes requêtes précompilées | C4.2 |
| Callbacks d'erreur | <code>onError</code> avec logging détaillé | C4.2 |
| Connection pooling | MySQLOO gère les connexions automatiquement | C4.3 |
| Chiffrement au repos (recommandation) | <code>innodb_file_per_table</code> , chiffrement tablespace possible | C4.2 |

5.3.4. Contrôle d'accès applicatif – C2.1

| Ownership CPPI | <code>ent:CPPIGetOwner() == ply</code> | Par prop individuel |
|-------------------------|---|---------------------------|
| Restriction job SWEP | <code>SWEPJobs</code> config + hook <code>PlayerLoadout</code> | Par job DarkRP |
| Restriction job caisses | <code>CrateAllowedJobs</code> + hook <code>CanPlayerUse</code> | Par job DarkRP |
| F4 entity spawn | <code>allowed = {TEAM_X}</code> dans DarkRP <code>entities.lua</code> | Par job DarkRP |
| FPP hooks | <code>PhysgunPickup</code> , <code>CanTool</code> , <code>GravGunPickupAllowed</code> | Par entité + propriétaire |
| Commandes admin | <code>ply:IsSuperAdmin()</code> | Par rang |
| Caisses chargées | <code>IsLoaded</code> NWBool → physgun/gravgun bloqué | Par état de la caisse |

5.3.5. Intégrité client/serveur (zero trust)

Le principe fondamental : **le client n'a jamais raison.**

| | |
|----------------------|---|
| | |
| Sélectionner un prop | <code>IsValid(ent)</code> , <code>GetClass() == "prop_physics"</code> , <code>CPPIGetOwner == ply</code> , rate limit |
| Sauvegarder | Rate limit, cooldown, props valides, ownership, nombre max, blacklist |
| Charger un blueprint | Rate limit, cooldown, <code>ValidateBlueprintData</code> (classes, limites, cohérence) |
| Confirmer placement | Rate limit, données preview existent, position valide |
| Matérialiser | ActiveCrate <code>IsValid</code> , matériaux > 0, ghost <code>IsValid</code> , distance |
| Véhicule reload | Rate limit, cooldown, trace valide, véhicule simfphys, caisse à proximité |

5.3.6. Sécurité Docker

| | | |
|-------------------------------|---|---|
| | | |
| Limites de ressources | 3 Go / 2 CPU (GMod), 512 Mo / 0.5 CPU (MySQL) | Pas de déni de service par consommation |
| Mode non-privilégié | Pas de <code>--privileged</code> | Container escape plus difficile |
| Réseau isolé | MySQL uniquement via Docker bridge | Pas d'accès externe à la DB |
| Healthcheck | MySQL ping 30s / 3 retries | Détection automatique des pannes |
| Bind mounts ciblés | Uniquement les répertoires nécessaires | Surface d'attaque minimale |
| Pas de <code>--cap-add</code> | Capabilities par défaut | Moindre privilège kernel |

5.4. TRAÇABILITÉ ET AUDIT

5.4.1. Logs serveur (console)

Chaque action significative est logguée en console :

```
[Construction] Player "John" (STEAM_0:0:12345) saved blueprint "maison" (45 props)
[Construction] Player "Jane" (STEAM_0:1:67890) materialized ghost #123
[Construction] RATE LIMIT: Player "Spammer" (STEAM_0:0:99999) - 61 req/min
[Construction] BLOCKED: Player "Hacker" attempted blacklisted entity "money_printer"
```

5.4.2. Logs base de données (optionnel)

Table `blueprint_logs` avec indexation complète (cf. §2.3) : - Chaque save, load, delete, share est enregistré - SteamID, nom joueur, action, détails, IP, timestamp - Requêtes d'audit rapides grâce aux index composites

5.4.3. Commandes d'audit admin

| <code>construction_logs [n]</code> | SuperAdmin | Affiche les n derniers logs (défaut: 20, max: 100) |
|------------------------------------|------------|---|
| <code>construction_stats</code> | SuperAdmin | Statistiques globales (blueprints, builders, props, logs, partages) |

5.4.4. Matrice de traçabilité complète

| Sauvegarde blueprint | | | Non | Console: session / DB: 90 jours |
|------------------------------|----------|--|-------------|---------------------------------|
| Chargement blueprint | | | Non | Idem |
| Matérialisation | | | Non | Idem |
| Suppression blueprint | | | Non | Idem |
| Dépassement rate limit | | | Log warning | Idem |
| Tentative classe blacklistée | | | Log warning | Idem |
| Erreur MySQL | | | Log error | Console: session |
| Connexion/déconnexion joueur | (DarkRP) | | Non | Console: session |

GLOSSAIRE

| | |
|----------------------------|--|
| | |
| Blueprint | Sauvegarde sérialisée d'un ensemble de props (positions, modèles, angles). Stocké en JSON dans un fichier <code>.dat</code> sur le PC du joueur. |
| Ghost / Fantôme | Entité <code>construction_ghost</code> – prop holographique bleu translucide, non-solide. Représente un prop à construire. |
| Matérialisation | Action de transformer un ghost en vrai <code>prop_physics</code> solide, en consommant 1 matériau d'une caisse. |
| Caisse de matériaux | Entité <code>construction_crate</code> ou <code>construction_crate_small</code> – conteneur de matériaux achetable au F4. |
| SWEP | Scripted Weapon – arme Lua custom dans GMod. Ici : <code>weapon_construction</code> . |
| CPPI | Common Prop Protection Interface – API standard pour vérifier la propriété des entités. |
| DarkRP | Gamemode roleplay pour Garry's Mod. Système de jobs, économie, entités F4. |
| simfphys | Framework de véhicules physiques réalistes pour GMod. |
| Net message | Message réseau envoyé entre client et serveur via la net library de GMod (UDP). |
| GLua | Garry's Mod Lua – dialecte Lua 5.1 avec extensions Source Engine. |
| Bind mount | Volume Docker monté depuis un chemin du host vers le container. |
| MySQLOO | Module binaire (.dll/.so) pour GMod permettant des requêtes MySQL asynchrones. |
| Rate limiting | Mécanisme limitant le nombre de requêtes par unité de temps. |
| Prepared statement | Requête SQL précompilée avec paramètres. Prévient l'injection SQL. |
| FPP | Falco's Prop Protection – système de protection des props intégré à DarkRP. |
| PCA | Plan de Continuité d'Activité – procédures de reprise après incident. |
| RBAC | Role-Based Access Control – contrôle d'accès basé sur les rôles. |
| RPO | Recovery Point Objective – perte de données maximale acceptable. |
| RTO | Recovery Time Objective – temps de remise en service maximal. |
| MTBF | Mean Time Between Failures – temps moyen entre deux pannes. |
| MTTR | Mean Time To Repair – temps moyen de réparation. |
| SLA | Service Level Agreement – accord sur le niveau de service. |
| QoS | Quality of Service – qualité de service réseau et applicative. |