

Parcours :

- 2013 - 2017 : Élève Normalien à l'ENS Cachan (License, Master MPRI, Agrégation)
- Sep. 2017 - Sep. 2020 : Thèse au LSV et LORIA avec Hubert Comon et Steve Kremer
- Nov. 2020 - Jul. 2022 : Postdoc au CISPA (Allemagne) avec Cas Cremers
- Sep. 2022 - . : Postdoc à Inria Paris avec Bruno Blanchet

Méthodes formelles en sécurité

Charlie Jacomme

21 Mars 2023

La sécurité de nos données et le droit à la vie privée sont essentiels !

Ma question scientifique

Comment fournir des garanties **formelles**, i.e., des preuves de sécurité ?

Protocoles



SSH

GPG

TLS (HTTPS)

...

La sécurité informatique en général

Matériel



OS



Implémentations



C++
Java
Python
...

Primitives



RSA
AES
...

Protocoles



SSH
GPG
TLS (HTTPS)
...

Utilisateurs



La sécurité informatique en général

Matériel



OS



Implémentations



C++
Java
Python
...

Primitives



RSA
AES
...

Protocoles



SSH
GPG
TLS (HTTPS)
...

Utilisateurs



La sécurité informatique en général

Matériel

OS

Implémentations

Primitives

Protocoles

Utilisateurs



C++
Java
Python
...



RSA
AES
...



SSH
GPG
TLS (HTTPS)
...



Objectif

En supposant que les primitives sont sécurisées, obtenir des preuves de sécurité des protocoles dans des modèles aussi **réalistes** que possible.

Objectif

En supposant que les primitives sont sécurisées, obtenir des preuves de sécurité des protocoles dans des modèles aussi **réalistes** que possible.

Problème

Faire des preuves devient vite trop **compliqué**.

Objectif

En supposant que les primitives sont sécurisées, obtenir des preuves de sécurité des protocoles dans des modèles aussi **réalistes** que possible.

Problème

Faire des preuves devient vite trop **compliqué**.

Solution : preuves assistée par ordinateur (depuis les années 2000)

Des programmes nous aident à **faire, vérifier ou automatiser** les preuves.

(**Proverif**, **Tamarin**, **DeepSec**, **EasyCrypt**, **CryptoVerif**, **Squirrel**, ...)

Les deux modèles principaux

Modèle symbolique

Modèle calculatoire

Les deux modèles principaux

Modèle symbolique

([Proverif](#), [Tamarin](#), [DeepSec](#), ...)

Modèle calculatoire

([EasyCrypt](#), [CryptoVerif](#), [Squirrel](#), ...)

Les deux modèles principaux

Modèle symbolique

(*Proverif*, *Tamarin*, *DeepSec*, ...)

Messages :

Termes abstraits

Modèle calculatoire

(*EasyCrypt*, *CryptoVerif*, *Squirrel*, ...)

Séquences de bits (+)

Les deux modèles principaux

Modèle symbolique

(*Proverif*, *Tamarin*, *DeepSec*, ...)

Messages :

Termes abstraits

Attaquant :

Règles d'inférences

Modèle calculatoire

(*EasyCrypt*, *CryptoVerif*, *Squirrel*, ...)

Séquences de bits (+)

Machine de Turing (+)

Les deux modèles principaux

Modèle symbolique

(*Proverif*, *Tamarin*, *DeepSec*, ...)

Messages :

Termes abstraits

Attaquant :

Règles d'inférences

Automatisation :

Beaucoup (+)

Modèle calculatoire

(*EasyCrypt*, *CryptoVerif*, *Squirrel*, ...)

Séquences de bits (+)

Machine de Turing (+)

Peu

Les deux modèles principaux

Modèle symbolique

(*Proverif*, *Tamarin*, *DeepSec*, ...)

Messages :	Termes abstraits
Attaquant :	Règles d'inférences
Automatisation :	Beaucoup (+)
Protocole :	Spécification complète (+)

Modèle calculatoire

(*EasyCrypt*, *CryptoVerif*, *Squirrel*, ...)

Séquences de bits (+)
Machine de Turing (+)
Peu
Le cœur en isolation

Les deux modèles principaux

Modèle symbolique

(*Proverif*, *Tamarin*, *DeepSec*, ...)

Messages :	Termes abstraits
Attaquant :	Règles d'inférences
Automatisation :	Beaucoup (+)
Protocole :	Spécification complète (+)

Modèle calculatoire

(*EasyCrypt*, *CryptoVerif*, *Squirrel*, ...)

Séquences de bits (+)
Machine de Turing (+)
Peu
Le cœur en isolation

État de l'art

- De nombreux outils utilisés avec succès, à la fois pour prouver la sécurité ou découvrir des vulnérabilités sur des systèmes complexes.

Les deux modèles principaux

Modèle symbolique

(*Proverif*, *Tamarin*, *DeepSec*, ...)

Messages :	Termes abstraits
Attaquant :	Règles d'inférences
Automatisation :	Beaucoup (+)
Protocole :	Spécification complète (+)

Modèle calculatoire

(*EasyCrypt*, *CryptoVerif*, *Squirrel*, ...)

Séquences de bits (+)
Machine de Turing (+)
Peu
Le cœur en isolation

État de l'art

- De nombreux outils utilisés avec succès, à la fois pour prouver la sécurité ou découvrir des vulnérabilités sur des systèmes complexes.
- Encore beaucoup de limitations, et c'est toujours difficile de manipuler des modèles réalistes.

Contributions principales

I - Faciliter les preuves calculatoires

II - Faciliter les preuves symboliques

III - Des études de cas

Contributions principales

I - Faciliter les preuves calculatoires

- Un résultat de composition pour des preuves **modulaires**. [CJS- CCS'20 (A*)]

II - Faciliter les preuves symboliques

III - Des études de cas

Contributions principales

I - Faciliter les preuves calculatoires

- Un résultat de composition pour des preuves **modulaires**. [CJS- CCS'20 (A*)]
- Complexité/décidabilité de l'équivalence de programmes probabilistes sur des corps finis. [BJK - LICS'20 (A*), ACM TOCL (A)] [BFGGJS - CCS'18 (A*)] [BGJKS - CSF'19 (A)]

II - Faciliter les preuves symboliques

III - Des études de cas

Contributions principales

I - Faciliter les preuves calculatoires

- Un résultat de composition pour des preuves **modulaires**. [CJS- CCS'20 (A*)]
- Complexité/décidabilité de l'équivalence de programmes probabilistes sur des corps finis. [BJK - LICS'20 (A*), ACM TOCL (A)] [BFGGJS - CCS'18 (A*)] [BGJKS - CSF'19 (A)]
- **Squirrel**, un assistant de preuve pour la sécurité des protocoles, contre des attaquants **quantiques**. [BDJKM - S&P'21 (A*)] [CFJ - S&P'22 (A*)]

II - Faciliter les preuves symboliques

III - Des études de cas

Contributions principales

I - Faciliter les preuves calculatoires

- Un résultat de composition pour des preuves **modulaires**. [CJS- CCS'20 (A*)]
- Complexité/décidabilité de l'équivalence de programmes probabilistes sur des corps finis. [BJK - LICS'20 (A*), ACM TOCL (A)] [BFGGJS - CCS'18 (A*)] [BGJKS - CSF'19 (A)]
- **Squirrel**, un assistant de preuve pour la sécurité des protocoles, contre des attaquants **quantiques**. [BDJKM - S&P'21 (A*)] [CFJ - S&P'22 (A*)]

II - Faciliter les preuves symboliques

- Détaillée dans la suite : [Hash Gone Bad - CCDHJK - USENIX'23 (A*)]
- [CJJK - USENIX'22 (A*)] [JKS - EuroS&P'17 (-)] [CJL - CSF'23 (A)]

III - Des études de cas

Contributions principales

I - Faciliter les preuves calculatoires

- Un résultat de composition pour des preuves **modulaires**. [CJS- CCS'20 (A*)]
- Complexité/décidabilité de l'équivalence de programmes probabilistes sur des corps finis. [BJK - LICS'20 (A*), ACM TOCL (A)] [BFGGJS - CCS'18 (A*)] [BGJKS - CSF'19 (A)]
- **Squirrel**, un assistant de preuve pour la sécurité des protocoles, contre des attaquants **quantiques**. [BDJKM - S&P'21 (A*)] [CFJ - S&P'22 (A*)]

II - Faciliter les preuves symboliques

- Détaillée dans la suite : [Hash Gone Bad - CCDHJK - USENIX'23 (A*)]
- [CJJK - USENIX'22 (A*)] [JKS - EuroS&P'17 (-)] [CJL - CSF'23 (A)]

III - Des études de cas

- Analyse de Signal : [CJN - USENIX'23 (A*)] **Nouvelle publication du dossier**
- [JK - CSF'18 (A), ACM TOPS (A)] [JKKR - USENIX'23 (A*)]

Améliorations des modèles symboliques de primitives

$$\begin{aligned} \text{dec}(\text{enc}(m, \text{sk}), \text{sk}) \\ = m \end{aligned}$$

$$\text{dec}(\text{enc}(m, \text{sk}), \text{sk}) \\ = m$$

$$h(x)$$

...

L'abstraction et le monde réel

$\text{dec}(\text{enc}(m, \text{sk}), \text{sk})$
 $= m$

$h(x)$

...

SHA-3
AES-CCM RSA EdDSA
ECDSA SHA-256
X25519 HMAC Ed25519
ChaCha20Poly1305
SHA-1 X448 AES
DSA AES-GCM
Ed448

L'abstraction et le monde réel

$$\text{dec}(\text{enc}(m, \text{sk}), \text{sk}) \\ = m$$

$h(x)$

...



SHA-3
AES-CCM RSA EdDSA
ECDSA SHA-256
X25519 HMAC Ed25519
ChaCha20Poly1305
SHA-1 X448 AES
DSA AES-GCM
Ed448

Faiblesses

- Une collision fixée :

$$\exists c_1, c_2. \text{SHA-1}(c_1) = \text{SHA-1}(c_2)$$

Faiblesses

- Une collision fixée :

$$\exists c_1, c_2. \text{SHA-1}(c_1) = \text{SHA-1}(c_2)$$

- Collisions à préfixes choisis (CPC) :

$$\forall p_1, p_2. \exists s_1, s_2. \text{SHA-1}(p_1 \| s_1) = \text{SHA-1}(p_2 \| s_2)$$

Faiblesses

- Une collision fixée :

$$\exists c_1, c_2. \text{SHA-1}(c_1) = \text{SHA-1}(c_2)$$

- Collisions à préfixes choisis (CPC) :

$$\forall p_1, p_2. \exists s_1, s_2. \text{SHA-1}(p_1 \| s_1) = \text{SHA-1}(p_2 \| s_2)$$

- ...

Dans un monde parfait

Intégrer les **faiblesses** connues des primitives classiques dans les modèles, en préservant l'**automatisation**.

Hash Gone Bad - Automated discovery of protocol attacks that exploit hash function weaknesses

*Cheval, Cremers, Dax, Hirschi, **Jacomme**, Kremer (USENIX'23)*

**Automated Analysis of Protocols that use Authenticated Encryption:
How Subtle AEAD Differences can impact Protocol Security**

*Cremers, Dax, **Jacomme**, Zhao (Article Soumis)*

L'approche évidente

Pour chaque faiblesse concrète que l'on connaît, ajouter une **équation** qui modélise ce comportement.

Une collision fixe sur les fonctions de hachage

functions: $h/1$, $c1/0$, $c2/0$

equations: $h(c1) = h(c2)$

L'approche évidente

Pour chaque faiblesse concrète que l'on connaît, ajouter une **équation** qui modélise ce comportement.

Une collision fixe sur les fonctions de hachage

functions: $h/1, c1/0, c2/0$

equations: $h(c1) = h(c2)$

Limites des outils symboliques

Collisions à prefixes choisis (CPC) : on a besoin d'une concaténation **associative**, pour laquelle l'unification est infinitaire.

Un renversement de paradigme

Des fonctions contrôlées par l'attaquant

```
compute Hash(x) =  
    in(y);  
    event IsHash(x,y);  
    return y
```

Un renversement de paradigme

Des fonctions contrôlées par l'attaquant

```
compute Hash(x) =  
    in(y);  
    event IsHash(x,y);  
    return y
```

Restrictions

- Déterministe : $\forall x \ y1 \ y2. \text{IsHash}(x,y1) \ \& \ \text{IsHash}(x,y2) \Rightarrow y1=y2$

Un renversement de paradigme

Des fonctions contrôlées par l'attaquant

```
compute Hash(x) =  
    in(y);  
    event IsHash(x,y);  
    return y
```

Restrictions

- Déterministe : $\forall x \ y1 \ y2. \text{IsHash}(x,y1) \ \& \ \text{IsHash}(x,y2) \Rightarrow y1=y2$
- Sans Collision : $\forall x1 \ x2 \ y. \text{IsHash}(x1,y) \ \& \ \text{IsHash}(x2,y) \Rightarrow x1=x2$

Un renversement de paradigme

Des fonctions contrôlées par l'attaquant

```
compute Hash(x) =  
    in(y);  
    event IsHash(x,y);  
    return y
```

Restrictions

- Déterministe : $\forall x \ y1 \ y2. \text{IsHash}(x,y1) \ \& \ \text{IsHash}(x,y2) \Rightarrow y1=y2$
- Sans Collision : $\forall x1 \ x2 \ y. \text{IsHash}(x1,y) \ \& \ \text{IsHash}(x2,y) \Rightarrow x1=x2$
- CPC : $\forall x1 \ x2 \ y. \text{IsHash}(x1,y) \ \& \ \text{IsHash}(x2,y) \Rightarrow$
 $x1=x2$
or $\exists p1,p2. x1=p1 || c1(p1,p2) \ \& \ x2=p2 || c2(p1,p2)$

Des fonctions contrôlées par l'attaquant

- Modélisation plus **expressive**.
- Plus **efficace** sur des cas complexes, e.g., pour les CPC.
- L'approche par restriction ouvre sur le long terme une nouvelle piste pour la **correction calculatoire**.

Des fonctions contrôlées par l'attaquant

- Modélisation plus **expressive**.
- Plus **efficace** sur des cas complexes, e.g., pour les CPC.
- L'approche par restriction ouvre sur le long terme une nouvelle piste pour la **correction calculatoire**.

Un dernier détail

Doit-on faire ces modélisations pour chaque outils au **cas par cas** ?

La plateforme Saptic+

Permet d'exporter un unique modèle vers différents outils
(Proverif, Tamarin, DeepSec).

- Permet de combiner les avantages disjoints des outils.
- Les traductions sont **prouvées** : un résultat prouvé dans un outil peut servir de lemme dans un autre, et on combine formellement les garanties.
- Les traductions sont **optimisées** : peu de perte de performances.

[Saptic+: protocol verifiers of the world, unite! - CJKK - **USENIX'22** (A*)]

La plateforme Saptic+

Permet d'exporter un unique modèle vers différents outils
(Proverif, Tamarin, DeepSec).

- Permet de combiner les avantages disjoints des outils.
- Les traductions sont **prouvées** : un résultat prouvé dans un outil peut servir de lemme dans un autre, et on combine formellement les garanties.
- Les traductions sont **optimisées** : peu de perte de performances.

[Saptic+: protocol verifiers of the world, unite! - CJKK - **USENIX'22** (A*)]

Bonus: on développe les modèles avancés dans Saptic+, qui sont directement valides dans les autres outils !

Le protocole EDHOC

Un protocole en cours de **standardisation** à l'IETF pour tous les objets connectés.

- Analyse précise du protocole avec **Sapic+** et nos **modèles avancés** de primitives.
- Implication dans le développement du standard.
 - Vulnérabilités trouvées et solutions adoptées (issue git & pull requests)
 - Modèles **mis à jour** à chaque nouvelle version.

[A comprehensive, formal and automated analysis of the EDHOC protocol - JKKR - **USENIX'23** (A*)]

Et après ?

Garanties formelles sur un TLS post-quantique ?

Fournir des analyses **précises** pendant toutes les phases de conception des protocoles.

Garanties formelles sur un TLS post-quantique ?

Fournir des analyses **précises** pendant toutes les phases de conception des protocoles.

Défis

- On a besoin de **combiner** les forces de tous les outils, mais il est impossible de maintenir à jour un modèle par outils.

Garanties formelles sur un TLS post-quantique ?

Fournir des analyses **précises** pendant toutes les phases de conception des protocoles.

Défis

- On a besoin de **combiner** les forces de tous les outils, mais il est impossible de maintenir à jour un modèle par outils.
 - ⚠ **Sapic+** ne résout ce point que pour le modèle symbolique.

Garanties formelles sur un TLS post-quantique ?

Fournir des analyses **précises** pendant toutes les phases de conception des protocoles.

Défis

- On a besoin de **combiner** les forces de tous les outils, mais il est impossible de maintenir à jour un modèle par outils.
 - ⚠ **Sapic+** ne résout ce point que pour le modèle symbolique.
- Il faut des preuves **modulaires**.

Garanties formelles sur un TLS post-quantique ?

Fournir des analyses **précises** pendant toutes les phases de conception des protocoles.

Défis

- On a besoin de **combiner** les forces de tous les outils, mais il est impossible de maintenir à jour un modèle par outils.
 - ⚠ **Sapic+** ne résout ce point que pour le modèle symbolique.
- Il faut des preuves **modulaires**.
 - ⚠ Mon résultat est puissant mais **complexe** à utiliser.

Garanties formelles sur un TLS post-quantique ?

Fournir des analyses **précises** pendant toutes les phases de conception des protocoles.

Défis

- On a besoin de **combiner** les forces de tous les outils, mais il est impossible de maintenir à jour un modèle par outils.
 - ⚠ **Sapic+** ne résout ce point que pour le modèle symbolique.
- Il faut des preuves **modulaires**.
 - ⚠ Mon résultat est puissant mais **complexe** à utiliser.
- Il faut autant d'**automatisation** que possible.

Garanties formelles sur un TLS post-quantique ?

Fournir des analyses **précises** pendant toutes les phases de conception des protocoles.

Défis

- On a besoin de **combiner** les forces de tous les outils, mais il est impossible de maintenir à jour un modèle par outils.
 - ⚠ **Sapic+** ne résout ce point que pour le modèle symbolique.
- Il faut des preuves **modulaires**.
 - ⚠ Mon résultat est puissant mais **complexe** à utiliser.
- Il faut autant d'**automatisation** que possible.
 - ⚠ L'automatisation des outils calculatoires reste **faible**.

Objectifs

- Vérification large d'un protocole, avec de très nombreux scénarios vérifiés automatiquement avec les outils symboliques, et les scénarios les plus critiques vérifiés avec les outils calculatoires Squirrel/CryptoVerif.
- Utilisation d'un résultat de Squirrel dans CryptoVerif et inversement.

Objectifs

- Vérification large d'un protocole, avec de très nombreux scénarios vérifiés automatiquement avec les outils symboliques, et les scénarios les plus critiques vérifiés avec les outils calculatoires Squirrel/CryptoVerif.
- Utilisation d'un résultat de Squirrel dans CryptoVerif et inversement.

Étapes

- 1 Étendre la sémantique du langage de Sapic+ avec une version calculatoire.
- 2 Concevoir des traductions prouvées $\text{CryptoVerif} \leftrightarrow \text{Sapic+} \leftrightarrow \text{Squirrel}$.
 - ⚠ Définition d'une nouvelle manière d'exprimer la sécurité en CryptoVerif.
- 3 Validation via l'application à l'amélioration des études de cas de TLS 1.3

Objectifs

- Application automatique de mon résultat de composition dans Sapic+.
- Avoir de la modularité dans tous les outils majeurs du domaine.

Objectifs

- Application automatique de mon résultat de composition dans **Sapic+**.
- Avoir de la modularité dans tous les outils majeurs du domaine.

Étapes

- 1 **Vérification** automatique des conditions d'applicabilité du théorème.
- 2 **Synthèse** automatique de certains des paramètres.
 - ⚠ Nouvelle notion de sous-programme le plus général vis-à-vis d'une valeur.
- 3 Implémentation **Sapic+**, puis validation sur le cas TLS 1.3.

Objectifs (parallèles)

- Vérification **avancée** du futur TLS post-quantique.

Objectifs (parallèles)

- Vérification **avancée** du futur TLS post-quantique.
 - ↪ Suivre le développement des futurs standards, avec des analyses au fur et à mesure.

Objectifs (parallèles)

- Vérification **avancée** du futur TLS post-quantique.
 - ↪ Suivre le développement des futurs standards, avec des analyses au fur et à mesure.
- Amélioration de l'**automatisation** des preuves calculatoires.

Objectifs (parallèles)

- Vérification **avancée** du futur TLS post-quantique.
 - ↪ Suivre le développement des futurs standards, avec des analyses au fur et à mesure.
- Amélioration de l'**automatisation** des preuves calculatoires.
 - ↪ Explorer des nouvelles pistes de résultats de **correction calculatoire** (une preuve en **Proverif/Tamarin** serait réutilisable en **CryptoVerif/Squirrel**).

Mes apports : Post-Quantique - Modèles de primitives - Outils

IRISA - Rennes

LMF - Paris Saclay

LIMOS - Clermont-Ferrand

Mes apports : Post-Quantique - Modèles de primitives - Outils

IRISA - Rennes

- Collaborations existantes (D. Baelde, S. Delaune, J.ALLEMAND)
- ★ Implication dans les standards (Mohamed Sabt).

LMF - Paris Saclay

LIMOS - Clermont-Ferrand

Mes apports : Post-Quantique - Modèles de primitives - Outils

IRISA - Rennes

- Collaborations existantes (D. Baelde, S. Delaune, J.ALLEMAND)
- ★ Implication dans les standards (Mohamed Sabt).

LMF - Paris Saclay

- Collaborations existantes (C. Fontaine, G. Scerri)
- ★ Approfondissement des modèles d'attaquants quantiques (Pablo Arrighi).

LIMOS - Clermont-Ferrand

Mes apports : Post-Quantique - Modèles de primitives - Outils

IRISA - Rennes

- Collaborations existantes (D. Baelde, S. Delaune, J. Lallemand)
- ★ Implication dans les standards (Mohamed Sabt).

LMF - Paris Saclay

- Collaborations existantes (C. Fontaine, G. Scerri)
- ★ Approfondissement des modèles d'attaquants quantiques (Pablo Arrighi).

LIMOS - Clermont-Ferrand

- Insertion naturelle avec Pascal Lafourcade sur les protocoles.
- ★ Étude de protocoles industriels.

Parcours

- 2013 - 2017 : Élève Normalien à l'ENS Cachan (License, Master MPRI, Agrégation)
- Sep. 2017 - Sep. 2020 : Thèse au LSV et LORIA avec Hubert Comon et Steve Kremer.
- Nov. 2020 - Jul. 2022 : Postdoc au CISPA (Allemagne) avec Cas Cremers.
- Sep. 2022 - . : Postdoc à l'Inria Paris avec Bruno Blanchet.

Résumé du CV académique et nouveautés

- Prix de thèse du **GdR sécurité** et **accessit Gilles Kahn**
- Conférences = 9 A*, 3 A, 1 New ; Journaux = 2 A
dont 1 nouvelle publication à **USENIX'23** (A*)
- 25 collaborateurs (Rennes, Paris, Nancy, Nice, Bochum, Tel-Aviv, New-York, Sarrebruck)
- **Comité de Programme** de INDOCRYPT'21, CSF'22, CSF'23 et EuroS&P'23
- Implication dans la standardisation à l'**IETF** du protocole EDHOC.