PROSECCO : formally proven PROtections for Secured Compiled Code

Workshop du projet

Projet ANR AAP 2015 – Durée 48 mois

Genèse (2014-2015)

- Les attaques sur le matériel soit par canaux auxiliaires, soit par faute sont un enjeu considérable
- Les industriels de l'industrie de la carte à puce sont au fait
  - Composants *tamper resistant*
  - Programmation « sécurisée » ou prenant en compte ces attaques
  - Savoir faire dédié, nombreuses recettes de cuisine *ad hoc*
- Monde de l'IoT et de l'embarqué
  - Composant *off the shelf*
  - Programmation orientée fonctionnalité
  - Pas de savoir faire dédié

Genèse

- Les contre-mesures contre les attaques restent manuellement ajoutées
  - Niveau source avec problématique d'optimisation et de *refactoring* de code
  - Niveau assembleur avec problématique de portabilité
- L' automatisation de l'insertion de contre-mesures est recherchée
  - Réduction des coûts
  - Capitalisation du savoir faire
  - Déploiement adapté à niveau de sécurité souhaité

Genèse

- L’ évaluation de l’efficacité (et présence) des contre-mesures contre les attaques est un processus long et fastidieux
  - Revue de code niveau source et assembleur
  - Simulation
  - Tests expérimentaux
- Besoin d’analyses de robustesse de code binaire / assembleur
  - Post-compilation
  - Automatisation
  - Exhaustivité versus scalabilité, formelles versus expérimentales

Genèse : besoins et problématiques identifiés

- Application automatisée de protections contre attaques en faute et SCA
  - Généricité nécessaire (vs. dédié)
  - À la compilation : compromis portabilité vs optimisation de code
- Protections versus besoins en sécurité contre attaques en faute et SCA
  - Différentes techniques de protection, à des coûts et efficacités variables
  - Combinaisons ou superpositions efficaces ?
- Garantie de la bonne insertion et de l'efficacité des protections
  - Problématique liée à la certification
  - Besoin de preuve de la présence des protections
  - Besoin d'analyse de l'efficacité des protections (seules, combinées ou empilées)

Objectifs du projet PROSECCO

- Automatisation de l'insertion de protections à la compilation
  - Contre les attaques en fautes et par canaux cachés
  - Combinaison de protections
- Méthodes d'évaluation de la robustesse de code binaire protégé
  - Automatisées et basées sur une approche formelle
  - Analyse de la présence et de l'efficacité des protections
  - Contre les attaques en fautes et par canaux cachés
- Validation expérimentale
  - Caractérisation d'une cible typique embarqué / IoT
  - Réalisation d'attaques physiques : code cible avec et sans protection

Partenaires : participants et compétences

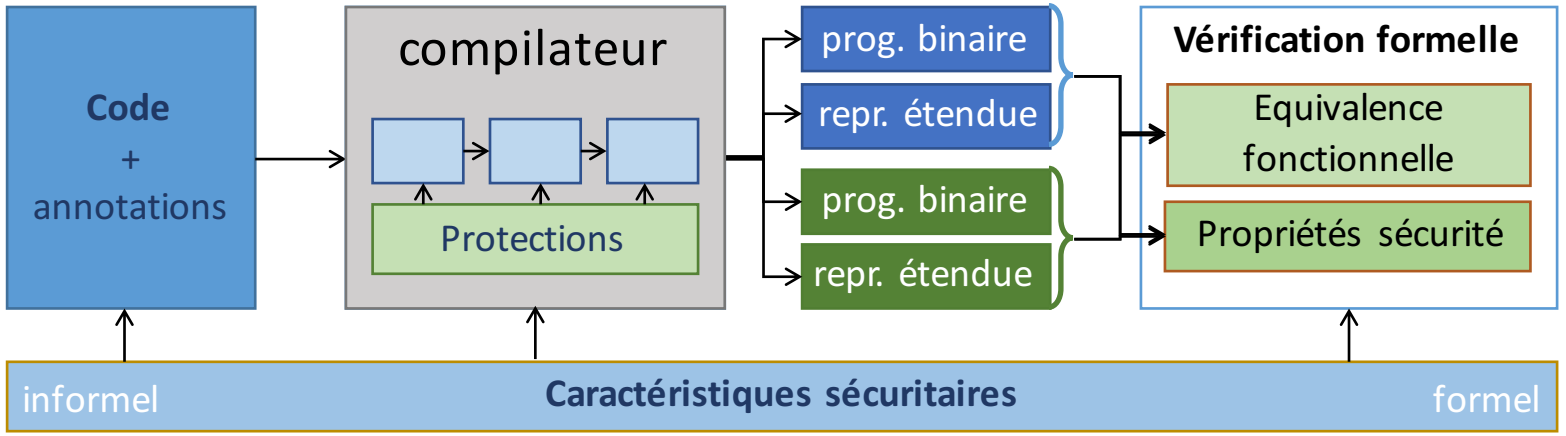
- CEA
  - Compilation statique, génération au *runtime* de code (toolchain COGITO),
  - Attaques physiques et leur modélisation, contre-mesures HW et SW
  - 2 permanents : D. Couroussé, P-A Moëllic
  - 1 doctorant et 1 post-doctorant : N. Belleville, B. Colombier
- Sorbonne-Université
  - Modélisation des injections de fautes, contre-mesures logicielles contre les fautes
  - Compilation et optimisation de code
  - Vérification formelle de protections et de séquences de code assembleur protégé contre les fautes
  - 3 permanents : E. Encrenaz, K. Heydemann, Q. Meunier
  - 2 doctorants (+1) : I. Ben El Ouahma, J-B Bréjon (+ Son Tuan Vu)

Point de départ du projet : état des lieux 2015

- Suite de la thèse N. Moro co-encadrée par le CEA – Sorbonne université
  - CEA : Thèse Thierno Barry en cours
    Automatisation d'un schéma de protection à la compilation (par duplication)
  - LIP6 : Travail de Lucien Goubet en cours
    Automatisation de la vérification formelle de la robustesse de code assembleur protégé (séquence)
- Projet ANR COGITO (CEA – INRIA –EMSE)
  - Génération au runtime de code polymorphe

Objectifs du projet

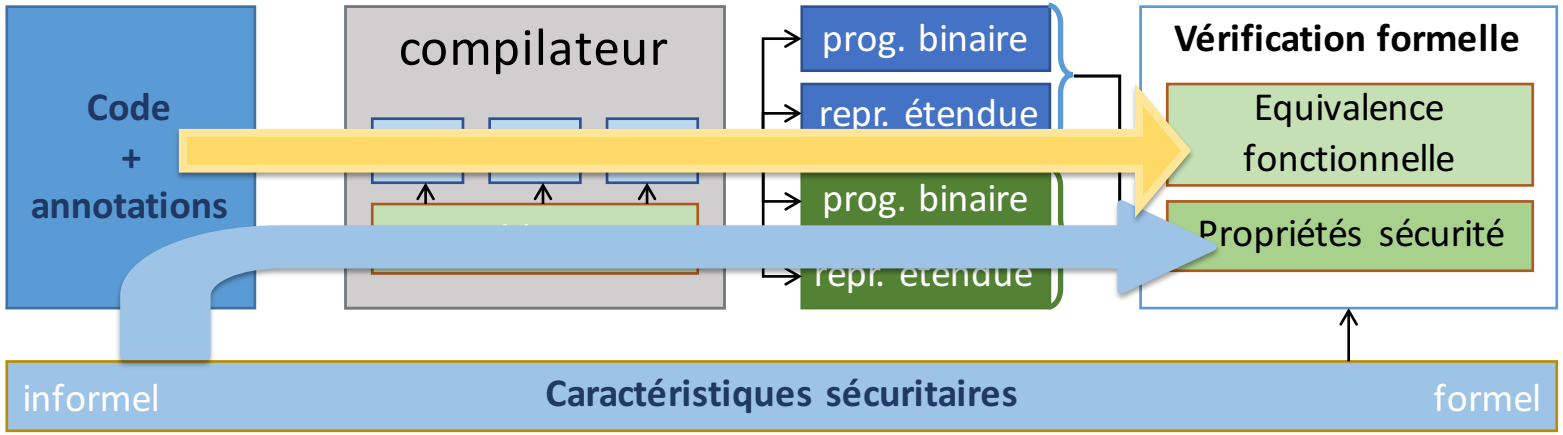
- Application automatisée à la compilation de contre-mesures logicielles
  - ⇒ via l'ajout de différentes passes de sécurisation
  - ⇒ code compilé avec une combinaison de protections
- Vérification de la fonctionnalité et de la sécurité du code sécurisé
  - ⇒ modèles et outils pour la vérification de propriétés de sécurité sur une représentation étendue du code et d'équivalence fonctionnelle



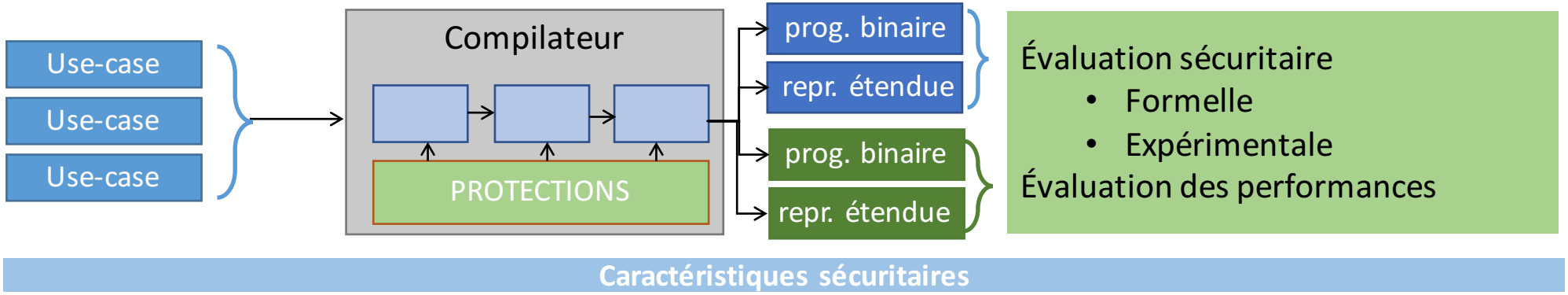
Objectifs du projet

Expression de propriétés de sécurité

- Annotation du code : données et zones sensibles + attaques considérées sur ces zones ou données
- Vérification formelle : propriétés logiques pour équivalence fonctionnelle et robustesse pour un modèle d'attaque
- Lien entre les 2 via le compilateur

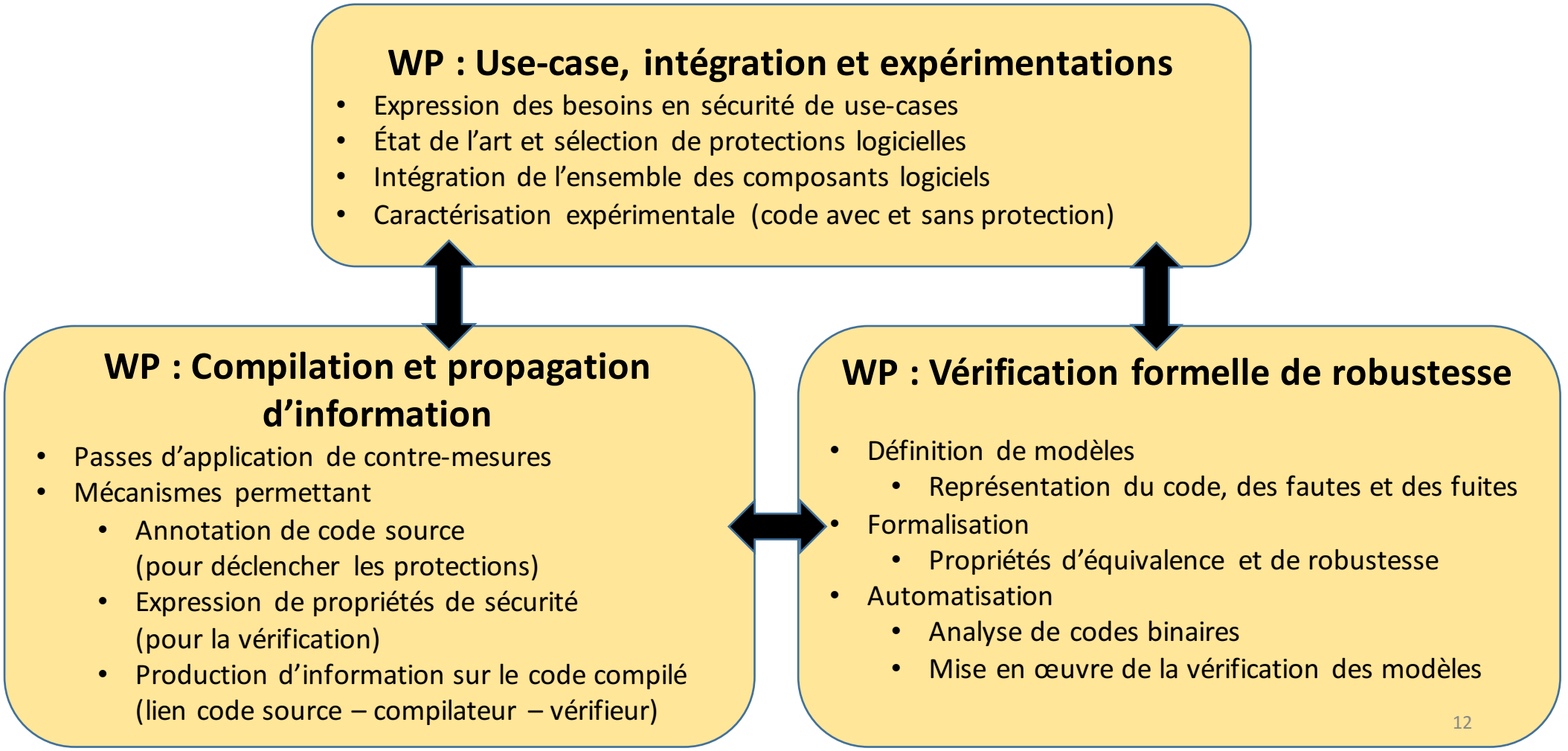


Objectifs du projet



- Sélection de use-cases
  - Difficulté croissante : verifyPIN, AES, authentification d'un *secure boot*
  - Expression des propriétés de sécurité souhaitées (éléments sensibles, menaces)
- Classification des principes de protections existantes
- Sélection des protections
  - À déployer sur les use-cases en cohérence avec propriétés de sécurité
  - Étude de la possibilité de leur vérification dans le projet

Structuration du projet



Dissémination

- Séminaires, exposés, posters
  - 2 présentations à la conférence Phisic en Mai 2018
  - 1 présentation au workshop CPSEd
  - 1 présentation au workshop CryptARCHI
  - Posters à WISG 2016, 2017, 2018
  - Keynote CS2, tutorial à la conférence HOST, séminaire lors de la journée Architecture COMPAS 2018
- Publications
  - Workshop PROOFS
  - Revue ACM TACO
  - Différentes soumissions en cours : 1 journal + 1 chapitre de livre + 1 conférence + 1 workshop
- Dépôt de brevet
  - Schéma de protection de l'intégrité d'exécution

Conclusion

- Projet à T0 + 35 mois
- Vise à démontrer la faisabilité de l'application automatisée de différentes protections et la vérification du code protégé automatiquement
  - Sur un ensemble de use-cases (VerifyPIN, AES, extrait d'un secure boot)
  - Avec des tests expérimentaux
- Objectifs de la journée : avoir vos retours
  - sur les travaux réalisés et en cours
  - sur l'intérêt de l'approche scientifique dans un contexte industriel ou applicatif donné
  - sur les problématiques liées à une mise en œuvre de l'approche dans un contexte industriel