Bugs, virus, intrusions, pirates, . . .

Tout un bestiaire de menaces et pas une seule parade ?

Si, les maths !

Thomas Genet
ISTIC - IRISA
Université de Rennes 1

18 octobre 2014

Plan

Plan

Un bestiaire de menaces : le bug

Bug

Erreur dans le texte d'un programme

- 1990 : AT&T, Appels longues distance, 9h00 de blocage
  Instruction mal placée dans le programme
- 1996 : Ariane 5, Navigation, ??? M€

  Dépassement arithmétique dans une variable du programme

- 1998 : USS Yorktown, Propulsion, 7h00 de blocage
  Une division par 0 dans le programme
- 1999 : Satellite Mars Climate Orbiter, Navigation, 125 M€
  Un module compte en mètres et l'autre en pouces

Un bestiaire de menaces : les virus/malwares

Malware

Programme développé dans le but de nuire à un système informatique, sans le consentement de l'utilisateur infecté.

Comme types de nuisances, un malware peut :

- espionner l'ordinateur où il se trouve
- offrir une porte dérobée à des pirates informatiques (cheval de Troie)
- détruire des données sur l'ordinateur où il se trouve, ...

Virus

Malware qui prend le contrôle d'un autre programme et se propage en se recopiant dans des logiciels légitimes.

Le virus informatique Stuxnet continue de toucher l'Iran

Le Monde.fr avec AFP, AP et Reuters | 27.09.2010 à 12h25 • Mis à jour le 27.09.2010 à 20h36

La minute du g
des consoles

Stuxnet s'attaque aux systèmes Windows à l'aide de quatre attaques dont trois « zero day ». [...]

Le virus est inoculé par des clés USB infectées ; il contamine ensuite d'autres ordinateurs WinCC du réseau à l'aide d'autres exploits.

Stuxnet est le premier virus découvert qui espionne et reprogramme des systèmes industriels, ce qui comporte un risque élevé. Il cible spécifiquement les systèmes SCADA utilisés pour le contrôle commande de procédés industriels. Stuxnet a la capacité de reprogrammer des automates programmables industriels produits par Siemens et de camoufler ses modifications. Les automates programmables Siemens sont utilisés tant par quelques centrales hydro-électriques ou nucléaires que pour la distribution d'eau potable ou les oléoducs. [...]

Le virus a affecté 45 000 systèmes informatiques, dont 30 000 situés en Iran, y compris des PC appartenant à des employés de la centrale nucléaire de Bouchehr. Les 15 000 autres systèmes informatiques sont des ordinateurs et des centrales situés en Allemagne, en France, en Inde et en Indonésie, utilisateurs de technologies Siemens. [...]

Top Categories

Security Software
Hacking
Privacy
Social Media
Top Threats

SEE ALL »

Trending Tags

malware
vulnerabilities
vulnerability
patch
antivirus
apple
adobe

SEE ALL »

Follow

Android Malware Makes Up This Week's Dangerous Apps List

May 17, 2013 10:28 AM EST | 0 Comments

By Fahmida Y. Rashid

Originally flagged by researchers from Lookout Mobile Security, Google removed several apps which used the BadNews advertising network from Google Play last month. Savage Knife for Android was the most well-known app on that list (which included several Russian-language apps).

As it turned out, the BadNews advertising network was really bad news for the users. Not only did this ad network aggressively push ads on to the user's mobile device, it was also responsible for sending out "malicious advertising content" such as links and payloads. Once the payload was installed, it performed various tasks on the device without the user knowing anything about it, such as sending SMS messages to prime rate numbers.

Le nombre de malwares Android augmente de 400% au second trimestre

Déjouant les prévisions des cabinets de sécurité

LES RUBRIQUES (ACTU, FORUMS, TUTOS) DE DÉVELOPPEZ

- Android
- Mobiles
- Sécurité
- Systèmes

RÉSEAUX SOCIAUX

4 0

Tweet Share on LinkedIn

Discussion forum

Le 05/07/2012, par Hinault Romaric, Responsable Actualités

Le nombre d'utilisateurs des terminaux Android évolue à

La rançon de ce succès est l'intérêt croissant des pirates
l'OS, déjouant les statistiques des cabinets de sécurité.

Le cabinet de sécurité Trend Micro, dans son rapport pour le seco
augmentation de près de 400 % par rapport au chiffre présenté au pr

La société qui prédisait une augmentation du nombre de malwares A
dont le nombre a littéralement doublé par rapport à ses prévisions.

Alors que Trend Micro mettait les utilisateurs en garde contre les gal
endroit dangereux pour les achats.

17 applications malveillantes ont été téléchargées plus de 700 000 foi

Tweet

Pin it

submit

SECURITY SOFTWARE app store, apps, apple, malware, antivirus

iOS app contains potential malware

Lex Friedman
@lexfri

May 2, 2013 12:26 PM

An app available for download from Apple's iOS App Store contains an embedded Trojan horse. And while the good news is that you're almost definitely safe from any malware danger, there's still reason for concern. The app itself is almost certainly harmless—and the malicious code is probably present unintentionally—but the fact that the code slipped through the App Store's review process isn't ideal.

The app Simply Find It, a $2 game from Simply Game, seems harmless enough. But if you run Bitdefender Virus Scanner—a free app in the Mac App Store—it will warn you about the presence of a Trojan horse within the app. A reader tipped *Macworld* off to the presence of the malware, and we confirmed it.

Apple declined to comment on the issue

Un bestiaire de menaces : les pirates informatiques

Pirate informatique

Personne qui utilise tous les moyens possibles pour accéder à des données protégées, ou prendre le contrôle d'un système protégé, etc.

Méthodes préférées des pirates :

- Virus, malware (Chevaux de troie)
- Outils d'espionnage du réseau (Scanners)
- Ingénierie sociale (« le principal bug est l'utilisateur »)
- Hameçonage (Phishing)
- ...

Des internautes iraniens victimes d'une vaste tentative d'espionnage

Le Monde.fr | 30.08.2011 à 16h31

Google a annoncé, lundi soir, avoir détecté une tentative de faude informatique généralisée visant principalement des internautes iraniens, qui permettait d'espionner les communications d'utilisateurs de services Google. La tentative

Une bonne et une mauvaise nouvelle

La bonne

Toutes ces menaces ont une seule origine :

les bugs (ou faiblesses) dans les programmes des ordinateurs, téléphones, ...

La mauvaise

Garantir l'absence de bugs dans un programme est très difficile :

- la taille importante des programmes empêche la vérification manuelle
- la vérification automatique (par un autre programme) est impossible

Théorème (Rice, 1953)

Toute propriété non triviale sur le langage reconnu par une machine de Turing est indécidable (non vérifiable par calcul).

Qu'est-ce que les mathématiciens ont à voir avec mon téléphone ?

Théorème (Rice, 1953)

Toute propriété non triviale sur le langage reconnu par une machine de Turing est indécidable (non vérifiable par calcul).

Basé sur des résultats de « mathématiques » des années 1930 (A. Turing)

15 ans avant la construction du premier ordinateur (1946), grâce aux mathématiques, on savait déjà qu'ils ne pourraient pas tout calculer.

D'autres exemples de décidabilité/indécidabilité :

- Arithmétique sur $\mathbb{N}$ avec $+$, décidable
  Existe-t-il des valeurs pour x et y telles que $x + 3 \geqslant y + x + 5$?
- Arithmétique sur $\mathbb{N}$ avec $+$ et $\times$, indécidable
  Existe-t-il des valeurs pour x, y et z telles que $x \times y \geqslant x \times z + 18$?

Plan

Calculs et programmes

- On peut voir un programme comme un ensemble de règles de calcul
- On peut voir l'exécution d'un programme comme un calcul

Exemple (Programme de simplification d'expressions arithmétiques)

Soient les règles de simplification suivantes :

- $y + 0 = y$
- $y \times 0 = 0$

L'expression $(B + 0) + (3 \times (A \times 0))$ se simplifie en :

$$
\begin{aligned}
&= (B + 0) + (3 \times 0) \\
&= (B + 0) + 0 \\
&= B + 0 \\
&= B
\end{aligned}
$$

Démonstration du calcul en Isabelle/HOL

Comment être sûr qu'un programme fait ce qu'il doit faire ?

- Problème 1 : comment décrire « ce qu'il doit faire » ?
  Spécification= Texte décrivant les fonctionnalités du programme
- Problème 2 : comment vérifier qu'un programme respecte sa spécification ?

Exemple

1. Donnez la spécification du programme `fastPower`.
2. Comment vérifier que le programme réalise sa spécification ?

Ces deux problèmes (difficiles) sont responsables des menaces que l'on a vu

Vérifier un programme est une tâche difficile

- Analyser à la main même un petit programme est dur
- Les programmes utilisés quotidiennement sont énormes

Exemple (Complexité de l'analyse d'un programme)

Needham Schroder Public Key Protocol (1978)

1. $A \hookrightarrow B : \{A, N_A\}_{K_B}$
2. $B \hookrightarrow A : \{N_A, N_B\}_{K_A}$
3. $A \hookrightarrow B : \{N_B\}_{K_B}$

Programme analysé par des chercheurs pendant 17 ans. Bug (énorme) trouvé par un ordinateur en 1995.

Exemple (Taille des programmes)

- Environ 6 millions d'éléments dans un Boeing 747
- Environ 12 millions de lignes de code dans Android
  - Assemblage de 40 (énormes) projets open source différents
  - Répartis en plus de 50000 fichiers
  - ... produits par des dizaines de milliers de développeurs

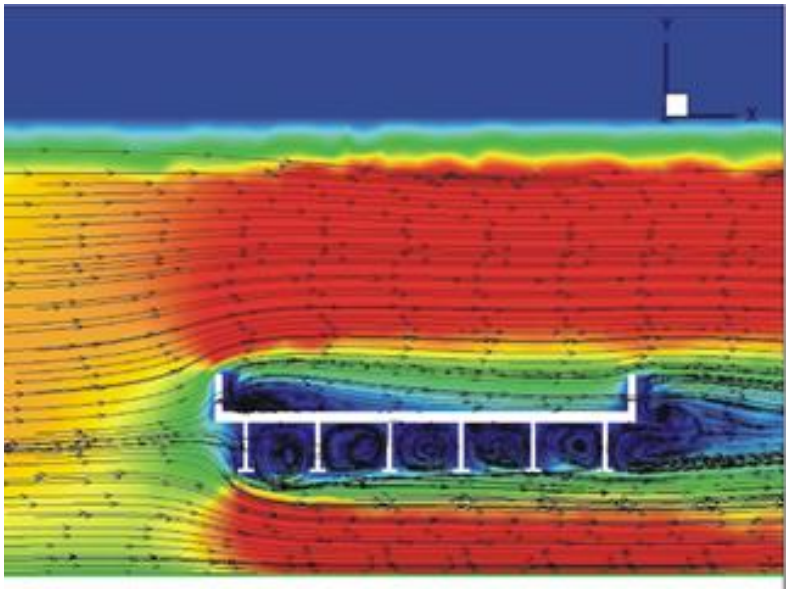
Des pistes pour venir à bout de tous ces problèmes ?

Utiliser les mathématiques pour :

- Spécifier le programme
- Modéliser le fonctionnement du programme
- Vérifier automatiquement que le modèle respecte la spécification

Plan

La modélisation est une pratique courante d'ingénierie



C'est nécessaire pour détecter les problèmes avant la construction !

Quelles maths pour modéliser/spécifier les programmes ?

Essentiellement des maths discrètes :

- Fonctions et arithmétique

$$\begin{cases} f : & \mathbb{N} \mapsto \mathbb{N} \\ & x \mapsto x + 3 \end{cases}$$

- Théorie des ensembles

  $e \in \{1, 2, 3\}$ $\qquad A \cap A = A$ $\qquad A \cap \varnothing = \varnothing$

- Logique

  $A \wedge B$ $\qquad A \vee B$ $\qquad A \longrightarrow B$

- Des combinaisons de tout ceci... (Démo `sum` Isabelle/HOL)

Qu'est-ce que cela permet de faire ?

- Problème sur le modèle $\longrightarrow$ Problème sur le programme

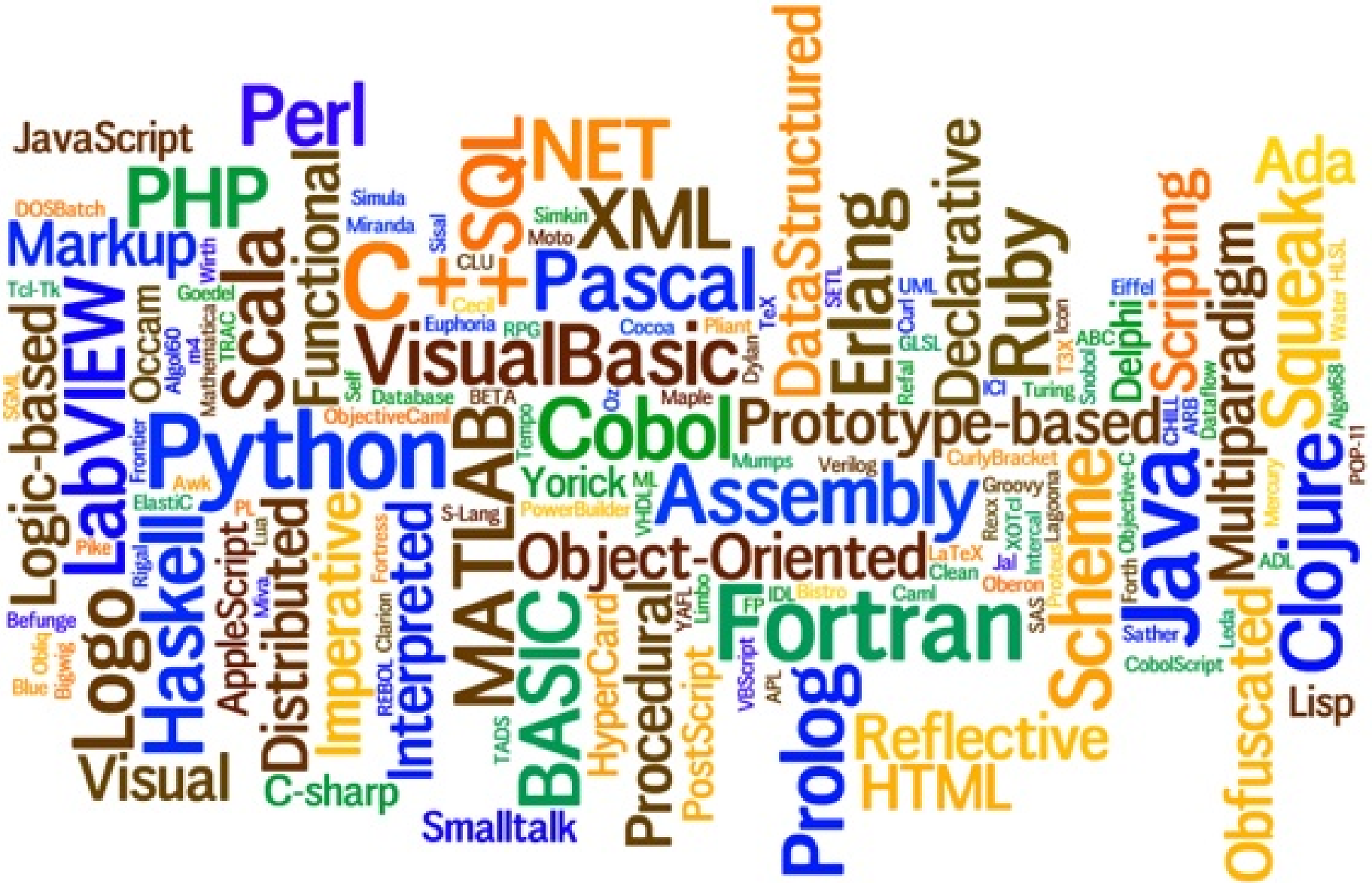
  ... dans tous les domaines de l'ingénierie

- ~~Problème~~ sur le système réel $\longleftarrow$ ~~Problème~~

  ... en informatique uniquement ! (si le modèle est fidèle)

Les programmes ne sont pas écrits en « maths »

... mais dans des langages de programmation variés



Encore une fois, les maths à la rescousse

Sur chaque langage de programmation :

- On peut exprimer la sémantique sous forme mathématique (existe, par exemple, pour Java)
- On peut prouver qu'un programme est exempt de bugs

Mais les preuves sont énormes, complexes et ... rarement automatisables

Théorème (Rice, 1953)

Toute propriété non triviale sur le langage reconnu par une machine de Turing est indécidable (non vérifiable par calcul).

Plan

En pratique : 2 alternatives pour contourner ces problèmes

1. Prouver des propriétés plus simples sur les programmes
   - Programmes prouvant l'absence d'erreur à l'exécution (ex. div. par 0)
   - Exemple : vérification du système de navigation des Airbus A340/380 (programme d'un million de lignes en langage C)

2. Utiliser des langages ayant des sémantiques plus simples

Le lambda-calcul (A. Church 1936) : langage simple et mathématique

- Limité à deux constructions :

  l'abstraction : $\lambda\, x.\, t$ l'application : $[f\ v]$

- Limité à une règle de sémantique : β-réduction

$$[(\lambda\, x.\, t)\ a] \twoheadrightarrow_\beta t\{x \mapsto a\}$$

(A comparer avec la sémantique de Java
`http://docs.oracle.com/javase/specs/jls/se7/html/index.html`)

Programmer et prouver en lambda-calcul ?

Certains langages de programmation sont inspirés du lambda-calcul :

Lisp, OCaml, Haskell, Scheme, SML, etc.

Certains respectent strictement le lambda-calcul et facilitent les preuves :

Langages des assistants de preuve **Coq, Isabelle/HOL, PVS**, etc.

Peut-on tout programmer avec le lambda-calcul strict ?

- En théorie oui : permet d'exprimer tout programme (prouvé en 1936 !)

  Il faut tout redéfinir à partir de zéro : *e.g.* les nombres, l'addition, ...

  (Démo Isabelle)

- ... mais pas forcément adapté pour programmer n'importe quoi !

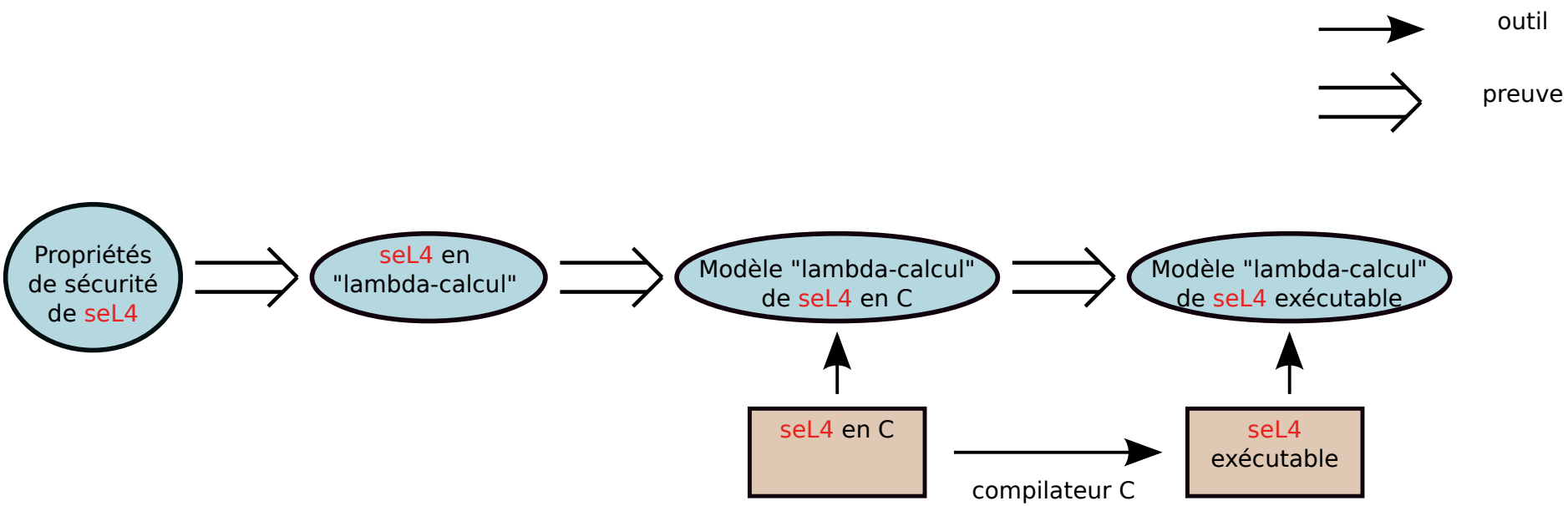
Pourrait-on re-programmer Android en lambda-calcul pour éradiquer les virus ?

Est-ce réaliste ? ... Est-ce nécessaire ?

Programmer et prouver en lambda-calcul ? (II)

Programmes en « lambda-calcul » qui ont été prouvés

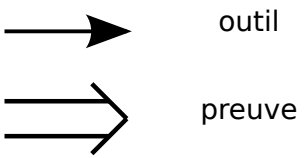
- Noyau de système d'exploitation sécurisé : seL4 en Isabelle (NICTA)
- Compilateur C : CompCert en Coq (INRIA/IRISA pour Airbus)

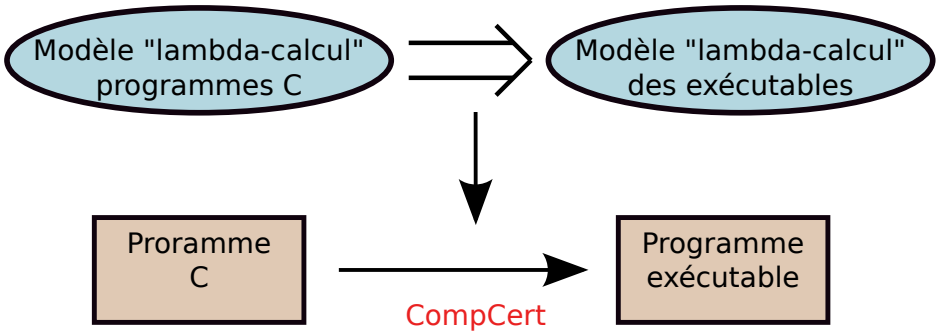


Programmer et prouver en lambda-calcul ? (II)

Programmes en « lambda-calcul » qui ont été prouvés

- Noyau de système d'exploitation sécurisé : seL4 en Isabelle (NICTA)
- Compilateur C : CompCert en Coq (INRIA/IRISA pour Airbus)





Conclusion

- La preuve de programme est utilisée dans les domaines critiques
  - Ligne 14 du métro parisien
  - Code du système de navigation des Airbus A340/380
  - Compilateur C pour Airbus
- Elle est en cours d'adoption là où la sécurité est importante
  - Protocoles de communication sécurisée sur internet
  - Noyaux de systèmes d'exploitation
  - ...
- Elle s'imposera dans l'informatique grand public... (votre téléphone)

  ... quand les usagers ne voudront plus de Virus, Malwares etc.

Un distributeur de billets perturbé par un bug

Publié le 13 mai 2009. 0 contributio

Like Tweet +1

A Plus gros | Plus petit

NEWSLETTER

HIGH-TECH

Recevez **une fois par semaine** toute l'actualité high-tech

Email

Je m'abonne

Un bug informatique a perturbé la semaine dernière à Lorient le fonctionnement d'un distributeur automatique de billets qui s'est alors mis à donner deux fois plus d'euros que demandé, selon l'édition de Ouest-France de ce mardi.

"J'ai voulu retirer 40 euros, le distributeur de billets m'en a donné 80" témoigne un des clients dans le quotidien. Le bouche à oreille aidant, les habitants ont fait la queue devant le distributeur. Lors de cette soirée, une centaine d'opérations ont été comptabilisées pour un montant d'environ 10 000 euros.

"Les quatre cassettes disposées dans l'automate distribuent des coupures différentes. Ce soir-là, deux cassettes étaient