

TorUsNet 9.0 FR

L'Architech (Gaberiel Breton-Breault)

1 Introduction générale

1.1 — Avant-propos : évolution naturelle vers TorUsNet 9.0

Le protocole initialement nommé *TorusNet* évolue dans cette version **9.0** vers une identité pleinement assumée : **TorUsNet**. Ce changement reflète une transition majeure dans la philosophie du projet.

Dans ses premières versions, TorusNet était conçu comme un protocole directionnel indépendant, démontrant la faisabilité d'une architecture de communication basée sur un tore privé inviolable et un tore public opaque. Avec **TorUsNet 9.0**, le protocole franchit une étape décisive : il repose désormais sur une **architecture directionnelle volumétrique 100×100×100**, capable de mutation interne, de réécriture dynamique et d'évolution tridimensionnelle.

Le nom TorUsNet exprime cette triple nature :

- **Tor** : continuité avec l'écosystème du navigateur Tor et ses objectifs de confidentialité.
- **Us** : référence directe au *torus*, cœur de la géométrie directionnelle.
- **Net** : couche réseau non linéaire, volumétrique et structurellement inviolable.

TorUsNet n'est pas un fork de Tor. C'est une **architecture directionnelle vivante**, conçue pour remplacer la couche de transport tout en conservant l'esprit du navigateur Tor : anonymat, confidentialité, résilience et liberté d'accès.

La version **9.0** marque le début d'une convergence : TorUsNet devient un protocole pensé pour Tor, mais également pour les architectures photonique-compatibles, le projet VSMD et le vaisseau SMD.

1.2 — Introduction des clés directionnelles 9.0 (transmises sur le réseau)

TorUsNet 9.0 introduit un nouveau type de clé directionnelle, optimisé pour fonctionner avec un tore **100×100×100**. Ces clés sont transmises avant le parcours directionnel et servent à initialiser correctement le tore privé volumétrique.

Structure des clés directionnelles 9.0

Chaque clé possède la structure suivante :

- **20 bits — Point de départ** Indique la case initiale dans le tore privé volumétrique où commence l'interprétation du parcours.
- **16 bits — ID de la case** Identifie la case visée dans le tore privé pour cette clé.

Résumé

Clé directionnelle = 20 bits (départ) + 16 bits (ID) = 36 bits

Compactes, rapides à transmettre, elles permettent d'initialiser le tore privé sans exposer sa géométrie interne tridimensionnelle.

Rôle dans le protocole

Ces clés :

- préparent le tore privé à recevoir le parcours complet,
- définissent la zone directionnelle initiale dans le volume,
- assurent la cohérence entre tore privé et parcours transmis,
- permettent une reconstruction plus précise,
- garantissent une interprétation correcte des déplacements volumétriques.

Elles sont transmises avant le parcours, car elles servent de pré-configuration directionnelle.

1.3 — Transmission directe des parcours directionnels (modèle 9.0)

Dans les versions précédentes, le protocole utilisait un seed pour générer le parcours directionnel. TorUsNet 9.0 introduit une alternative étendue : la transmission directe du **parcours directionnel volumétrique complet**.

Désormais :

- le parcours directionnel peut être envoyé directement sur le réseau,
- le tore privé volumétrique l'utilise tel quel pour reconstruire le fichier,
- aucune information interne du tore n'est exposée,
- la sécurité repose entièrement sur la géométrie directionnelle 3D.

Le parcours transmis ne contient :

- aucune clé interne,
- aucune structure du tore privé,
- aucune géométrie directionnelle,
- aucune information exploitable.

Il s'agit uniquement d'un vecteur de déplacement volumétrique, sans valeur cryptographique.

1.4 — Transmission du hash du fichier reconstruit

Pour garantir que le fichier reconstruit par le tore privé volumétrique est :

- complet,
- intact,
- non corrompu,
- cohérent avec le parcours transmis,

TorUsNet 9.0 introduit la transmission du **hash du fichier reconstruit**.

Ce hash :

- est envoyé sur le réseau,
- ne révèle aucune information directionnelle,
- ne permet aucune dérivation du tore,
- ne compromet aucune clé,
- sert uniquement à valider l'intégrité du fichier final.

Il devient une preuve externe que la reconstruction directionnelle s'est déroulée correctement, même en cas de :

- perte de clés,
- parcours partiellement reçu,
- reconstructions multiples,
- interruptions réseau.

1.5 — Coexistence des deux modèles : seed ou parcours transmis

TorUsNet 9.0 permet toujours de remplacer complètement le seed par un parcours intégral transmis directement sur le réseau. Cependant, cette transmission directe peut alourdir le trafic réseau.

Pour cette raison, TorUsNet 9.0 conserve les deux modèles :

Modèle A — Seed directionnel (compact, local)

- Le seed de 70 bits est transmis.
- Le tore privé volumétrique génère le parcours localement.
- Le trafic réseau est minimal.
- Idéal pour les environnements à faible bande passante.

Modèle B — Parcours directionnel transmis (complet, externe)

- Le parcours complet est envoyé sur le réseau.
- Le seed n'est plus nécessaire.
- Le protocole devient plus simple à implémenter.
- Idéal pour l'intégration dans un navigateur Tor ou une architecture photonique.

Choix du modèle

Le choix entre seed ou parcours transmis doit être fait au moment de la programmation, selon :

- les contraintes du système,
- la bande passante,
- la simplicité recherchée,
- le type d'application,
- les préférences du développeur.

Les deux modèles sont officiellement supportés et documentés dans **TorUsNet 9.0**.

Chapitre 1 — Introduction à l'Architecture Directionnelle (TorUsNet 9.0)

1.1. Vision générale

L'architecture directionnelle est une technologie de communication entièrement nouvelle. Elle ne repose pas sur des mathématiques, des dérivations cryptographiques ou des structures classiques, mais sur une **mécanique interne vivante**, impossible à dériver, impossible à explorer, impossible à falsifier.

TorUsNet 9.0 s'appuie désormais sur **quatre éléments fondamentaux**, révisés pour la nouvelle architecture volumétrique :

1. **Le Tore Privé Volumétrique** — une structure interne de **100×100×100 = 1 000 000 cases**, chacune contenant :
 - 1 bit normal,
 - 1 bit fantôme implicite (NOT du bit normal),
 - 4 directions possibles. Le tore privé est unique à chaque session, jamais transmis, jamais exporté, jamais dérivable.
2. **La Mécanique Directionnelle Volumétrique** — un ensemble de règles définissant :
 - comment une clé se déplace dans le tore 3D,
 - comment les IDs mutent dans les trois axes,
 - comment les chemins internes s'activent,
 - comment les paires normal/fantôme s'inversent,
 - comment la cohérence interne est maintenue dans un volume.
3. **La Forteresse de Verre Volumétrique** — un module interne qui :
 - valide la cohérence directionnelle 3D,
 - surveille les ports directionnels étendus,
 - surveille les IDs internes,
 - détruit la session en cas d'anomalie,
 - bloque les IP hostiles. C'est un pare-feu structurel directionnel.
4. **Les Clés Directionnelles Transmises (TorUsNet 9.0)** — un type de clé envoyé avant le parcours :

- 20 bits : point de départ,
- 16 bits : ID de la case, soit **36 bits par clé**. Ces clés préparent le tore privé volumétrique à recevoir le parcours transmis.

L'architecture directionnelle n'utilise :

- ni chiffrement,
- ni dérivation mathématique,
- ni signatures,
- ni certificats,
- ni clés privées,
- ni primitives cryptographiques.

Elle repose exclusivement sur :

- des fenêtres directionnelles,
- des constantes directionnelles,
- des parcours internes volumétriques,
- des mutations d'ID,
- des chemins internes,
- des paires normal/fantôme,
- des structures internes non dérivables.

1.2. Pourquoi une architecture directionnelle ?

Les protocoles classiques (TLS, SSH, IPSec, WireGuard, etc.) reposent sur :

- des mathématiques,
- des dérivations,
- des clés privées,
- des certificats,
- des signatures.

Ces systèmes sont vulnérables à :

- l'analyse mathématique,
- les ordinateurs quantiques,
- les ordinateurs photoniques,
- les attaques par dérivation,
- les collisions,
- les attaques par répétition,
- les injections.

L'architecture directionnelle élimine entièrement ces vecteurs d'attaque. Elle ne repose sur **aucune structure mathématique exploitable**. Elle repose sur une mécanique interne **impossible à dériver, impossible à explorer, impossible à inverser**, même avec une puissance infinie.

Un attaquant ne peut pas :

- dériver un tore privé volumétrique,
- explorer un tore privé,
- reconstruire un tore privé,
- prédire un parcours directionnel,
- falsifier une clé directionnelle,
- répéter une clé directionnelle,
- reconstruire un ID interne.

1.3. Les trois éléments fondamentaux (version 9.0)

1.3.1. Le Tore Privé Volumétrique

Le Tore Privé est une structure interne de **1 000 000 cases**, chacune contenant :

- 1 bit normal,
- 1 bit fantôme implicite,
- 4 directions possibles.

Il est :

- unique à chaque session,

- reconstruit fractalement,
- jamais transmis,
- jamais exporté,
- jamais dérivable.

Il contient :

- les IDs internes,
- l'ID global,
- l'ID unique du tore,
- les chemins internes,
- les paires normal/fantôme.

Inversion des bits normal/fantôme lors du parcours

Dans TorUsNet 9.0, chaque fois qu'une case est traversée par le parcours directionnel volumétrique, les bits normal et fantôme s'inversent automatiquement :

- normal = 0 $\rightarrow$ 1
- fantôme = 1 $\rightarrow$ 0 ou inversement.

Cette inversion :

- crée une oscillation directionnelle,
- modifie la géométrie interne du tore 3D,
- empêche toute prédiction,
- renforce la non dérivabilité,
- rend chaque parcours unique et non reproductible.

Cette règle reste **fondamentale** dans TorUsNet 9.0.

Mutation directionnelle volumétrique

Nouvelle règle 9.0 :

Si l'intensité traverse N couches, alors les N couches mutent.

Dans un tore 100×100×100 :

- une intensité traverse 100 couches,
- déclenche 100 mutations,
- reboucle,
- retraverse,
- remute.

La mutation se propage dans les trois axes : **X, Y, Z**.

1.3.2. La Mécanique Directionnelle Volumétrique

Elle définit :

- comment une clé se déplace dans le tore 3D,
- comment les IDs mutent,
- comment les chemins internes s'activent,
- comment les paires normal/fantôme s'inversent,
- comment la cohérence interne est maintenue.

Elle est :

- non linéaire,
- non dérivable,
- non réversible,
- vivante,
- oscillante,
- mutante.

1.3.3. La Forteresse de Verre Volumétrique

La Forteresse de Verre :

- valide la cohérence directionnelle 3D,
- surveille les ports directionnels,
- surveille les IDs internes,
- détruit la session en cas d'anomalie,

- bloque les IP hostiles.

Elle est l'équivalent directionnel d'un **pare-feu structurel volumétrique**.

1.4. Ports directionnels : ports numériques étendus

Les ports directionnels sont :

- des ports numériques classiques,
- mais avec une plage étendue dans les milliards,
- générés par les clés directionnelles,
- validés par la Forteresse de Verre.

Ils servent à :

- transporter les transmissions directionnelles,
- éviter les collisions,
- empêcher les scans,
- empêcher les injections,
- empêcher les détournements de flux.

Exemples :

- 1 482 993 112
- 7 000 221 991
- 3 991 000 000

Impossible à scanner. Impossible à prédire. Impossible à falsifier.

1.5. Une architecture post-Internet

L'architecture directionnelle est conçue pour l'ère :

- post quantique,
- post cryptographique,
- photonique.

Elle ne dépend pas :

- des mathématiques,

- des certificats,
- des clés privées,
- des primitives cryptographiques.

Elle utilise encore :

- DNS,
- HTTPS,

mais uniquement comme **sas d'entrée temporaire**, avant que la session ne devienne **100 % directionnelle**.

1.6. Une session directionnelle : vivante et autonome

Une session directionnelle est :

- vivante,
- autonome,
- auto cohérente,
- auto validante,
- auto mutante.

Elle évolue à chaque clé :

- mutation d'ID,
- inversion normal/fantôme,
- progression interne,
- mise à jour du tore privé volumétrique,
- validation par la Forteresse de Verre.

Elle ne peut pas être :

- copiée,
- clonée,
- dérivée,
- prédite,

- falsifiée.

1.7. Objectifs du protocole directionnel

TorUsNet vise à :

1. Créer des sessions inviolables par conception.
2. Remplacer les protocoles cryptographiques classiques.
3. Créer des réseaux privés directionnels (multi-tores volumétriques).
4. Permettre la reconstruction directionnelle de fichiers.
5. Créer une couche de communication interne (ports directionnels).
6. Créer une identité directionnelle unique.
7. Assurer une sécurité post quantique et photonique.
8. Étendre l'architecture aux systèmes photoniques, au VSMD et au vaisseau SMD.

1.8. Résumé du Chapitre 1

L'architecture directionnelle :

- n'est pas cryptographique,
- n'est pas mathématique,
- n'est pas dérivable,
- n'est pas falsifiable.

Elle repose sur :

- un tore privé volumétrique $100 \times 100 \times 100$,
- une mécanique interne vivante,
- une cohérence directionnelle 3D,
- des ports numériques étendus,
- une Forteresse de Verre volumétrique,
- des clés directionnelles transmises,
- un parcours transmis ou généré par seed,
- un hash final validant la reconstruction,

- l'inversion automatique des bits normal/fantôme,
- la mutation directionnelle volumétrique par traversée.

TorUsNet 9.0 constitue une nouvelle génération de protocole, conçue pour l'ère post quantique, photonique et directionnelle.

Chapitre 2 — Refonte du Seed Directionnel et Introduction du Modèle à Parcours Transmis (TorUsNet 9.0)

TorUsNet 9.0 introduit une évolution majeure dans la manière dont les trajectoires directionnelles sont générées, transmises et interprétées. Deux modèles coexistent désormais :

- **Le modèle classique basé sur le seed directionnel (70 bits)**
- **Le modèle moderne basé sur le parcours transmis directement sur le réseau (modèle 9.0)**

Les deux modèles sont officiellement supportés et documentés.

2.1 — Structure de la clé directionnelle (modèle seed, 128 bits)

La clé directionnelle complète conserve sa structure originale de **128 bits**, composée de :

- **2 bits — Type de clé**
- **20 bits — Point de départ directionnel**
- **20 bits — Point d'arrivée directionnel**
- **16 bits — ID de case visée**
- **70 bits — Seed directionnel**

Total : 128 bits

Cette clé reste compatible avec le tore volumétrique $100 \times 100 \times 100$, car elle définit uniquement les points directionnels, pas la géométrie interne.

2.2 — Structure du seed directionnel (70 bits)

Le seed directionnel est un bloc compact de **70 bits** contenant cinq composantes :

- **Base exponentielle (4 bits)**
- **Sélection de la tour fixe (4 bits)**

- **Hauteur de la tour fixe (2 bits)**
- **Étages dynamiques (48 bits)**
- **Modulation finale (12 bits)**
 - Addition : 4 bits
 - Multiplication : 8 bits

Ce moteur exponentiel reste inchangé dans TorUsNet 9.0, car il est indépendant de la géométrie volumétrique.

2.2.1 — Base exponentielle (4 bits)

Les 4 bits définissent une base parmi 16 valeurs hiérarchiques :

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 100, 1000, 10 000, 100 000, 1 000 000, 10 000 000

2.2.2 — Sélection de la tour fixe (4 bits)

Liste complète des 15 tours fixes disponibles, chacune suivant la règle :

$$n^{(n+1)(n+2)(n+3)}$$

où $n \in [1, 15]$.

Les 4 bits permettent :

- **0000 → aucune tour fixe**
- **0001 à 1111 → sélection d'une tour fixe parmi les 15 possibilités**

2.2.3 — Hauteur de la tour fixe (2 bits)

Détermine le nombre d'étages utilisés :

- 00 → 1 étage
- 01 → 2 étages
- 10 → 3 étages
- 11 → 4 étages

2.2.4 — Étages dynamiques (48 bits)

Les 48 bits sont divisés en 6 étages dynamiques, chacun représenté par une valeur brute sur 8 bits :

$$D_1, D_2, D_3, D_4, D_5, D_6 \in [0, 255]$$

Règle directionnelle :

- si un étage vaut 0 → ignoré
- si > 0 → actif dans la tour exponentielle

2.2.5 — Modulation finale (12 bits)

La modulation finale est composée de :

- **Addition (A)** — 4 bits → 0 à 15
- **Multiplication (M)** — 8 bits → 0 à 255

La valeur exponentielle finale devient :

$$E_{\text{final}} = (X + A) \times M$$

où X représente la tour complète.

2.3 — Structure exponentielle finale

Avec tour fixe :

$$\text{Base}^{e_1^{e_2^{e_3^{e_4^{D_1^{D_2^{D_3^{D_4^{D_5^{D_6}}}}}}}}}}$$

Sans tour fixe :

$$\text{Base}^{D_1^{D_2^{D_3^{D_4^{D_5^{D_6}}}}}}$$

Puis modulation :

$$E_{\text{final}} = (X + A) \times M$$

2.4 — Capacité de représentation

- **Espace brut :**

$$2^{48}$$

- **Espace exponentiel :** dépasse largement les capacités cryptographiques, quantiques et photoniques.

2.5 — Impact sur le tore privé volumétrique

Le tore privé volumétrique 100×100×100 utilise :

- la base exponentielle,
- la tour fixe (si activée),
- les 6 étages dynamiques,
- la modulation finale (A et M),
- la règle d'ignorance des étages nuls,

pour déterminer la trajectoire directionnelle interne **dans les trois axes X/Y/Z.**

Inversion normal/fantôme

Lors du parcours :

- chaque case traversée inverse son bit normal et son bit fantôme,
- modifiant la géométrie interne du tore volumétrique,
- rendant chaque parcours unique et non reproductible.

Mutation directionnelle volumétrique (nouveau 9.0)

Si l'intensité traverse N couches, les N couches mutent.

Dans un tore 100×100×100 :

- intensité traverse 100 couches,
- déclenche 100 mutations,
- reboucle,
- retraverse,
- remute.

Mutation dans **X, Y, Z.**

2.6 — Introduction du modèle à parcours transmis (TorUsNet 9.0)

TorUsNet 9.0 introduit un second modèle :

Le parcours directionnel volumétrique complet est transmis directement sur le réseau.

Ce modèle :

- supprime l'utilisation du seed,
- simplifie la génération directionnelle,
- permet une intégration naturelle dans un navigateur Tor,
- augmente légèrement le trafic réseau,
- nécessite les nouvelles clés directionnelles (36 bits).

Ordre de transmission :

1. **Clés directionnelles (36 bits)**
2. **Parcours directionnel volumétrique complet**
3. **Hash du fichier reconstruit**

2.7 — Conclusion du Chapitre 2

TorUsNet 9.0 introduit :

- un seed directionnel de 70 bits,
- un moteur exponentiel multi-étages,
- 15 tours fixes,
- une hauteur configurable,
- 6 étages dynamiques sur 48 bits,
- une modulation finale sur 12 bits,
- un modèle basé sur le parcours transmis,
- de nouvelles clés directionnelles (36 bits),
- la transmission du hash final,
- l'inversion automatique des bits normal/fantôme,
- la mutation directionnelle volumétrique par traversée,
- une cohérence directionnelle 3D,
- une architecture vivante et auto-mutante.

Cette structure renforce la nature inviolable, non dérivable et géométriquement imprévisible de TorUsNet 9.0.

Chapitre 3 — Forteresse de Verre (Pare-feu directionnel structurel, TorUsNet 9.0)

La Forteresse de Verre est le **pare-feu directionnel structurel** du tore privé volumétrique. Elle valide, surveille et protège toutes les opérations directionnelles dans un tore **100×100×100**, selon le modèle choisi par les programmeurs :

- **Modèle A : Clé directionnelle transmise (36 bits)**
- **Modèle B : Clé directionnelle complète (128 bits)**

Les deux modèles sont supportés, mais **ne doivent jamais être utilisés simultanément**. La Forteresse de Verre s'adapte automatiquement au modèle actif.

3.1. Vue d'ensemble

La Forteresse de Verre valide :

- la clé directionnelle (36 bits ou 128 bits),
- l'ID transmis (16 bits),
- l'ID reconstruit (16 bits),
- la cohérence fractale du tore volumétrique,
- la reconstruction directionnelle du fichier,
- l'inversion normal/fantôme lors du parcours,
- la mutation directionnelle volumétrique (nouveau 9.0).

Elle protège :

- le tore privé 100×100×100,
- la clé directionnelle du modèle actif,
- la cohérence directionnelle 3D,
- la reconstruction du fichier,
- la stabilité fractale interne.

Elle détruit la session en cas d'anomalie.

3.2. Rôle fondamental

✓ Validation ✓ Surveillance ✓ Protection ✓ Destruction

La Forteresse de Verre est un **pare-feu structurel directionnel**, conçu pour l'ère post-quantique et photonique.

3.3. Validation conjointe : Forteresse de Verre + Host

Peu importe le modèle choisi (36 bits ou 128 bits), la Forteresse de Verre valide toujours la clé directionnelle **avec le Host**.

✓ Fonctionnement

1. Le Host reçoit la clé directionnelle (36 ou 128 bits).
2. Le Host effectue une pré-validation directionnelle.
3. La Forteresse de Verre effectue la validation structurelle finale.

✓ Si les deux valident la clé :

La Forteresse de Verre autorise :

- la mutation des 16 bits internes,
- la reconstruction interne,
- l'exécution du parcours volumétrique,
- l'inversion normal/fantôme,
- la mutation directionnelle par traversée (100 couches mutées si 100 traversées).

✗ Si l'un des deux rejette la clé :

La Forteresse de Verre :

- bloque toute mutation,
- bloque toute reconstruction,
- détruit la session,
- efface le tore privé,
- bloque l'IP.

✓ Règle directionnelle absolue

La clé directionnelle doit être validée par le Host ET par la Forteresse de Verre.

3.4. Modèle A — Validation de la clé directionnelle transmise (36 bits)

Ce modèle est utilisé lorsque les programmeurs choisissent de transmettre la clé directionnelle sur le réseau.

La Forteresse de Verre vérifie :

1. Les 20 bits du point de départ

- case existante dans le volume,
- cohérence directionnelle 3D,
- cohérence fractale.

2. Les 16 bits de l'ID transmis

- ID existant,
- ID non corrompu,
- ID non falsifié.

3. Les 16 bits de l'ID reconstruit

- cohérence interne,
- mutation valide,
- compatibilité avec le modèle 36 bits.

4. Les 36 bits de la clé directionnelle transmise

- cohérence fractale,
- absence de dérive,
- absence de falsification.

Si un seul bit est incohérent → session détruite.

3.5. Modèle B — Validation de la clé directionnelle complète (128 bits)

Ce modèle est utilisé lorsque les programmeurs choisissent de ne pas transmettre la clé, mais de la reconstruire entièrement dans le tore privé volumétrique.

La Forteresse de Verre vérifie :

1. Les 20 bits du point de départ

2. Les 16 bits de l'ID transmis
3. Les 16 bits de l'ID reconstruit
4. Les **128 bits** de la clé directionnelle complète

La clé directionnelle de 128 bits est :

- la seule clé qui mute,
- la seule clé qui inverse,
- la seule clé qui oscille,
- la seule clé qui se déplace dans le volume.

Si la clé directionnelle est incohérente → session détruite.

3.6. Règle structurelle : les IDs de case peuvent se répéter

✓ Un ID de case peut se répéter sans aucune limite. ✓ Un ID peut être partagé par : → 10 cases, → 100 cases, → 1 000 cases, → 50 000 cases, → ou davantage.

✓ Cela ne compromet pas la sécurité, car elle repose sur :

- la géométrie directionnelle volumétrique,
- les mutations internes,
- les déplacements directionnels 3D,
- la cohérence fractale,
- l'inversion normal/fantôme,
- la mutation directionnelle par traversée.

✓ 50 % des 16 bits sont réservés aux mutations directionnelles. ✓ Les autres 50 % sont les IDs actifs, libres et non contraints.

✓ Les **1 000 000 cases** du tore privé peuvent :

- partager massivement les mêmes IDs,
- se regrouper en clusters directionnels 3D,
- muter vers des IDs déjà utilisés,
- se superposer fractalement.

Ce n'est pas une collision : → c'est une propriété directionnelle normale, attendue et validée.

✓ Rôle de la Forteresse de Verre

Elle valide que :

- les répétitions d'ID sont légales,
- les mutations vers des IDs déjà utilisés sont cohérentes,
- la fractale interne reste stable malgré les regroupements,
- aucune dérive directionnelle n'est détectée.

3.7. Validation de la reconstruction du fichier

La Forteresse de Verre vérifie :

- la cohérence de la clé directionnelle (36 ou 128 bits),
- la cohérence du tore volumétrique,
- la cohérence fractale,
- l'absence de falsification,
- l'absence de dérive.

Elle vérifie aussi que :

- la taille du fichier n'est jamais transmise,
- la taille du fichier n'est jamais visible,
- la reconstruction est purement directionnelle 3D.

✗ Si une incohérence est détectée :

- la session est détruite,
- le tore privé est effacé,
- la clé directionnelle est détruite,
- la reconstruction est annulée,
- l'IP est bloquée.

3.8. Protection structurelle

La Forteresse de Verre protège :

✓ Le tore privé volumétrique ✓ La clé directionnelle du modèle actif ✓ Le fichier reconstruit ✓ La cohérence fractale ✓ La cohérence directionnelle 3D

Elle rend toute falsification impossible par conception.

3.9. Destruction automatique

Si une incohérence est détectée :

- la session est détruite,
- le tore privé est effacé,
- la clé directionnelle est détruite,
- la reconstruction est annulée,
- l'IP est bloquée.

La destruction est :

- immédiate,
- irréversible,
- totale.

3.10. Résumé du Chapitre 3

La Forteresse de Verre :

- valide la clé directionnelle (36 ou 128 bits),
- valide l'ID transmis,
- valide l'ID reconstruit,
- valide la clé conjointement avec le Host,
- valide la reconstruction du fichier,
- accepte les IDs répétés sans limite,
- protège le tore privé volumétrique,
- détruit la session en cas d'anomalie,
- surveille la mutation directionnelle volumétrique,

- garantit la cohérence directionnelle 3D.

Elle est le **pare-feu directionnel structurel**, inviolable par conception, au cœur de TorUsNet 9.0.

Chapitre 4 — Tore Privé (Structure interne, géométrie, fichiers, directions, IDs) — TorUsNet 9.0

4.1. Vue d'ensemble

Le Tore Privé est la **structure directionnelle volumétrique interne** utilisée pour :

- parcourir les directions internes 3D,
- exécuter les sauts exponentiels circulaires,
- muter les IDs dans les trois axes,
- inverser normal/fantôme,
- reconstruire le fichier,
- stabiliser la cohérence fractale volumétrique.

Il contient :

- **1 000 000 cases**, organisées en **100 canevas de 100×100**,
- chaque canevas représente **une couche directionnelle**,
- les 100 canevas forment un tore volumétrique **100×100×100**.

Chaque case possède :

- un ID sur 16 bits,
- un bit normal,
- un bit fantôme implicite,
- 4 directions internes,
- un déplacement torique 3D,
- un saut exponentiel circulaire.

Le Tore Privé est programmé avec **deux fichiers distincts**, lus en parallèle.

4.2. Structure interne du tore volumétrique

Chaque case du tore contient :

✓ **1. Un ID sur 16 bits**

- 100 % des valeurs possibles (0 à 65535) sont utilisées.
- Les IDs peuvent se répéter sans aucune limite.
- Aucune contrainte d'unicité, de collision ou de disponibilité.

✓ **2. Un bit normal**

- 0 = normal
- 1 = fantôme (implicite, jamais transmis)

✓ **3. Quatre directions internes**

Codées sur 2 bits :

Code Direction

00 direction 1

01 direction 2

10 direction 3

11 direction 4

✓ **4. Cohérence fractale volumétrique**

Chaque case possède une cohérence interne qui :

- se met à jour à chaque mutation,
- se met à jour à chaque inversion normal/fantôme,
- se met à jour à chaque déplacement directionnel 3D.

4.3. Organisation interne : 100 canevas de 100×100 (100 couches)

Le tore privé est structuré en **100 canevas indépendants**, chacun de **100×100 cases**.

Chaque canevas représente :

- une **couche directionnelle**,
- une **surface interne autonome**,

- un **plan fractal indépendant**,
- un **niveau de mutation volumétrique**.

Les 100 canevas sont lus :

- **en parallèle**,
- **en profondeur**,
- comme **100 couches directionnelles superposées**,
- formant un tore volumétrique **100×100×100**.

Cette structure :

- permet la mutation directionnelle par traversée (nouveau 9.0),
- permet l'intensité cyclique mod 100,
- permet la propagation des mutations dans les trois axes,
- rend la géométrie interne impossible à dériver.

4.4. Déplacements cardinaux, diagonaux et volumétriques (comportement torique 3D)

Les déplacements internes se comportent comme un tore parfait **dans les trois axes** :

- dépasser en haut → arrivée en bas
- dépasser en bas → arrivée en haut
- dépasser à gauche → arrivée à droite
- dépasser à droite → arrivée à gauche
- dépasser en profondeur → retour à la surface
- dépasser la surface → retour en profondeur

Ce comportement :

- supprime les bords,
- supprime les limites,
- supprime les zones mortes,
- garantit une continuité parfaite,
- rend impossible toute sortie du tore volumétrique.

Les déplacements diagonaux et volumétriques suivent la même logique.

4.5. Sauts exponentiels (comportement circulaire 3D)

Chaque direction interne contient un **saut exponentiel circulaire**.

Ce saut :

- ne suit pas les axes cardinaux,
- ne suit pas les diagonales,
- ne suit pas la grille 3D,
- ne suit pas la géométrie du tore volumétrique.

Il suit une **orbite circulaire interne**, indépendante de la géométrie 3D.

✓ Le saut exponentiel se déplace sur un cercle. ✓ Lorsqu'il atteint la fin du cercle, il revient au début. ✓ Il ne peut jamais sortir de l'orbite. ✓ Il ne peut jamais se bloquer. ✓ Il ne peut jamais dériver.

Ce comportement circulaire :

- amplifie la portée directionnelle,
- crée une géométrie interne indépendante,
- rend les parcours impossibles à inverser,
- stabilise la cohérence fractale,
- empêche toute dérivation externe.

4.6. Géométrie hybride : tore 3D + cercle

La structure interne combine deux géométries :

✓ Géométrie torique 3D

Pour les déplacements cardinaux, diagonaux et volumétriques.

✓ Géométrie circulaire

Pour les sauts exponentiels.

Cette dualité :

- crée une surface directionnelle vivante,

- modifie la géométrie selon le type de déplacement,
- rend les parcours impossibles à inverser,
- stabilise la cohérence fractale,
- empêche toute dérivation externe.

4.7. Architecture logicielle du Tore Privé

Le Tore Privé est programmé avec **deux fichiers distincts**, chacun ayant un rôle précis.

4.7.1. Fichier des bits normaux (binaire pur, chargé en RAM)

Ce fichier :

- contient uniquement les bits normaux,
- est un fichier binaire pur,
- est chargé entièrement en RAM,
- ne contient aucune direction,
- ne contient aucun ID,
- ne contient aucun saut exponentiel.

Le chargement en RAM permet :

- une mutation instantanée des cases,
- une lecture ultra rapide des bits normaux,
- une reconstruction continue du fichier,
- une inversion normal/fantôme immédiate.

4.7.2. Fichier directionnel (texte, reste sur le disque dur)

Ce fichier :

- reste sur le disque dur,
- est un fichier texte,
- contient les 4 directions de chaque case,
- contient les sauts exponentiels,
- contient les IDs de chaque case,

- contient **1 000 000 lignes**,
- est parcouru en parallèle avec le fichier binaire.

Structure d'une ligne :

Code

<ID> 101 2^5, 110 3^7, 101 2^7, 111 4^7

4.8. Mutation des IDs (mise à jour TorUsNet 9.0)

Deux types de mutation existent, selon le modèle directionnel choisi.

4.8.1. Mutation locale (modèle 36 bits)

Dans le modèle 36 bits :

- la clé transmise fournit uniquement la case de départ et son ID,
- la clé ne parcourt pas le tore,
- la mutation locale est effectuée par le parcours directionnel interne,
- la clé ne mute jamais directement,
- aucune inversion normal/fantôme n'est effectuée par la clé.

Mutation locale :

$$ID_{\text{case}} \leftarrow ID_{\text{resultat du parcours}}$$

4.8.2. Mutation structurelle (modèle 128 bits)

Dans le modèle 128 bits :

- la clé interne parcourt le tore 3D,
- inverse normal/fantôme,
- reconstruit le fichier,
- mute l'ID global du tore,
- mute l'ID de la case de départ,
- modifie la cohérence fractale volumétrique.

Mutation structurelle :

$$ID_{\text{tore}} \leftarrow ID_{\text{aleatoire}}$$

$$ID_{\text{case départ}} \leftarrow ID_{\text{aleatoire}}$$

4.9. Normal / Fantôme

✓ Le modèle 128 bits peut inverser normal/fantôme. ✗ Le modèle 36 bits ne peut pas inverser normal/fantôme.

4.10. ID du tore

Le tore possède un ID global sur 16 bits.

Cet ID :

- change à chaque clé interne (modèle 128 bits),
- est choisi aléatoirement,
- peut se répéter librement,
- n'a aucune contrainte d'unicité.

4.11. Rôle du tore dans la reconstruction du fichier

Le Tore Privé :

- reconstruit le fichier bit par bit,
- stabilise la cohérence fractale volumétrique,
- applique les mutations directionnelles,
- applique les inversions normal/fantôme,
- applique les sauts exponentiels,
- applique les déplacements toriques 3D.

La taille du fichier :

- n'est jamais transmise,
- n'est jamais visible,
- est déduite directionnellement.

4.12. Résumé du Chapitre 4

Le Tore Privé :

- contient **100 canevas de 100×100**, lus comme **100 couches directionnelles**,

- forme un tore volumétrique **100×100×100**,
- possède 1 000 000 cases,
- chaque case possède un ID sur 16 bits,
- les IDs peuvent se répéter librement et massivement,
- les déplacements cardinaux sont toriques 3D,
- les sauts exponentiels sont circulaires,
- la géométrie hybride rend les parcours impossibles à inverser,
- le tore est programmé avec deux fichiers (binaire RAM + directionnel disque),
- le modèle 36 bits fournit la case de départ et son ID,
- la mutation locale du modèle 36 bits est effectuée par le parcours directionnel interne,
- le modèle 128 bits mute structurellement,
- l'inversion normal/fantôme dépend du modèle choisi,
- la cohérence fractale est mise à jour en continu,
- la mutation directionnelle volumétrique par traversée est appliquée (nouveau 9.0),
- la structure hybride rend le tore **inviolable par conception**.

Chapitre 5 — Système des ID (16 bits, répétition, mutation, rôle)

Version TorUsNet 9.0 — mise à jour complète

5.1. Vue d'ensemble

Le protocole directionnel utilise un système d'identifiants (ID) entièrement basé sur :

- des IDs sur **16 bits**,
- **100 % des valeurs possibles**,
- des IDs répétables **sans aucune limite**,
- des mutations aléatoires,
- des redéfinitions locales ou structurelles,

- une interaction dépendante du modèle directionnel choisi :
 - **modèle 36 bits** (mutation locale via parcours volumétrique),
 - **modèle 128 bits** (mutation structurelle).

Les IDs sont au cœur :

- du Tore Privé volumétrique,
- des 1 000 000 cases,
- des mutations directionnelles 3D,
- de la cohérence fractale,
- de la reconstruction du fichier.

5.2. Taille des IDs : 16 bits

Tous les IDs du protocole sont sur **16 bits** :

- ID de case,
- ID du tore,
- ID transmis par le modèle 36 bits,
- ID reconstruit par le modèle 128 bits,
- ID muté localement (modèle 36 bits),
- ID muté structurellement (modèle 128 bits).

✓ Valeurs possibles

0 → 65535

✓ 100 % des valeurs sont utilisées

Aucune valeur n'est :

- réservée,
- interdite,
- mise de côté.

5.3. Répétition des IDs

Les IDs peuvent se répéter **librement**, sans aucune limite.

Plusieurs cases peuvent partager le même ID, y compris :

- des dizaines,
- des centaines,
- des milliers,
- ou des dizaines de milliers de cases.

✓ Raison structurelle

- L'ID n'est **pas** un identifiant unique.
- L'ID est un **marqueur directionnel**.
- La sécurité repose sur la **géométrie directionnelle 3D**, et non sur l'unicité des IDs.

✓ Conséquence

- 1 000 000 cases,
- 65 536 IDs possibles,
- répétitions normales,
- collisions normales,
- aucun impact sur la cohérence fractale volumétrique.

5.4. Mutation des IDs

Deux types de mutation existent, selon le modèle directionnel choisi.

5.4.1. Mutation locale (modèle 36 bits)

(mise à jour TorUsNet 9.0)

Dans le modèle 36 bits :

- la clé fournit uniquement :
 - la case de départ,
 - l'ID de départ ;
- la clé ne parcourt pas le tore,
- la clé n'exécute aucun déplacement,
- la mutation locale est effectuée par le **parcours directionnel volumétrique**,

- aucune autre case n'est modifiée directement,
- aucune inversion normal/fantôme n'est effectuée par la clé.

Mutation locale :

$$ID_{\text{case}} \leftarrow ID_{\text{resultat du parcours}}$$

La mutation est donc une **conséquence du déplacement directionnel 3D**, et non une action directe de la clé.

5.4.2. Mutation structurelle (modèle 128 bits)

Dans le modèle 128 bits :

- la clé interne parcourt le tore volumétrique,
- inverse normal/fantôme,
- reconstruit le fichier,
- mute l'ID global du tore,
- mute l'ID de la case de départ,
- modifie la cohérence fractale 3D.

Mutation structurelle :

$$ID_{\text{tore}} \leftarrow ID_{\text{aleatoire}}$$

$$ID_{\text{case depart}} \leftarrow ID_{\text{aleatoire}}$$

Les IDs aléatoires sont choisis sur 16 bits, même s'ils sont déjà utilisés.

5.5. Mutation aléatoire sur 16 bits

Lors d'une mutation, le programme choisit un ID **au hasard** sur 16 bits, même si cet ID est déjà utilisé.

✓ Propriétés

- mutation = tirage aléatoire sur 0–65535,
- aucune contrainte d'unicité,
- aucune contrainte de collision,
- aucune contrainte de disponibilité,

- mutation **purement directionnelle**.

5.6. Rôle des IDs dans le Tore Privé volumétrique

Chaque case du tore possède :

- un ID sur 16 bits,
- un bit normal,
- un bit fantôme implicite,
- quatre directions internes,
- un déplacement torique 3D,
- un saut exponentiel circulaire.

L'ID :

- identifie la case dans le fichier directionnel,
- sert de point d'entrée pour les mutations,
- sert de marqueur pour la cohérence fractale,
- sert de repère pour la reconstruction interne.

5.7. ID du tore

Le tore possède un ID global sur 16 bits.

Cet ID :

- change à chaque clé interne (modèle 128 bits),
- est choisi aléatoirement,
- peut se répéter,
- n'a aucune contrainte d'unicité.

Le modèle 36 bits ne modifie **jamais** l'ID global du tore.

5.8. Interaction entre IDs et modèles directionnels

✓ Modèle 36 bits

- fournit la case de départ,
- fournit l'ID de départ,

- déclenche un parcours interne 3D,
- la mutation locale est effectuée par le parcours,
- ne modifie jamais l'ID du tore,
- n'inverse jamais normal/fantôme.

✓ **Modèle 128 bits**

- parcourt le tore volumétrique,
- inverse normal/fantôme,
- reconstruit le fichier,
- mute l'ID du tore,
- mute l'ID de la case de départ,
- stabilise la cohérence fractale 3D.

5.9. Cohérence fractale et IDs

La cohérence fractale :

- accepte les IDs répétés,
- accepte les collisions,
- accepte les mutations aléatoires,
- se stabilise automatiquement,
- se met à jour à chaque mutation directionnelle 3D.

Les IDs ne sont pas une structure de sécurité. Ils sont une **structure directionnelle**.

5.10. Résumé du Chapitre 5

Le système des IDs fonctionne ainsi :

- tous les IDs sont sur 16 bits,
- 100 % des valeurs sont utilisées,
- les IDs peuvent se répéter librement,
- les mutations choisissent un ID aléatoire,
- le modèle 36 bits fournit la case de départ et mute via le parcours interne 3D,

- le modèle 128 bits mute le tore et la case de départ,
- les collisions d'ID sont normales,
- la cohérence fractale reste stable,
- la sécurité repose sur la géométrie directionnelle volumétrique, et non sur l'unicité des IDs.

Chapitre 6 — Handshake Directionnel (Séquence Host ↔ Client)

Version TorUsNet 9.0 — mise à jour complète

6.1. Vue d'ensemble

Le handshake directionnel est le mécanisme structurel qui établit une session cohérente entre un Host et un Client.

Contrairement aux protocoles classiques (TLS, SSH, IPSec) reposant sur :

- clés privées,
- certificats,
- signatures,
- dérivations mathématiques,
- échanges cryptographiques,

le handshake directionnel repose exclusivement sur :

- la géométrie interne des tores **100×100×100**,
- la cohérence fractale volumétrique,
- les déplacements directionnels 3D,
- les IDs internes,
- la validation conjointe Host + Forteresse de Verre.

Il ne contient **aucune cryptographie**, aucune dérivation, aucune structure mathématique exploitable.

6.2. Objectifs du handshake

Le handshake directionnel assure :

1. la validation de la première clé directionnelle (36 bits ou 128 bits),
2. la synchronisation initiale entre Host et Client,
3. l'inscription de l'IP du Client comme IP de confiance,
4. la création de l'ID global de session (16 bits),
5. l'activation de la Forteresse de Verre volumétrique,
6. l'initialisation du Tore Privé (100 canevas de 100×100),
7. la mise en place de l'ID global du tore (16 bits),
8. la synchronisation directionnelle complète via **5 transmissions**.

Ces étapes sont structurelles, non cryptographiques.

6.3. La séquence officielle : 5 transmissions

La séquence directionnelle comporte **5 transmissions**, chacune ayant un rôle structurel précis. Elle est obligatoire, inviolable, et constitue la base de la synchronisation directionnelle.

Transmission 1 — Host → Client

Envoi initial de la clé directionnelle

Le Host envoie une clé directionnelle :

- **Modèle 36 bits :**

- 20 bits : case de départ,
- 16 bits : ID de départ.

- **Modèle 128 bits :**

- 128 bits : clé directionnelle complète.

Le Client :

- reçoit la clé,
- vérifie sa structure,
- vérifie son modèle (36 ou 128 bits),
- vérifie l'ID visé,
- vérifie la cohérence fractale initiale,

- transmet la clé à la Forteresse de Verre pour validation conjointe.

Aucune exécution directionnelle n'est encore effectuée.

Transmission 2 — Client → Host

Renvoi de la clé pour confirmer la réception

Le Client renvoie **exactement la même clé** au Host.

Le Host :

- compare la clé envoyée et la clé reçue,
- valide la réception,
- inscrit l'IP du Client comme IP de confiance,
- active la Forteresse de Verre volumétrique.

Si la clé diffère → **destruction immédiate**.

Transmission 3 — Host → Client

Autorisation d'exécution

Le Host renvoie une seconde copie de la clé.

Le Client :

- compare la clé renvoyée avec celle reçue,
- valide la cohérence,
- exécute le parcours directionnel interne dans son Tore Privé (100 couches),
- met à jour les chemins internes,
- met à jour la cohérence fractale volumétrique.

Effets selon le modèle :

✓ Modèle 36 bits

- mutation locale de l'ID de la case de départ,
- mutation effectuée par le parcours directionnel 3D,
- aucune inversion normal/fantôme.

✓ Modèle 128 bits

- inversion normal/fantôme,
- mutation structurelle de l'ID du tore,
- mutation structurelle de l'ID de la case de départ,
- mutation directionnelle volumétrique (100 couches mutées si 100 traversées).

Le Tore Privé du Client devient :

- vivant,
- oscillant,
- modifié.

Transmission 4 — Client → Host

Confirmation d'exécution du parcours

Le Client renvoie la clé **après avoir exécuté le parcours interne**.

Ce renvoi indique au Host :

- que la mutation locale (36 bits) ou structurelle (128 bits) a été effectuée,
- que les chemins internes ont été mis à jour,
- que la progression fractale est cohérente,
- que le Tore Privé du Client est à jour.

Le Host peut alors :

- exécuter le même parcours,
- appliquer les mêmes mutations,
- synchroniser son Tore Privé avec celui du Client.

Transmission 5 — Host → Client

Confirmation finale de synchronisation

Le Host renvoie la clé une dernière fois pour confirmer :

- qu'il a exécuté le parcours,
- qu'il a appliqué les mutations,
- que son Tore Privé est synchronisé avec celui du Client,

- que la session est directionnellement stable.

Le Client reçoit cette confirmation et la session devient :

- cohérente,
- stable,
- inviolable.

6.4. Pourquoi cinq transmissions ?

6.4.1. Validation bidirectionnelle complète

Host valide Client. Client valide Host. Puis chacun valide l'exécution directionnelle de l'autre.

6.4.2. Protection contre les injections

Un attaquant ne peut pas :

- intercepter la clé,
- la modifier,
- la renvoyer,
- la falsifier.

Toute incohérence est détectée structurellement.

6.4.3. Protection contre les répétitions

Une clé répétée :

- est détectée,
- est rejetée,
- détruit la session.

6.4.4. Protection contre les dérivations

Un attaquant ne peut pas dériver :

- la case de départ,
- l'ID de départ,
- les déplacements internes 3D,

- les mutations,
- la cohérence fractale.

La structure directionnelle est **non dérivable**.

6.4.5. Synchronisation directionnelle parfaite

Les 5 transmissions garantissent que :

- les deux tores privés ont exécuté le même parcours,
- les mutations sont identiques,
- la géométrie interne est cohérente,
- aucune divergence n'est possible.

6.5. Synchronisation initiale

Une fois la clé validée :

- le Tore d'Authentification est utilisé une seule fois,
- le Tore Privé (100 canevas) est initialisé,
- l'ID global est créé,
- les paires normal/fantôme sont définies (modèle 128 bits),
- les chemins internes sont activés,
- l'ID global du tore est établi,
- la Forteresse de Verre est armée.

Cette synchronisation est **fractale, non mathématique**.

6.6. Reconstruction initiale du Tore Privé

La première clé déclenche :

- l'initialisation directionnelle,
- la mise en place des chemins internes,
- la mutation locale (36 bits) ou structurelle (128 bits),
- la stabilisation fractale volumétrique,
- l'activation des directions internes,

- l'activation des sauts exponentiels.

Le Tore Privé devient :

- vivant,
- oscillant,
- opérationnel.

6.7. Gestion des erreurs pendant le handshake

6.7.1. Clé incohérente

Si la clé :

- contient un ID invalide,
- contient une case impossible,
- contient un modèle illégitime,

→ la session est détruite.

6.7.2. Clé modifiée

Si la clé renvoyée diffère :

- destruction immédiate,
- effacement du Tore Privé,
- invalidation de l'IP.

6.7.3. Absence de réponse

Si le Host ne reçoit aucune réponse après 5 secondes :

- la clé est considérée perdue,
- une nouvelle clé est générée,
- avec une case de départ différente,
- mais reconstruisant la même structure directionnelle.

Aucune retransmission du même paquet n'est effectuée.

6.8. Résumé du Chapitre 6

Le handshake directionnel :

- repose sur **5 transmissions**,
- valide la première clé directionnelle,
- inscrit l'IP du Client comme IP de confiance,
- initialise le Tore Privé (100 canevas),
- active la Forteresse de Verre volumétrique,
- crée l'ID global,
- établit l'ID unique du tore,
- garantit une cohérence directionnelle totale,
- protège contre les injections, répétitions, dérivations et falsifications.

Il constitue la base de la **sécurité structurelle directionnelle**.

Chapitre 7 — Multi Tore et Réseaux Parallèles

Version TorUsNet 9.0 — mise à jour complète

7.1. Vue d'ensemble

Le **multi tore** est l'un des fondements les plus puissants de l'architecture directionnelle TorUsNet 9.0. Il permet de créer plusieurs réseaux directionnels **parallèles**, chacun possédant :

- son propre Tore Privé **100×100×100**,
- son propre Tore d'Authentification,
- ses propres clés directionnelles (36 bits ou 128 bits),
- ses propres IDs internes (16 bits),
- son propre ID global de session,
- son propre ID global du tore,
- sa propre Forteresse de Verre volumétrique.

Le **Tore Public**, pré-généré et universel, constitue la base directionnelle commune à toutes les branches.

Chaque réseau parallèle est totalement isolé des autres, même s'ils coexistent :

- sur la même machine,
- sur le même système,
- dans le même environnement réseau.

Le multi tore permet :

- la segmentation directionnelle,
- la souveraineté structurelle,
- l'isolation totale des branches,
- la création de réseaux privés inviolables,
- la duplication directionnelle contrôlée.

7.2. Définition d'un réseau parallèle

Un réseau parallèle est une instance complète de l'architecture directionnelle comprenant :

1. Le Tore Public (pré-généré, universel)
2. Un Tore d'Authentification
3. Un Tore Privé **100×100×100**
4. Un ensemble de clés directionnelles
5. Une Forteresse de Verre volumétrique
6. Un ID global de session
7. Un ID global du tore

Chaque réseau parallèle est un **univers directionnel indépendant**.

Il ne partage :

- aucune clé,
- aucun ID,
- aucun tore privé,
- aucune cohérence fractale,
- aucun chemin interne.

7.3. Création d'une nouvelle branche directionnelle

La création d'une nouvelle branche suit **trois étapes**.

7.3.1. Utilisation du Tore Public pré-généré

Le Tore Public :

- est pré-généré,
- est fourni avec le logiciel,
- est universel,
- est identique pour toutes les installations,
- est non modifiable,
- est non dérivable.

Il constitue la base directionnelle commune, mais ne contient **aucune information privée**.

7.3.2. Génération du Tore d'Authentification

Un Tore d'Authentification est généré. Il valide la première clé directionnelle et initialise :

- les premières paires normal/fantôme (modèle 128 bits),
- les premiers chemins internes,
- l'ID global de session,
- l'ID global du tore.

Ce tore n'est utilisé **qu'une seule fois** : lors du handshake initial.

7.3.3. Génération du Tore Privé volumétrique

Un Tore Privé **100×100×100** est généré. Il sera reconstruit fractalement à partir des clés directionnelles :

- inversion normal/fantôme (modèle 128 bits),
- mutation d'ID (locale ou structurelle),
- activation des chemins internes,
- stabilisation fractale volumétrique,
- mutation directionnelle par traversée (100 couches mutées si 100 traversées).

Chaque Tore Privé est **indépendant** des autres tores privés existants.

7.4. Branches directionnelles

Chaque réseau parallèle est appelé **branche directionnelle**.

Une branche possède :

- son propre Tore Privé 100×100×100,
- sa propre progression fractale,
- ses propres IDs internes,
- sa propre Forteresse de Verre,
- son propre ID global,
- son propre ID global du tore.

7.4.1. Branches indépendantes

Les branches ne communiquent pas entre elles. Elles ne peuvent pas :

- partager des clés,
- partager des IDs,
- partager des tores privés,
- partager des chemins internes,
- partager des mutations.

7.4.2. Branches souveraines

Chaque branche est **souveraine** :

- elle ne dépend d'aucune autre branche,
- elle ne peut pas être influencée par une autre branche,
- elle ne peut pas être dérivée par une autre branche.

7.5. Duplication directionnelle (Tore primaire hors ligne)

Une fonctionnalité essentielle du multi tore est la **duplication directionnelle hors ligne**.

7.5.1. Tore primaire

Un tore primaire peut être :

- généré,
- stocké hors ligne,
- transféré physiquement,
- utilisé pour créer une nouvelle branche.

7.5.2. Transfert hors ligne

Le tore primaire peut être transféré :

- sur clé USB,
- sur disque dur,
- sur support physique,
- sur tout média non réseau.

Ce transfert ne contient **aucune clé directionnelle**. Il contient uniquement :

- la géométrie directionnelle 100×100×100,
- les IDs internes,
- les directions internes,
- les sauts exponentiels,
- les 100 canevas de 100×100.

7.5.3. Création d'une branche à partir du tore primaire

Une fois le tore primaire transféré :

- une nouvelle branche est créée,
- un nouveau Tore Privé volumétrique est reconstruit,
- un nouvel ID global est généré,
- un nouvel ID global du tore est établi,
- une nouvelle Forteresse de Verre est activée.

La branche est **totale**ment indépendante de la branche d'origine.

7.6. Isolation structurelle

L'isolation structurelle est **totale**.

7.6.1. Isolation fractale

Les actions fractales d'une branche :

- ne peuvent pas influencer une autre branche,
- ne peuvent pas dériver une autre branche,
- ne peuvent pas reconstruire une autre branche.

7.6.2. Isolation directionnelle

Les chemins internes d'une branche :

- ne peuvent pas être utilisés dans une autre branche,
- ne peuvent pas être dérivés par une autre branche.

7.6.3. Isolation des IDs

Les IDs internes :

- ne sont pas compatibles entre branches,
- ne peuvent pas être transférés,
- ne peuvent pas être dérivés.

7.7. Multi Tore et sécurité post-quantique

Le multi tore renforce la sécurité post-quantique :

- un ordinateur quantique ne peut pas dériver un Tore Privé,
- ne peut pas dériver une branche,
- ne peut pas dériver une clé directionnelle,
- ne peut pas dériver un ID,
- ne peut pas dériver l'ID global du tore.

Même avec une puissance infinie, un attaquant ne peut pas :

- explorer un Tore Privé,
- dériver un Tore Privé,
- reconstruire un Tore Privé,
- analyser un Tore Privé.

Les tores privés sont **géométriquement inviolables**.

7.8. Multi Tore et sécurité photonique

Avec l'arrivée des ordinateurs photoniques :

- les structures mathématiques deviennent obsolètes,
- les cryptographies classiques deviennent vulnérables.

Le multi tore est conçu pour cette ère :

- les tores privés sont non transmissibles,
- non dérivables,
- non analysables,
- non explorables.

Un ordinateur photonique ne peut pas :

- pénétrer un Tore Privé,
- analyser un Tore Privé,
- dériver un Tore Privé.

7.9. Résumé du Chapitre 7

Le multi tore permet :

- la création de réseaux parallèles,
- l'isolation totale des branches,
- la duplication directionnelle hors ligne,
- la souveraineté structurelle,
- la sécurité post-quantique,
- la sécurité photonique.

Chaque branche est un **univers directionnel indépendant**, inviolable par conception. Le Tore Public, pré-généré et universel, sert de base directionnelle commune, mais **jamais de tore privé partagé**.

Chapitre 8 — Ports Directionnels et Géométrie Interne de Communication

Version TorUsNet 9.0 — mise à jour complète

8.1. Vue d'ensemble

Les ports directionnels de TorUsNet 9.0 constituent une **extension structurelle** des ports classiques. Contrairement aux ports TCP/UDP (0–65535), TorUsNet utilise :

- des ports étendus (plusieurs milliards de possibilités),
- des ports mutables,
- non scannables,
- non dérivables,
- non falsifiables,
- validés structurellement par la Forteresse de Verre volumétrique.

Un port directionnel n'est **pas** :

- un entier,
- un socket,
- un canal TCP,
- un identifiant réseau.

Un port directionnel est une **structure interne du Tore Privé 100×100×100**, générée directionnellement, mutée à chaque renouvellement du tore et à chaque nouvelle connexion.

8.2. Définition d'un port directionnel

Un port directionnel est :

- un numéro de port étendu (plusieurs milliards de possibilités),
- généré par le Tore Privé volumétrique,
- muté à chaque renouvellement du tore,
- muté à chaque nouvelle connexion,
- impossible à prédire,
- impossible à scanner,

- impossible à dériver.

Règle structurelle : une seule connexion par port

Chaque port directionnel :

- n'autorise **qu'une seule connexion**,
- est invalidé dès que la session se termine,
- est remplacé par un nouveau port étendu lors de la prochaine connexion.

8.3. Pourquoi les ports directionnels sont inviolables

8.3.1. Absence dans la couche réseau classique

Un attaquant ne peut pas :

- scanner les ports,
- sonder les ports,
- ouvrir un port,
- envoyer un paquet vers un port.

Les ports directionnels **n'existent pas** dans la couche réseau classique. Ils n'existent que dans la **mécanique interne du Tore Privé 100×100×100**.

8.3.2. Ports mutables et non dérivables

Les ports directionnels :

- mutent automatiquement,
- ne sont jamais réutilisés,
- ne peuvent pas être prédits,
- ne peuvent pas être dérivés mathématiquement.

8.3.3. Validation par la Forteresse de Verre

La Forteresse de Verre volumétrique :

- surveille les ports directionnels,
- valide leur cohérence fractale,
- détruit la session si un port est violé,

- bloque l'IP hostile dans le pare-feu du système d'exploitation.

8.3.4. Ports non transmissibles

Un port directionnel :

- ne peut pas être envoyé,
- ne peut pas être copié,
- ne peut pas être exporté,
- ne peut pas être dérivé.

Il n'existe **que localement**, dans le Tore Privé.

8.4. Mutation des ports directionnels

8.4.1. Mutation lors du renouvellement du tore

Chaque fois que le Tore Privé est régénéré :

- un nouveau set de ports étendus est généré,
- tous les ports précédents deviennent invalides.

8.4.2. Mutation à chaque nouvelle connexion

À chaque nouvelle session :

- un nouveau port étendu est généré,
- une seule connexion est autorisée sur ce port,
- toute tentative de seconde connexion entraîne une destruction immédiate.

8.5. Ports directionnels et transmission

Host → Client

Le Host ouvre un port directionnel étendu. Le Client doit répondre sur ce même port.

Client → Host

Le Client renvoie la clé directionnelle sur ce port. Le Host valide la cohérence.

Host → Client

Le Host renvoie la clé finale sur le même port.

Violation d'un port

Si un port directionnel est violé :

- la session est détruite,
- le Tore Privé est effacé,
- l'IP est bloquée.

8.6. Ports directionnels et multi tore

Dans un environnement multi tore :

- chaque branche possède ses propres ports directionnels,
- les ports ne sont pas compatibles entre branches,
- les ports ne peuvent pas être dérivés d'une branche à l'autre.

Un port directionnel est **strictement local** à sa branche directionnelle.

8.7. Ports directionnels et sécurité post-quantique

Les ports directionnels sont inviolables par un ordinateur quantique :

- impossible de dériver un port étendu,
- impossible de prédire une mutation,
- impossible d'explorer le Tore Privé $100 \times 100 \times 100$,
- impossible de reconstruire les ports internes.

8.8. Ports directionnels et sécurité photonique

Les ordinateurs photoniques peuvent casser :

- RSA,
- ECC,
- TLS,
- SSH.

Mais ils ne peuvent pas :

- analyser une géométrie directionnelle volumétrique,
- dériver un tore,
- reconstruire un port directionnel,

- prédire un port étendu muté.

Les ports directionnels sont **structurellement inviolables**.

8.9. Résumé du Chapitre 8

Les ports directionnels :

- sont des ports classiques étendus à plusieurs milliards de possibilités,
- mutent à chaque renouvellement du tore,
- mutent à chaque nouvelle connexion,
- n'autorisent qu'une seule connexion par port,
- sont non scannables,
- non dérivables,
- non falsifiables,
- validés par la Forteresse de Verre,
- inviolables par conception.

Ils constituent la **couche de communication interne de TorUsNet 9.0**, totalement indépendante des protocoles réseau classiques.

Chapitre 9 — Authentification DNS et HTTPS (Couche de Confiance Temporaire)

Version TorUsNet 9.0 — mise à jour complète

9.1. Vue d'ensemble

Avant que la géométrie directionnelle **100×100×100** ne prenne le contrôle total de la session, une **couche de confiance temporaire** est nécessaire pour :

- identifier l'hôte demandé,
- éviter les redirections malveillantes,
- empêcher les attaques MITM classiques,
- sécuriser la transmission de la première clé directionnelle,
- garantir que le Client contacte bien le bon Host.

Cette couche repose sur deux technologies de l'Internet classique :

- **DNS** (résolution du nom de domaine),
- **HTTPS** (certificat TLS).

Ces technologies **ne font pas partie de la mécanique directionnelle**. Elles servent uniquement de **sas d'entrée**, avant l'activation :

- du Tore d'Authentification,
- du Tore Privé (100 canevas de 100×100),
- de la Forteresse de Verre volumétrique,
- des IDs internes,
- de l'ID global,
- de l'ID global du tore.

Une fois la première clé validée, **DNS et HTTPS disparaissent complètement**.

9.2. Rôle du DNS dans TorUsNet 9.0

DNS est utilisé pour **une seule raison** :

Identifier l'hôte demandé avant d'entrer dans la géométrie directionnelle.

9.2.1. Vérification du domaine

Lorsqu'un Client contacte un Host :

1. DNS résout le nom de domaine.
2. Le protocole vérifie que ce domaine appartient à la liste des hôtes de confiance.
3. Si le domaine est inconnu ou suspect :
 - la connexion passe en mode confirmation utilisateur,
 - aucune clé directionnelle n'est transmise.

9.2.2. Protection contre les redirections

DNS protège contre :

- les redirections malveillantes,
- les faux domaines,
- les hôtes clonés,

- les détournements de trafic.

9.2.3. Rôle strictement temporaire

Une fois la première clé validée :

DNS n'est plus utilisé. La géométrie directionnelle prend le relais.

9.3. Rôle du HTTPS dans TorUsNet 9.0

HTTPS est utilisé pour :

Sécuriser la transmission de la première clé directionnelle.

9.3.1. Vérification du certificat

Le Client vérifie :

- le certificat TLS,
- la validité du domaine,
- l'absence d'interception MITM classique.

Si le site n'a pas de HTTPS :

- la connexion passe en mode confirmation utilisateur,
- aucune clé directionnelle n'est envoyée.

9.3.2. Protection contre les attaques classiques

HTTPS protège contre :

- les interceptions,
- les injections,
- les manipulations de trafic,
- les attaques de downgrade.

9.3.3. Rôle strictement temporaire

Une fois la première clé transmise :

HTTPS disparaît. La session devient 100 % directionnelle.

9.4. Interaction DNS ↔ HTTPS ↔ Tore d'Authentification

La séquence complète est :

1. DNS identifie l'hôte.
2. HTTPS sécurise la première transmission.
3. Le Tore d'Authentification valide la première clé.
4. La Forteresse de Verre active la session directionnelle.
5. Le Tore Privé (100×100×100) prend le contrôle.
6. DNS et HTTPS sont abandonnés.

9.4.1. Cas de confiance totale

- DNS valide
- HTTPS valide

→ La première clé est transmise automatiquement.

9.4.2. Cas de confiance partielle

- DNS valide
- HTTPS absent

→ Confirmation utilisateur obligatoire.

9.4.3. Cas de suspicion

- DNS suspect
- HTTPS absent ou invalide

→ Connexion bloquée jusqu'à validation explicite.

9.5. Pourquoi TorUsNet utilise encore DNS et HTTPS

TorUsNet est une architecture directionnelle **post-Internet**. Cependant, l'écosystème actuel impose :

- l'usage de noms de domaine,
- l'usage de certificats TLS,
- l'usage de serveurs HTTP(S).

DNS et HTTPS servent donc de **pont** entre :

- l'Internet classique,

- l'Internet directionnel.

9.5.1. Objectif final : disparition totale

Dans les versions futures :

- DNS sera remplacé par un **DNS directionnel**,
- HTTPS sera remplacé par une **validation directionnelle native**,
- les certificats seront remplacés par des **IDs directionnels**,
- les connexions seront **entièrement géométriques**.

9.6. Interaction avec les mises à jour directionnelles (TorUsNet 9.0)

Inversion et mutation d'ID

Une fois la première clé validée :

- l'ID de la case de départ mute (modèle 36 bits ou 128 bits),
- l'ID global est créé,
- l'ID global du tore est établi,
- DNS et HTTPS ne sont plus nécessaires.

Session directionnelle autonome

La session devient :

- continue,
- vivante,
- autonome,
- fractale,
- volumétrique (100 couches),
- non transmissible,
- non dérivable.

9.7. Résumé du Chapitre 9

Le chapitre établit que :

- DNS et HTTPS ne servent **qu'à sécuriser la première clé**,

- ils disparaissent immédiatement après validation,
- la session devient **100 % directionnelle**,
- la Forteresse de Verre prend le contrôle,
- le Tore Privé (100×100×100) devient l'unique couche de communication,
- les IDs internes remplacent les certificats,
- les ports directionnels remplacent les sockets,
- la géométrie directionnelle remplace la cryptographie.

Chapitre 10 — Logique de Lecture d'une Case, des Couches et de l'Intensité

Version TorUsNet 9.0 — mise à jour complète

10.1. Vue d'ensemble

La lecture d'une case dans TorUsNet 9.0 repose sur trois mécanismes directionnels fondamentaux :

1. **La lecture circulaire déclenchée par un saut exponentiel suivi d'un opérateur (+, -, ×, ÷)**
2. **La lecture des couches (100 canevas de 100×100) comme un tore vertical**
3. **La logique d'intensité (mod 100) qui détermine la couche lue et les couches mutées**

Ces trois mécanismes fusionnent pour créer une lecture :

- volumétrique,
- circulaire,
- non linéaire,
- non inversable,
- impossible à dériver,
- vivante et fractale.

10.2. Logique de lecture des couches (100 canevas = 100 couches)

Le Tore Privé de TorUsNet 9.0 est composé de :

100 canevas de 100×100 cases, soit **100 couches verticales**, chacune contenant 10 000 cases.

10.2.1. Lecture verticale directionnelle

Lorsque le parcours directionnel demande la lecture d'une couche :

- la couche est lue **ligne par ligne**,
- de la case 1 à la case 10 000,
- comme un cercle (voir section 10.3).

10.2.2. Dépassement vertical

Si la lecture dépasse la couche 100 :

- elle revient à la couche 1.

Si elle descend sous la couche 1 :

- elle revient à la couche 100.

10.2.3. Lecture volumétrique

La lecture peut traverser :

- une seule couche,
- plusieurs couches,
- toutes les couches,
- selon l'intensité et le saut exponentiel.

10.3. Logique de lecture circulaire (saut exponentiel + opérateur)

Lorsque le parcours directionnel exécute :

un saut exponentiel suivi d'un opérateur (+, −, ×, ÷)

alors la lecture bascule en mode **circulaire** :

10.3.1. Lecture circulaire d'un canevas

- Les 10 000 cases sont lues comme un cercle.
- La case 10 000 renvoie à la case 1.
- Aucun bord n'existe.

- La lecture ne peut jamais s'arrêter sur une limite.

10.3.2. Lecture circulaire des couches

- La couche 100 renvoie à la couche 1.
- La couche 1 renvoie à la couche 100.
- Les 100 couches deviennent un anneau vertical.

10.3.3. Effet des opérateurs

Opérateur Effet directionnel

+	avance circulaire accélérée
-	recul circulaire inversé
×	amplification de la vitesse de lecture
÷	réduction fractale de la vitesse de lecture

10.4. Logique de l'intensité (mod 100)

L'intensité est un nombre directionnel qui détermine :

1. **la couche lue,**
2. **le nombre de couches traversées,**
3. **le nombre de couches mutées,**
4. **la profondeur directionnelle du parcours.**

10.4.1. Calcul de la couche lue

$$\text{couche lue} = \text{intensité} \bmod 100$$

Exemples :

- Intensité 254 → $254 \bmod 100 = \mathbf{54}$ → lecture de la couche 54
- Intensité 999 → $999 \bmod 100 = \mathbf{99}$
- Intensité 10 000 → $10\,000 \bmod 100 = \mathbf{0}$ → couche 100

10.4.2. Traversée des couches

Si l'intensité dépasse 100 :

- elle boucle,
- elle revient à la couche 1,
- elle continue son chemin.

10.4.3. Mutation des couches traversées

Règle directionnelle TorUsNet 9.0 :

Si l'intensité traverse N couches, alors les N couches mutent.

Exemple :

- Intensité 254
- $254 \bmod 100 = 54$
- Traversée : $100 \rightarrow 1 \rightarrow 2 \rightarrow \dots \rightarrow 54$
- Nombre de couches traversées : **54**
- Les **54 couches mutent**.

10.5. Inversion automatique des bits lors de la lecture d'une couche

TorUsNet 9.0 introduit une règle directionnelle fondamentale :

Quand une couche est lue, tous les bits traversés s'inversent automatiquement.

Cette inversion s'applique :

- au **bit normal**,
- au **bit fantôme implicite**,
- à **chaque case traversée**,
- dans **chaque couche parcourue**,
- selon la logique d'intensité (mod 100).

10.5.1. Inversion dans la couche lue

Lorsque l'intensité détermine une couche (ex. couche 54) :

- la lecture circulaire parcourt les 10 000 cases,
- **chaque case traversée inverse son bit normal/fantôme**,
- l'inversion est immédiate,

- l'inversion est irréversible,
- l'inversion modifie la cohérence fractale de la couche.

10.5.2. Inversion dans les couches traversées

Si l'intensité traverse N couches :

- les N couches sont lues,
- les N couches mutent,
- **tous les bits traversés dans ces N couches s'inversent**,
- même si la couche n'est pas la couche finale lue.

10.5.3. Interaction avec les modèles directionnels

Modèle 36 bits

- La clé ne parcourt pas le tore.
- Le parcours interne déclenche la lecture circulaire.
- **Les bits traversés s'inversent**, même si le modèle 36 bits n'inverse pas normal/fantôme par lui-même.
- L'inversion est causée par la lecture, pas par la clé.

Modèle 128 bits

- La clé parcourt le tore.
- Le modèle 128 bits inverse déjà normal/fantôme.
- **La lecture ajoute une inversion supplémentaire**, indépendante de la clé.
- Double inversion possible selon le parcours.

10.6. Fusion des trois logiques

Lorsqu'un saut exponentiel suivi d'un opérateur est exécuté **dans une couche déterminée par l'intensité**, la lecture devient :

1. **circulaire dans la couche**,
2. **circulaire dans les 100 couches**,
3. **volumétrique**,
4. **mutante**,

5. **fractale,**

6. **non inversable.**

10.7. Exemple complet

Intensité : 254 Saut exponentiel : 2^7 Opération : $\times$

Étape 1 — Détermination de la couche

$254 \bmod 100 = 54$

Étape 2 — Traversée des couches

Traversée : $100 \rightarrow 1 \rightarrow 2 \rightarrow \dots \rightarrow 54 \rightarrow$ **54 couches lues $\rightarrow$ 54 couches inversent tous leurs bits traversés**

Étape 3 — Lecture circulaire

Lecture circulaire dans chaque couche traversée.

Étape 4 — Mutation directionnelle

Selon le modèle :

- mutation locale (36 bits),
- mutation structurelle (128 bits).

10.8. Résumé du Chapitre 10

La logique de lecture d'une case dans TorUsNet 9.0 repose sur :

- la lecture circulaire déclenchée par un saut exponentiel + opérateur,
- la lecture des 100 couches comme un anneau vertical,
- l'intensité mod 100 qui détermine la couche lue,
- la mutation des couches traversées,
- **l'inversion automatique des bits traversés,**
- la cohérence fractale volumétrique,
- la fusion des lectures circulaires + volumétriques.

La lecture directionnelle devient :

- vivante,

- autonome,
- non linéaire,
- non inversable,
- impossible à dériver,
- impossible à falsifier.

Chapitre 11 — Nouvelle Logique de Lecture des Couches par Intensité

Version TorUsNet 9.0 — mise à jour complète

11.1. Vue d'ensemble

TorUsNet 9.0 introduit une nouvelle mécanique directionnelle : **la fusion entre l'intensité transmise sur le réseau et l'intensité privée interne.**

Chaque déplacement directionnel dans le Tore Privé dépend désormais de deux intensités :

1. **Intensité transmise** — envoyée dans le parcours transmis sur le réseau — visible par le Host et le Client — fixe pour la durée du parcours
2. **Intensité privée** — propre au Tore Privé — invisible sur le réseau — change à chaque déplacement — dépend de la case traversée — contient un additif ou une soustraction interne

La lecture des couches devient donc un mécanisme **hybride**, combinant :

- intensité externe (réseau),
- intensité interne (privée),
- lecture circulaire,
- lecture volumétrique,
- inversion des bits traversés,
- mutation directionnelle.

11.2. Intensité transmise (externe)

L'intensité transmise est incluse dans le parcours envoyé sur le réseau.

Elle sert à :

- déterminer la couche initiale à lire,
- déterminer le nombre de couches à traverser,
- synchroniser Host ↔ Client,
- stabiliser la cohérence directionnelle entre les deux tores.

11.2.1. Calcul de la couche initiale

$$\text{couche initiale} = \text{intensité transmise} \bmod 100$$

Exemples :

- Intensité transmise 87 → couche 87
- Intensité transmise 254 → $254 \bmod 100 = 54$
- Intensité transmise 10 000 → $10\,000 \bmod 100 = 0 \rightarrow$ couche 100

11.3. Intensité privée (interne)

L'intensité privée est **propre au Tore Privé**.

Elle :

- n'est jamais transmise,
- n'est jamais visible,
- n'est jamais exportée,
- change à chaque déplacement,
- dépend de la case traversée.

Chaque case du tore contient :

- un ID (16 bits),
- un bit normal/fantôme,
- 4 directions internes,
- un saut exponentiel,
- **un additif ou une soustraction d'intensité privée.**

11.3.1. Mise à jour de l'intensité privée

À chaque déplacement :

$$I_{\text{privé}}' \leftarrow I_{\text{privé}}' + A_{\text{case}}$$

où :

- A_{case} = additif ou soustraction interne de la case
- valeur typique : -7 à +7
- jamais transmise
- jamais visible

11.3.2. Intensité privée = intensité vivante

L'intensité privée :

- évolue à chaque déplacement,
- dépend de la géométrie interne,
- dépend des mutations,
- dépend des inversions normal/fantôme,
- dépend des couches traversées.

Elle est **vivante, oscillante, non dérivable**.

11.4. Fusion des intensités (externe + interne)

La lecture des couches dépend de :

$$I_{\text{final}} = I_{\text{transmise}} + I_{\text{privé}}'$$

Cette intensité finale détermine :

1. **la couche lue,**
2. **le nombre de couches traversées,**
3. **le nombre de couches mutées,**
4. **le nombre de couches inversées,**
5. **la profondeur directionnelle,**
6. **la vitesse de lecture circulaire.**

11.4.1. Calcul de la couche finale

$$\text{couche finale} = I_{\text{final}} 100$$

11.5. Lecture des couches par intensité finale

Une fois l'intensité finale calculée :

1. La couche finale est déterminée.
2. Le tore lit cette couche **ligne par ligne comme un cercle**.
3. Les couches traversées sont lues comme un **anneau vertical**.
4. **Tous les bits traversés s'inversent**.
5. Les couches traversées mutent.
6. La cohérence fractale est mise à jour.

11.5.1. Traversée des couches

Si $I_{\text{final}} = 254$:

- $254 \bmod 100 = 54$
- Traversée : $100 \rightarrow 1 \rightarrow 2 \rightarrow \dots \rightarrow 54$
- **54 couches lues**
- **54 couches inversent leurs bits**
- **54 couches mutent**

11.6. Interaction avec les modèles directionnels

Modèle 36 bits

- la clé ne parcourt pas le tore
- l'intensité transmise est utilisée
- l'intensité privée évolue à chaque déplacement interne
- la mutation locale dépend du résultat de la lecture
- **les bits traversés s'inversent**

Modèle 128 bits

- la clé parcourt le tore

- l'intensité privée évolue à chaque déplacement de la clé
- l'intensité finale modifie la lecture circulaire
- inversion normal/fantôme + inversion de lecture
- mutation structurelle dépendante de l'intensité finale

11.7. Exemple complet

Intensité transmise : 200 Intensité privée initiale : 17 Case traversée : additif interne = +3

Étape 1 — Mise à jour de l'intensité privée

$$I_{\text{privé}}' = 17 + 3 = 20$$

Étape 2 — Calcul de l'intensité finale

$$I_{\text{final}} = 200 + 20 = 220$$

Étape 3 — Détermination de la couche finale

$$220 \div 100 = 2$$

→ Lecture de la couche 20

Étape 4 — Traversée des couches

Traversée : 100 → 1 → 2 → ... → 20 → **20 couches lues** → **20 couches inversent leurs bits** → **20 couches mutent**

Étape 5 — Lecture circulaire

Lecture ligne par ligne comme un cercle.

Étape 6 — Mutation directionnelle

Selon le modèle :

- mutation locale (36 bits)
- mutation structurelle (128 bits)

11.8. Résumé du Chapitre 11

La nouvelle logique de lecture des couches par intensité repose sur :

- l'intensité transmise sur le réseau,
- l'intensité privée interne,

- la fusion des deux intensités,
- la lecture circulaire,
- la lecture volumétrique,
- l'inversion automatique des bits traversés,
- la mutation des couches traversées,
- la cohérence fractale vivante.

Cette mécanique rend :

- chaque lecture unique,
- chaque parcours non reproductible,
- chaque tore impossible à dériver,
- chaque session inviolable par conception.