イントロダクション

Linuxでも正式サポート，
組み込みや車載でも注目を集める

# Rustと歩む未来

高野 祐輝

## Rustは安全性と低レイヤ開発を兼ね備えた言語

### ● 安全性が高くC/C++の代替となりうる言語

Rust言語は，システム・ソフトウェア向けのプログラミング言語であり，C/C++の代替となりうるため，注目を集めています．

システム・ソフトウェア開発言語には，実行時のオーバヘッドが少なく，かつメモリを直接読み書きするような操作など，ハードウェアに近いレイヤの記述も行えることが求められています．

しかし，C/C++を利用すると，どうしても，バッファ・オーバランなどのバグをコード中に埋め込んでしまいます．バッファ・オーバランやぶら下がりポインタ（無効なメモリ領域を指すポインタ）などに関する安全性は，メモリ安全性と呼ばれています．C/C++はメモリ安全でないプログラミング言語に分類されます．

Rustはメモリ安全かつ，システム・ソフトウェアで必要とされる特徴を兼ね備えた，商用レベルのソフトウェアでも利用可能なプログラミング言語です．

### ● Linuxでも正式サポートされた

2022年12月にリリースされたLinuxカーネル 6.1では，Rustのサポートを正式に開始し，Linuxカーネル・モジュールやデバイス・ドライバがRustで実装できるようになりました．Linuxカーネル6.1は記念碑的なリリースであるとともに，これにより，今後ますますRustの重要性が増していくことが予想されます．

当然この流れは組み込みLinuxをはじめとして，組み込みソフトウェア分野にも波及していくと思われます．「30億のデバイスで走るRust」の時代が迫ってきているのです．

## Rust言語の特徴

### ● バッファ・オーバランはほぼ防げる

Rust言語には幾つもの魅力的な特徴がありますが，筆者はやはり，その安全性が最も重要だと考えています．

Stuxnet[1]は，2010年ころに発見された，制御システムを標的とするマルウェアで，核施設を攻撃したことで大きな話題になりました．2021年には，水道や交通機関などのインフラ監視システムを対象とした，Stuxnetベースのマルウェアである Duqu[2]が登場しています．Duquは，TrueTypeフォント・パーサの脆弱性[3]を用いてゼロデイ攻撃を行いました．この脆弱性の原因は，バッファ・オーバランだと言われており，これはRustではほぼ防ぐことができるバグの一種です．

### ● 安全性のためには全体的な底上げが必要

セキュリティを考える上では，脅威分析やリスク評価が重要だと言われています．これらの手法では，リスクの大きさを判定して，リスクが大きいものを優先して対策していきます．しかし，誰がTrueTypeフォント・パーサ実装のリスクが大きいと考えるのでしょうか．

この件から得られる教訓は，安全性については，とにかく全体的な底上げをしなければ，その向上は難しいということです．

Rustは，安全性の求められるシステムに向いていると言われていますが，個人的には，安全性を求めるには全体ソフトウェア開発で底上げしなければならず，全体的にRustのような安全な言語を用いて実装すべきと考えています．この実現には，Rustが第1の選択肢となり得るでしょう．

### ● 強力な開発環境も整っている

Rustには，安全性以外にも多くの特徴があります．それを端的に表したのが**図1**です．

この図は，Visual Studio Code（VSCode）というエ