National Institute of
Advanced Industrial Science
and Technology
AIST

Research Center for
Information Security

Fail-Safe C: 安全性を保証する実用C言語コンパイラの開発

大岩　寛

National Institute of Advanced Industrial Science and Technology
AIST

Fail-Safe C: 概要

Research Center for Information Security

- メモリ安全性を保証する ANSI-C コンパイラ
  - 100% ANSI-C 規格互換
    - ANSI-C 準拠したプログラムは完全に動作
  - 100% メモリ安全性を保証
    - 仮にプログラムに脆弱性があっても実行中のコードが書き換えられる(≒ウィルスの侵入)ことを完全に防止
      - Java, ML, Lisp などと同等の安全性を確保

この2つを同時に満たす処理系は他に例がない

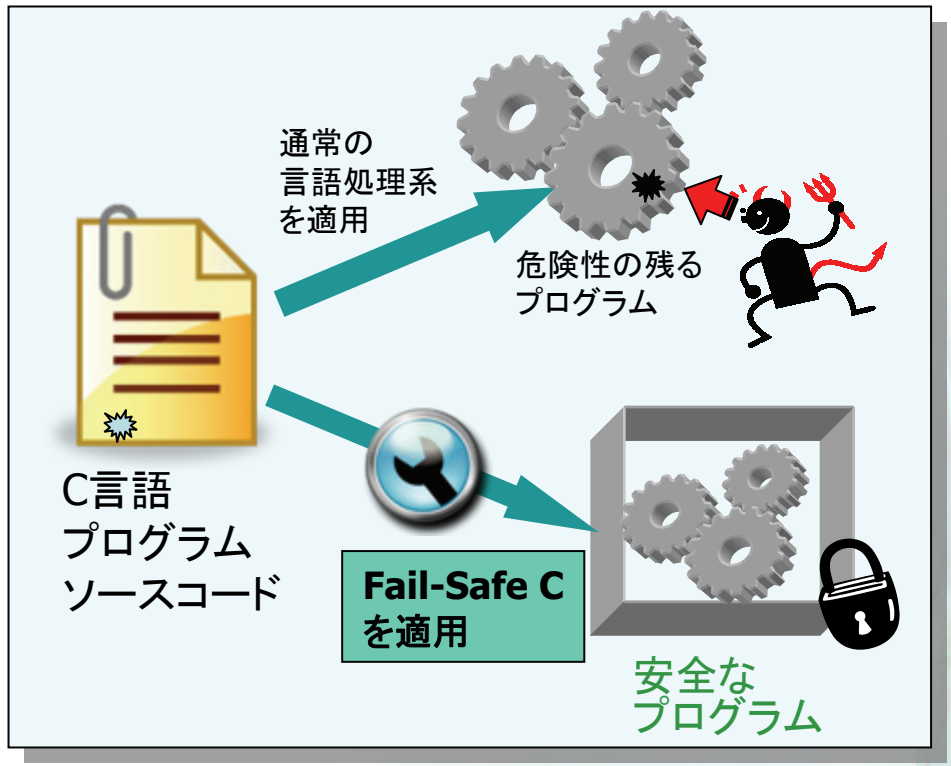
  - 様々な実用サーバアプリケーションが動作

National Institute of Advanced Industrial Science and Technology
AIST

Fail-Safe C の利用イメージ

Research Center for Information Security

- 既存プログラムに Fail-Safe Cを適用
  - 自動的に、全ての メモリ破壊脆弱性を 検知し防止する安全な プログラムを生成
  - サーバプログラム等に 事前に適用しておけば、 仮に未知の脆弱性が 発見され攻撃されても 被害を最小限に抑える ことができる



National Institute of Advanced Industrial Science and Technology AIST

動作確認済み サーバプログラムの例

Research Center for Information Security

- OpenSSL – 暗号通信ライブラリ
- OpenSSH – 暗号遠隔ログイン
- BIND9 (named) – 標準ネームサーバ
- thttpd – 軽量Webサーバ
- Postfix*, qmail* – 実用メールサーバ

- これらのプログラムが本質的な書き換え無しで動作
  - *: これらの動作のための改良はまもなく公開予定

National Institute of Advanced Industrial Science and Technology
AIST

現在のシステム

Research Center for Information Security

- 実装をホームページで公開中
  - https://www.rcis.aist.go.jp/project/FailSafeC-ja.html
  - x86 (32bit) Linux 対応
    - 組み込み (MIPS, ARM など) 向け実装も近日公開
    - x86-64 は今後対応する予定
  - コンパイラ本体の他、リンカや500以上のライブラリ部品を整備
  - 分割コンパイルなどもきちんとサポート
  - 詳細はポスターまたは論文 (PLDI2009) にて

Fail-Safe C: 安全性を保証する実用C言語コンパイラの開発

大岩　寛 (ソフトウェアセキュリティ研究チーム)

Fail-Safe C Compiler: C言語セキュリティの切り札

- **100% ANSI C89 規格 (JIS規格) 上位互換**
  - ポインタ演算, キャスト, 共用体等 C言語特有の操作に全て対応
  - プログラムの書き換えが基本的に不要
- **100% メモリ安全**
  - どんなプログラムの誤りがあっても プログラム実行中のメモリを破壊しない
    - Java, C#, Lisp, ML と同等の安全性を確保

この2つを同時に100%両立させたシステムは研究レベルでも他に例がない

- C言語プログラム特有の記述を多数サポート
  - 既存のプログラムとの互換性を可能な限り確保 (厳密にANSI C 規格に準拠しないプログラムが多い)
- 可能な限り高速に動作

研究の背景

- インターネットの普及により セキュリティ脆弱性の脅威が増大
  - 特に、C言語プログラムのメモリ脆弱性は サーバ乗っ取りに直結する
    - 「任意コード脆弱性」の大半の原因
- 一方で、C言語プログラムはいまだに 広く使い続けられている
  - 安全な言語への乗り換えは時間がかかる

➔今既に存在するC言語プログラムを できるだけ簡単に安全にする技術が必要

内部の実装技術

(1) 型を使ったメモリ管理

- メモリ領域をブロック単位でオブジェクト化
  - 要素数に加え、データ型の情報をブロックに記録
  - 基本的なメモリの読み書き操作をメソッドとして付加

キャスト操作があっても安全なメモリ操作を実現

(2) Smart ポインタ と Cast フラグ

- ポインタを内部的に2ワードで表現
  - 参照先のオブジェクトと内部オフセットを常に記録
  - さらに、キャストの有無を示すフラグを付加

キャストがない場合は高速なメモリアクセスを実現

- 仮想化による既存プログラムとの高い互換性

現在のシステム

- ホームページから一般に公開
- I386 の Linux 上で動作
  - MIPS*, ARM* 等でも動作 (ネイティブ、クロス) (*実装の公開を準備中)
- 従来互換のユーザインタフェースを提供
  - Gcc の代わりに "fscc" を用いるだけで動作
  - 分割コンパイルもきちんとサポート
    - モジュール間整合性のリンク時検査機構実装
- 安全なC言語標準ライブラリ
  - POSIX 規格の500以上の関数をサポート
  - 関数1つ1つに対応した 安全性検査付きの実装を作成
  - 関数の誤った使用をきちんと検出・ メモリ破壊を防止
- 複数の実用アプリケーションが動作
  - OpenSSL – 暗号通信ライブラリ
  - OpenSSH – 暗号遠隔ログイン
  - BIND9 (named) – 標準ネームサーバ
  - thttpd – 軽量Webサーバ
  - Postfix*, qmail* – 実用メールサーバ

などがほとんどプログラム書き換え無しに動作

- 性能: 計算時間で平均 3~5倍程度
  - プログラムにより 1.01~7 倍程度
  - 最適化による更なる改善を予定

今後の展開：実用化へ向けて

- 処理系の更なる効率化・利便性の向上
  - 静的解析による最適化の導入
  - 支援ツール類の実装 など
- 対応環境の拡大
  - 64bit (x86-64 他) 環境への対応
  - Linux 以外の環境への対応
- 実用化を目指した取り組み
  - 1-CD 環境・アプライアンス等への適用
  - OS 配布などへの統合による簡単な導入
  - 実際に安全にしたソフトウェアの配布
- その他 企業等への働きかけと導入サポート
  - 知見を基に実用システムのための改善に反映
  - 興味がありましたらぜひお声をおかけ下さい

■ ホームページ: http://www.rcis.aist.go.jp/project/FailSafeC-ja.html or http://failsafec.jp/

* 本研究の実装の一部は、経済産業省「新世代情報セキュリティ研究開発事業」の一部として株式会社レピダムと共同で行いました。
* MIPS, ARM 対応およびクロスコンパイラ対応は、科学技術振興調整費「組込みシステム向け情報セキュリティ技術」研究の一部として行いました。

Technical Reference: Yutaka OIWA. Implementation of the Memory-safe Full ANSI-C Compiler. In *Proc. PLDI 2009*, June 2009. To appear.