**CAHIER DES CLAUSES TECHNIQUES PARTICULIERES**

**Évaluation et composition des logiciels depreuve à divulgation nulle de connaissance ou de calcul vérifiable**

**Marché n°2025-1295**

[**1.** **OBJET DU MARCHÉ** 3](#_Toc204161380)

[**2.** **CONTEXTE DU PROJET** 4](#_Toc204161381)

[**3.** **CONTENU DES PRESTATIONS ATTENDUES** 6](#_Toc204161382)

[Tâche 1 : Programmation, recensement et sélection de bibliothèques de preuve à divulgation nulle de connaissance et de calcul vérifiable 6](#_Toc204161383)

[Tâche 2a : Programmation de cas d’usage industriels, benchmarks systématiques 7](#_Toc204161384)

[Tâche 2b : interfaçage des bibliothèques 9](#_Toc204161385)

[Tâche 3a : consolidation et diffusion des cas d’usage 10](#_Toc204161386)

[Tâche 3b : consolidation et documentation des compositions 10](#_Toc204161387)

[**4.** **CONDITIONS D’EXÉCUTION** 10](#_Toc204161388)

[Accès aux outils informatiques 11](#_Toc204161389)

[Documents et moyens mis à disposition par Inria 11](#_Toc204161390)

[Interactions avec Inria 11](#_Toc204161391)

[Lois, règlements, normes et directives à respecter 11](#_Toc204161392)

[Éthique 11](#_Toc204161393)

[Format des livrables 11](#_Toc204161394)

[Lieu d’exécution 12](#_Toc204161395)

[Modalités de suivi de l’exécution du marché 12](#_Toc204161396)

[Documentation et réversibilité 12](#_Toc204161397)

[**5.** **OPERATIONS DE VERIFICATION DES PRESTATIONS** 13](#_Toc204161398)

[**6.** **Exigences techniques particulières attendues du titulaire** 13](#_Toc204161399)

|  |
| --- |
| 1. **OBJET DU MARCHÉ** |

La prestation consiste en *a)* une évaluation industrielle et technologique des logiciels de preuve de connaissance à divulgation nulle de connaissances (ou de calcul vérifiable) sur des exemples de nature industrielle, *b)* en la programmation d’interfaces entre ces logiciels. Ces logiciels permettent de prouver qu’un programme s’est correctement exécuté : un « prouveur » donne le résultat d’un calcul accompagné d’une « preuve » ou « certificat », ce qui permet à un « vérifieur » de s’assurer que ce calcul est correct sans devoir le refaire. De plus certaines données du calcul peuvent être rendues secrètes, par une technique cryptographique de « divulgation nulle de connaissance », *zero-knowlegde (ZK).*

Les applications sont nombreuses et prometteuses : authentification de photographies, preuve d’existence de filigrane (*watermark)*, preuve d’exécution de moteur de jeu (*speedrunning), etc.*,sans compter les applications à l’intelligence artificielle.

Nous appellerons de manière très abusive ces logiciels des logiciels **ZK** (*zero-knowledge)*. Cette terminologie abusive vient du monde des *blockchains*, qui est très demandeur de ces technologies et de leurs promesses, qui permettraient d’avoir plus de transactions par seconde et des transactions confidentielles. Dans la terminologie, on trouvera aussi les termes de *(zk-)snarks* et *(zk-)starks*.

Conséquemment, il y a une explosion du financement de ces technologies, et il s’ensuit un grand nombre de ces logiciels. De manière préliminaire, Inria en a recensé une dizaine : RISC0, Gnark, Lambdaworks, Arkworks, Winterfell, Plonky2, Halo2, Cairo, Midem VM, Matterlabs, Zisk, Triton-VM, liste non exhaustive et probablement incomplète.

Ces logiciels permettent de programmer facilement une application utilisant les snarks. Dans le cas de l’exemple *chess* deRISC0, un [programme court en Rust](https://github.com/risc0/risc0/blob/main/examples/chess/methods/guest/src/main.rs) décrit les trois coups qui permettent d’arriver à un échec et mat. Ce programme est ensuite exécuté, et le système RISC0 donne une preuve *ZK* que trois coups permettent d’arriver à un échec et mat, sans révéler ces trois coups. Cet exemple montre que RISC0 offre une abstraction et une généricité telle que le programmeur n’a pas à se soucier des mécanismes et des algorithmes cryptographiques.

Inria souhaite faire un recensement et une analyse systématiques de ces logiciels ainsi que des *benchmarks* sur des exemples avancés dans chaque bibliothèque recensée. Les travaux sont des travaux de développement de scripts et d’intégration de ces logiciels de manière à faire ces tests facilement.

Au lancement de la prestation, le cas d’usage considéré est celui de la preuve des transformations effectuées sur des images authentifiées (selon le standard C2PA). Pendant la réalisation des travaux, il est envisagé que l’Inria détermine d’autres cas d’usage industriels pertinents, et demande au titulaire de les intégrer dans les travaux.

De plus, l’Inria souhaite étudier comment ces bibliothèques peuvent s’interfacer. Chaque bibliothèque a développé sa technologie en silo, mais les techniques sous-jacentes varient beaucoup, avec leurs avantages et leurs inconvénients propres. Chaque logiciel peut avoir en interne une combinaison de techniques cryptographiques, et l’interfaçage proposé vise à combiner ces techniques d’un logiciel à l’autre, sans les recoder.

Le nombre de logiciels est élevé, et il semble vraisemblable que la bibliothèque RISC0 puisse servir de *hub* central, au vu de sa généricité, de ses performances, et de son avancée technologique.

Il y aura quatre tâches :

* **Tâche 1** : recensement, prise en main, et sélection de logiciels (durée maximum de 6 mois),
* **Tâche 2a** : programmation d’exemples industriels (durée maximum de 12 mois),
* **Tâche 2b** : interfaçage de ces logiciels (durée maximum de 12 mois),
* **Tâche 3** : consolidation, document et évaluation des résultats (durée maximum de 6 mois).

Les tâches 2a et 2b seront exécutées en parallèle.

Inria récupèrera une base logicielle qui devra être réutilisable facilement ensuite dans ses travaux de recherche pour mener des benchmarks et des évaluations facilement. Sur les cas d’usage pour lesquels la technologie s’est avérée performante, Inria envisage de possibles transferts à des partenaires industriels.

1. **CONTEXTE** **DU PROJET**

Inria est un établissement public à caractère scientifique et technologique placé sous la tutelle des ministres chargés de la recherche et de l’industrie. Créé en 1967 dans le cadre du « Plan Calcul », l’institut de recherche en informatique et en automatique (Iria) est devenu institut national (Inria) en 1979. Aujourd’hui, Inria se présente comme l’institut national de recherche dans les sciences et technologies du numérique.

L’Institut a principalement pour missions, dans les domaines de l’informatique, de l’automatique et des mathématiques appliquées , d'entreprendre des recherches fondamentales et appliquées, de réaliser des développements technologiques et des systèmes expérimentaux, d’organiser des échanges scientifiques internationaux, d’assurer le transfert et la diffusion des connaissances et du savoir-faire, de contribuer à la valorisation des résultats des recherches, et de développer une capacité d'expertise et d'appui aux politiques publiques.

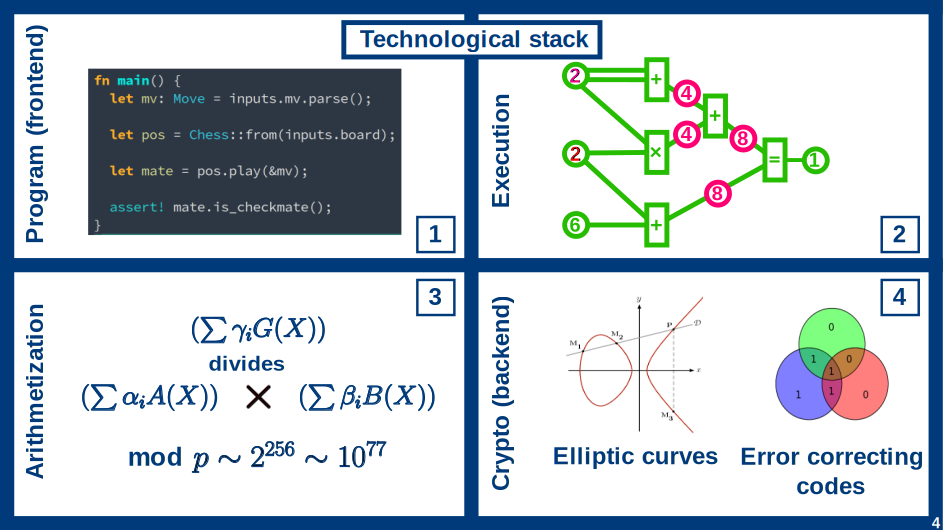
Inria a son siège à Rocquencourt, en Ile-de-France, et déploie son activité sur neuf centres de recherche (à Bordeaux, Grenoble, Lille, Lyon, Nancy, Paris, Rennes, Saclay et Sophia-Antipolis) et trois antennes (Montpellier, Pau et Strasbourg). L’activité de recherche d’Inria est menée au sein d’équipes-projets créées pour une durée de 4 ans, renouvelable ; ces équipes-projets sont pour la plupart communes avec des partenaires : universités, écoles, organismes de recherche, entreprises. Par ailleurs, Inria pilote au niveau national et de manière transverse aux différents centres des « programmes » et « missions » qui couvrent l’ensemble de ses activités (recherche, innovation, formation, expertise, etc.).

Créé en 2008, le Centre de recherche Inria de Saclay est situé au cœur du pôle d’excellence scientifique et technologique Paris-Saclay qui réunit à lui seul 15% de la recherche française. Au service du développement de l’Université Paris-Saclay et de l’Institut Polytechnique de Paris, le Centre Inria de Saclay compte 80 personnes travaillant dans les services d’appui à la recherche et 500 scientifiques issus de 54 nationalités.

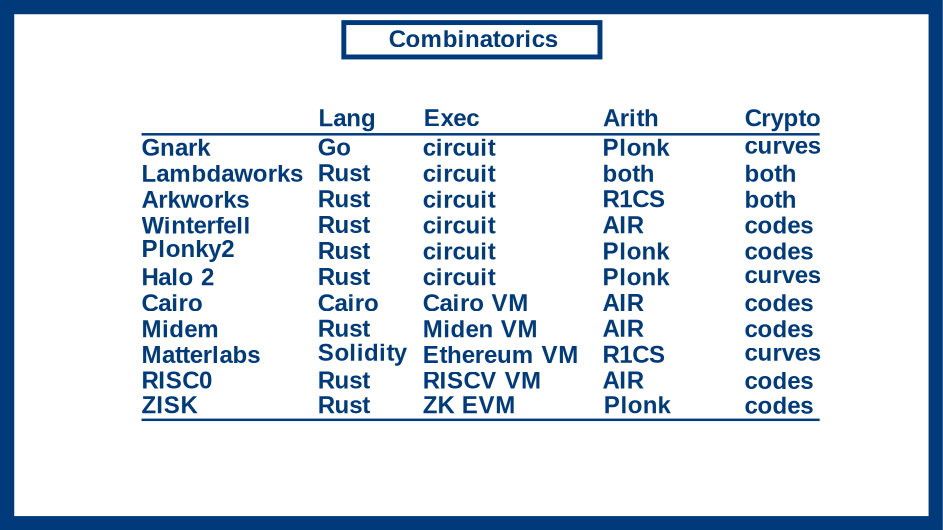
Bénéficiant d'une croissance continue, le centre totalise aujourd'hui 40 équipes-projets et deux en cours de création, dont 19 communes avec l’Institut Polytechnique de Paris, 16 avec l’Université Paris-Saclay, ainsi que 7 EP Inria dont une en collaboration avec l’Onera et une en collaboration avec le Pôle Universitaire Centre Val de Loire. Ces équipes de recherche sont réparties sur plus de dix sites.

L’équipe de cryptographie « Grace » du centre Inria de Saclay, commune avec le laboratoire d’informatique de l’École polytechnique, est spécialisée en cryptographie. Elle travaille sur la cryptographie à base ce courbes elliptiques, en classique et en post-quantique, et aussi sur la cryptographie à base de codes correcteurs et de réseaux euclidiens. Dans ce contexte, l’équipe a démarré en 2018 ses travaux sur le calcul vérifiable et les preuves *zero-knowledge* associées. Deux doctorants ont soutenu leur thèse avec succès en 2022, et deux thèses sont actuellement en cours.

En réalité, dans les détails de la réalisation informatique, les logiciels du domaine reposent sur une pile technologique présentant une grande complexité. Le processus est le suivant : le programme est décrit dans un langage de haut niveau. Il est ensuite compilé vers un modèle d’exécution virtuel, qui aussi bien peut être aussi rudimentaire qu’un circuit arithmétique, qu’une architecture de machine virtuelle. Ensuite, ce programme est exécuté selon ce modèle, et tous les états de la mémoire sont tracés pendant l’exécution. Une modélisation algébrique du la correction de l’exécution est faite (relation entre des polynômes sur les corps finis) Enfin, un post-traitement cryptographique permet de transformer cette trace en un condensat très court, sorte de certificat cryptographique qui peut être validé par un programme cryptographique de vérification.



Chaque étape de ce processus peut varier (langage de programmation, exécution, arithmétisation et traitement cryptographique), et les développeurs d’un système ZK doivent faire des choix, selon la technologie qu’ils veulent promouvoir. Il en découle un grand nombre de logiciels, selon les choix faits aux différentes étapes.



Le nombre, la variété et la complexité de ces logiciels implique un travail conséquent de programmation, déploiement, test, récolte des résultats et d’analyse. De plus l’Inria souhaite tester ces logiciels sur des cas d’usage plus avancés, *proof-of-*concept, que les *benchs* publiés dans les conférences et revues académiques.

Il est devenu clair qu’au vu du nombre de ces logiciels (une dizaine) et de leur variété, leur prise en main et leur évaluation systématique ne permettent pas à un doctorant de faire un état de l’art autre que des références bibliographiques, car la tâche de mener une évaluation systématique de qualité ces logiciels est trop lourde et chronophage pour pouvoir être exécutée dans la durée d’une thèse. De plus ces tâches, par leur ampleur et leur nature, sortent du cadre de ce qui est attendu dans un travail de doctorat,

Dans le cadre d’un financement de deux ans par Programme Inria Quadrant (PIQ), l’équipe « Grace » va mener une action d’analyse et de traitement systématiques des logiciels de preuve à divulgation nulle de connaissances et de calcul vérifiable. Ces logiciels seront de plus testés sur différentes architectures matérielles, ce qui contribue significativement à l’ampleur de l’évaluation.

L’équipe, de manière préliminaire, a identifié la bibliothèque RISC0 comme environnement prometteur et de manière générale, l’architecture RISC-V semble très en vogue comme cible d’exécution.

1. **CONTENU DES PRESTATIONS ATTENDUES**

Tâche 1 : Programmation, recensement et sélection de bibliothèques de preuve à divulgation nulle de connaissance et de calcul vérifiable

Il faudra recenser (voir liste non exhaustive donnée ci-dessus) les principales bibliothèques, analyser leurs principales caractéristiques industrielles (niveau de maturité, cadre logiciel, soutien industriel, communauté et dynamique, licences logicielles), par une prise en main de chacun d’entre eux.

Le titulaire analyse leurs principales caractéristiques industrielles (niveau de maturité, cadre logiciel, soutien industriel, communauté et dynamique, licences logicielles).

Il déterminera leurs caractéristiques techniques : langage de description du calcul, modèle d’exécution (circuit arithmétique, registres, VM), arithmétisation et arithmétique associée (AIR, PLONK, R1CS), système de preuve (codes correcteurs, courbes elliptiques, réseaux euclidiens).

Pour cela, un cas d’usage prédéfini sert de vecteur de test. Le scénario est le suivant : une image est signée électroniquement et une transformation a été appliquée à cette image pour donner une nouvelle image.

Il faudra coder la preuve de l’énoncé suivant : «la signature de l’image d’origine a été vérifiée et la nouvelle image est bien le résultat de la transformation donnée appliquée à l’image d’origine ».

On se rapprochera du cadre fourni par le standard C2PA [https://c2pa.org](https://c2pa.org/) d’authentification de photographies professionnelles, à des fins de mesure de performance, sans toutefois programmer (dans un premier temps) exactement les spécifications du standard. Les images considérées et les transformations seront cependant de même ordre de grandeur de complexité que celles typiques du standard.

Le titulaire devra être capable de gérer plusieurs environnements complexes (Rust, Go, C++ et des langages spécifiques comme Cairo) et des logiques de build/développement variés.

Il devra aussi pouvoir faire une analyse de la qualité, de la communauté, de la vivacité, du soutien industriel et de l’évolution prévisible de ces logiciels.

**Livrable attendu :**

1. Un document « **L1 »** avec les éléments suivants
   1. Recensement les logiciels avec leurs caractéristiques
   2. Code source et instructions de compilation et d’exécution de l’exemple « images authentifiées » pour chaque bibliothèque recensée
   3. Analyse des bibliothèques à fins de sélection pour les tâches suivantes

Tâche 2a : Programmation de cas d’usage industriels, benchmarks systématiques

Il s’agira de coder des cas d’usages de nature industrielle.

Par défaut, le cas d’usage « images authentifiées » servira de test avec le standard c2pa. Il est possible voire attendu que d’autres cas d’usage soient amenés par Inria, issus de ses contacts industriels.

Ce ou ces cas d’usage seront testés sur chaque bibliothèque sélectionnée dans la tâche 1.

De plus, au moment de la rédaction de ce cahier des clauses techniques et particulières, la plateforme d’exécution la plus performance n’est pas encore identifiée, et il est vraisemblable qu’elle varie selon les logiciels considérés. Cette tâche vise aussi à déterminer le meilleur contexte d’exécution.

L’usage de l’informatique en nuage est pertinent pour déployer systématiquement des benchmarks sur une variété d’architectures matérielles.

Cette mission présente une certaine élasticité, car, au moment de la rédaction de ce cahier des clauses techniques particulières, le nombre de logiciels qui auront été sélectionnés en phase 1 et le nombre de cas d’usage apportés par Inria ne sont pas connus (hormis l’exemple des transformations d’images). Ainsi, cette tâche pourrait aussi bien consister en deux cas d’usage avec deux bibliothèques que trois cas d’usage avec cinq bibliothèques.

Il n’est donc pas possible de déterminer au préliminaire le volume d’activité de la tâche 2a.

Le financement PIQ permet d’abonder de manière dynamique les fonds alloués à l’équipe « Grace », et cela permettra d’augmenter la demande d’Inria.

Le titulaire devra pouvoir se montrer souple et s’adapter.

Ainsi, pour chaque cas d’usage, il y a une unité d’œuvre. Toutefois, il est assuré que le cas d’usage « images authentifiées » est un cas de base qui devra être traité, et la prestation correspondante détermine le coût minimal. Les cas d’usages supplémentaires feront l’appel de bons de commande. Il y a au minimum une unité d’œuvre « images authentifiées ».

La prestation du cas d’usage « images authentifiées » présente les objectifs suivants, **pour chaque logiciel sélectionné en tâche 1 :**

1. Création et vérification de preuve de la vérification du certificat de naissance de photographies issues d’appareils C2PA
2. Création et vérification de preuve de transformation
3. Combinaison des deux preuves précédentes
4. Création et vérification de preuves ultérieures, pour des transformations opérées suite à la première
5. Déploiement sur des services d’informatique en nuage
6. Benchmarks systématiques et scriptés sur les couples logiciel-plateforme
7. Reporting scripté des benchmarks.

Une difficulté est de se rapprocher du contexte industriel.

En collaboration avec l’Inria, le titulaire construira une banque de quelques images et de quelques transformations.

En cas de problèmes de droit sur les images et de logiciel associé (Photoshop ou autres), le problème sera contourné avec des images ayant des droits utilisables, et des logiciels *open-*source présentant des capacités de traitement aux standard industriels.

De même, d’éventuels droits sur les certificats C2PA (si nécessaires) peuvent poser problème.

Chaque cas d’usage donnera lieu à une deuxième phase de sélection de bibliothèques pour la tâche 3, de consolidation.

Les bibliothèques qui ne passent pas à l’échelle seront abandonnés dans la tâche 3.

Il faudra factoriser et dégager un maximum de code et de scripts communs à différents cas d’usage, permettant à Inria de capitaliser dessus.

**Livrable(s) attendu(s) :**

1. **L2a/cas-d-usage/bibliothèque.**

**Pour chaque couple cas d’usage/bibliothèque, un livrable**

**(par exemple « L2a/images-authentifiées/Winterfell »).**

1. Une base de code et une mini banque d’exemples de taille réelle permettant de les exécuter, ainsi que les scripts de provisionnement, de déploiement, et d’exécution sur les architectures en nuage.
2. Scripts d’exécution des *benchmarks*, scripts de récolte et de mise en forme des résultats**.**
3. Chacune de ces bases de code devra être documentée et inclure une suite de tests avec une bonne couverture déployable sur toutes les architectures cibles.
4. Description de la meilleure architecture d’exécution
5. **L2a/outil-benchmark**

S’il y a possibilité de factoriser les codes et les scripts commun à plusieurs cas d’usage, ce livrable contiendra cette partie commune

Tâche 2b : interfaçage des bibliothèques

Chaque logiciel ou bibliothèque de preuve ZK a deux composantes, un algorithme de preuve et un algorithme de vérification. Pour interfacer les systèmes, il faut prouver dans une bibliothèque donnée qu’une preuve a été vérifiée dans une autre bibliothèque. Concrètement, pour deux logiciels de preuve ZK **A** et **B**, il faut faire une preuve dans **B** de l’exécution correcte du vérifieur de **A**, donnant une preuve de type « **AB** », vérifiable par le logiciel **B**.

Il faut plutôt travailler au niveau des interfaces logicielles, et RISC0 semble un choix pertinent pour servir de nœud central (*hub)*. RISC0 permet en théorie de prouver l’exécution de n’importe quel exécutable (ELF) sur l’architecture RISC-V, cet exécutable pouvant être issu d’un langage de plus au niveau comme C++ ou Rust.

Ainsi, si une bibliothèque tierce autre que RISC0 permet de compiler son vérifieur vers le format ELF pour RISCV, RISC0 est en théorie capable de faire une preuve l’exécution de ce vérifieur. Le travail est un travail de glu et d’interface entre les logiciels, même à travers des langages de haut niveau différent, puisque le langage commun est celui des ELF RISC-V, qui est une cible commune de compilation.

Il faut déterminer si cette capacité d’interfaçage par exécutable ELF est assez robuste pour interfacer RISC0 avec les vérifieurs des autres bibliothèques.

Ces compositions et interfaçage seront d’abord testés sur des exemples simples, et, si les performances sont prometteuses, mis à l’épreuve dans le cadre des benchmarks et preuves de concept produits en parallèle dans la tâche 2a.

Le volume de la tâche 2b est lui aussi élastique en fonction des logiciels sélectionnés par la tâche 1. Il y a une unité d’œuvre pour chaque logiciel autre que RISC0 sélectionné dans la tâche 1.

*Nota bene : RISC0 est donné en titre indicatif. Il semble vraisemblable que ce choix persiste, mais la dynamique du domaine du ZK peut amener à choisir un autre logiciel pour servir de « hub ». Cependant, il n’est pas prévu de travailler avec plusieurs « hubs ».*

**Livrable(s) attendu(s) :**

1. **« L2b/composition/bibliothèque »**

(un livrable par logiciel autre que RISC0, **e**xemple « L2b/Winterfell »)

Pour chaque bibliothèque autre que RISC0, des programmes et des scripts qui permettent de faire des vérifications en RISC0 des preuves issues de cette autre bibliothèque. Rapport de performance sur diverses architectures en nuage

Tâche 3a : consolidation et diffusion des cas d’usage

Consolidation du code, des scripts et la documentation de l’outil permettant de systématiser et des récolter les résultats de benchmarks sur des architectures en nuage (avec le provisionnement), à fins de réutilisation dans l’équipe de recherche « Grace » de l’Inria.

Consolidation du code, des scripts et de la documentation fournis pour chaque couple cas d’usage/bibliothèque performant. Rapport sur les performances sur les diverses architectures en nuage ;

Dans le cas des diffusions *open-source,* il faudra veiller à la clarté du code et de la documentation, au choix de la plateforme (*readthedocs, github, etc.)*, aux éventuels tutoriels, etc.

Des outils d’intégration continue seront utilisés.

**Livrable(s) attendu(s) :**

1. **L3a/outil-benchmark :**  partie commune du code, des scripts et documentation
2. **L3a/cas-d-usage/bibliothèque** dans les cas réussis : code « version 1 » avec une documentation diffusable (code source, scripts de déploiement et d’exécution, code d’exécution et de récolte des résultats des benchmarks)
3. **L3a/rapport-négatif** listant et motivant les couples cas d’usage/bibliothèque non concluant. Ce rapport n’est pas destiné à être diffusé.

Tâche 3b : consolidation et documentation des compositions

Consolidation du code, évaluation de la robustesse des compositions (par exemple selon la durabilité prévisible des APIs, le soutien industriel), documentation, rapport sur les performances sur diverses architectures. Il n’est pas prévu le même niveau de diffusion que pour les cas d’usage.

**Livrable(s) attendu(s) :**

1. **L3b/composition/bibliothèque**. Pour chaque composition, code, scripts et documentation, rapport de performance
2. **L3b/rapport-négatif** listant et motivant les compositions non concluantes. Ce rapport n’est pas destiné à être diffusé.
3. **CONDITIONS D’EXÉCUTION**

Accès aux outils informatiques

La forge logicielle gitlab.inria.fr sera utilisée pour la gestion, la sauvegarde et le traçage des programmes et des scripts, avec l’outil « git ».

Pour la tâche 1, il n’est pas envisagé que les calculs demandés soient de grande envergure, et les exemples pourront être traités localement sur des machines de travail standard du titulaire, du niveau « station de travail ».

Pour la tâche 2a, il est attendu que le titulaire loue lui-même diverses infrastructures de serveurs en nuage, selon des critères classiques (nombre de cœurs CPU, nombre de cœurs GPU, mémoire) pour des *benchmarks* et des *proof-of-concept* de plus grande envergure.

L’accès à ces serveurs sera donné aux chercheurs d’Inria.

Documents et moyens mis à disposition par Inria

Les documents de la soumission PIQ tout-zk, comprenant les objectifs et les jalons généraux du projet. Inria apportera son expertise pour éclairer si besoin sur les différentes briques technologiques de la pile ZK (notamment arithmétisation, *backend* crypto).

Interactions avec Inria

Le titulaire interagira avec Daniel Augot et Gustavo Banegas sur le projet et la conduite du projet.

Il interagira si besoin avec d’autres chercheurs de l’équipe Grace travaillant sur les thématiques de preuve ZK et de calcul vérifiable, sur des questions de fond liées aux systèmes cryptographiques de preuves et à l’architecture RISC-V et aux machines virtuelles. Ces interactions se feront sous trois formes :

1. Visioconférences deux fois par mois,
2. Présentation des résultats aux deux comités de pilotage,
3. Une visite trimestrielle (*coding sprint)* du titulaire dans les locaux d’Inria, d’une durée d’un à deux jours (selon le besoin).

Lois, règlements, normes et directives à respecter

Le titulaire s’engage également à respecter les licences logicielles et les droits d’usage des documents utilisés. Il n’y pas lieu dans ce projet de collecter ni de traiter de données personnelles.

Éthique

Il est attendu que les personnes employées par le titulaire travaillent dans un cadre matériel confortable, et dans un cadre professionnel respectueux de leur personne et leur activité, de leur autonomie, et que le titulaire veille à limiter *au maximum* les facteurs de stress et risques psycho-sociaux.

Format des livrables

Tous les documents livrables, en particulier les documentations, doivent être mis à disposition sous forme électronique (pas d’impression) sous un format éditable (*markdown*, *latex*, etc.), avec une version figée et datée.

En ce qui concerne le code source, un tag ou équivalent permettra aussi de fixer la date et l’état du livrable. De plus, il est vraisemblable que le livrable **L3a/cas-d-usage/bibliothèque** soit en réalité une évolution et une amélioration du livrable **L2a/cas-d-usage/bibliothèque.**

Si c’est le cas, cela devraêtre identifiable. Toutefois, ce n’est pas une obligation, et si le titulaire le souhaite, deux documents d’évolution distincte sont recevables.

La forge gitlab.inria.fr devrait permettre de gérer de manière évolutive les livrables.

Lieu d’exécution

Les développements informatiques se feront dans les locaux et le cadre du travail du titulaire, sauf les coding sprints et les réunions précités qui se feront dans les locaux d’Inria.

La tâche 1 peut se faire sur les machines du titulaire.

Pour la tâche 2b, et les livrables L3a/cas-d-usage/bibliothèque les évaluations de performance seront faites sur des architectures en nuage typiques. Ces serveurs en nuage seront loués par le titulaire.

Modalités de suivi de l’exécution du marché

* Réunions de suivi et de partage d’information toutes les deux semaines, en visioconférence.
* *Coding sprint* dans les locaux d’Inria tous les trois mois.
* Dans un délai maximum de six mois à compter du début de la première tâche, puis et dans un délai maximum de 18 mois de la première tâche, un comité de pilotage se réunira pour déterminer les travaux des tâches suivantes. Il aura notamment la tâche de sélectionner les logiciels et les cas d’usage lors d’un passage d’une phase à l’autre.

Ce comité de pilotage est composé de : Daniel Augot (directeur de recherche), Gustavo Banegas (chargé de recherche), un ingénieur de développement Inria, un chargé de projets partenariat et innovation Inria. Le titulaire sera invité aux réunions de ce comité pour éclairer les choix qui restent de la responsabilité d’Inria.

Documentation et réversibilité

Les livrables (code source, scripts, et documentation) et les rapports (« négatifs ») seront versés à Inria, dans la forge gitlab.inria.fr. Les résultats (reproductibles) des benchmarks seront versés à Inria. Le code source sera hébergé, sauf contrainte technique spécifique, sur la forge logicielle gitlab.inria.fr et devient la propriété d’Inria.

Des *coding sprint* précités auront lieu dans les locaux d’Inria, en présentiel tous les trois mois, avec déplacement de personnel du titulaire. Ces évènements permettront de transférer les compétences acquises à Inria.

1. **OPERATIONS DE VERIFICATION DES PRESTATIONS**

* **Tâche 1**: Inria devra pouvoir, avec l’aide du titulaire, prendre en main sur ses machines les logiciels traités par le titulaire, sur la base des dépôts dans gitlab.inria.fr
* **Tâche 2a** : Inria devra pouvoir, avec l’aide du titulaire, accéder au déploiement en nuage des cas d’usages traités, sur la base des dépôts dans gitlab.inria.fr, par exemple par accès SSH. Inria testera les scripts de benchmarks et de récolte de leurs résultats.
* **Tâche 2b** : Inria devra pouvoir, avec l’aide du titulaire, exécuter les compositions sur ses propres machines (ordinateurs portables puissants), sur la base des dépôts dans gitlab.inria.fr
* **Tâche 3a** : livrable L3a/cas-d-usage/bibliothèque, Inria fera une révision du code et des scripts, une relecture de la documentation, sur la base des dépôts dans gitlab.inria.fr. Inria devra pouvoir exécuter les cas d’usage en nuage, par accès SSH.

La documentation destinée à être diffusée publiquement fera l’objet d’un soin particulier, quitte à subir plusieurs allers-retours entre Inria et le titulaire. Cette documentation-là sera en Anglais

* **Tâche 3b** : Inria fera une révision du code, des scripts, de la documentation, sur la base des dépôts dans gitlab.inria.fr

1. **EXIGENCES TECHNIQUES PARTICULIERES ATTENDUES DU TITULAIRE**

La grande variété des bibliothèques entraîne une grande variété de langages de développement (Rust, Go, C++, langage dédié, domain specific language). Il faudra montrer la capacité de mobiliser des compétences sur tous ces points. Toutefois, comme le montre l’exemple [chess](https://github.com/risc0/risc0/blob/main/examples/chess/methods/guest/src/main.rs) (16 lignes de code Rust), l’effort de codage est en lui-même est peu important, et le « nombre de lignes de codes » à produire sera vraisemblablement peu conséquent.

En revanche, pour la conduite de benchmarks systématiques, il faudra montrer des compétences et une expérience solide sur les scripts de benchmarks, de provisionnement de machines en nuage, d’exécution de ces benchmarks et de récolte de ces résultats, avec mise en forme (tableaux, figures).

Il n’est pas demandé de compétences particulières cryptographiques, par exemple en algorithmique cryptographique, algèbre, théorie des nombres, etc., car les prestations ne portent pas sur ces mécanismes précis. Une compréhension des primitives cryptographiques standard sera utile, notamment des fonctions de hachage cryptographiques et des signatures électroniques.

En revanche, le titulaire devra pouvoir abstraire suffisamment l’informatique, pour intégrer le cheminement (workflow) programme → exécution selon modèle d’exécution→ trace d’exécution → preuve, et ensuite le cheminement (programme, preuve) → vérification → acceptation de la preuve.

Traiter la variété des langages, des cas d’usage, la mise en place de benchmarks avec une récolte de résultats utiles, ainsi que la technologie ZK et les abstractions associées, nécessite un pilotage de projet d’assez haut niveau, avec une vision profonde de l’informatique. La quantité de code à fournir ne semble pas a priori considérable, mais une bonne compréhension de la demande est déterminante pour assurer la qualité de la prestation.